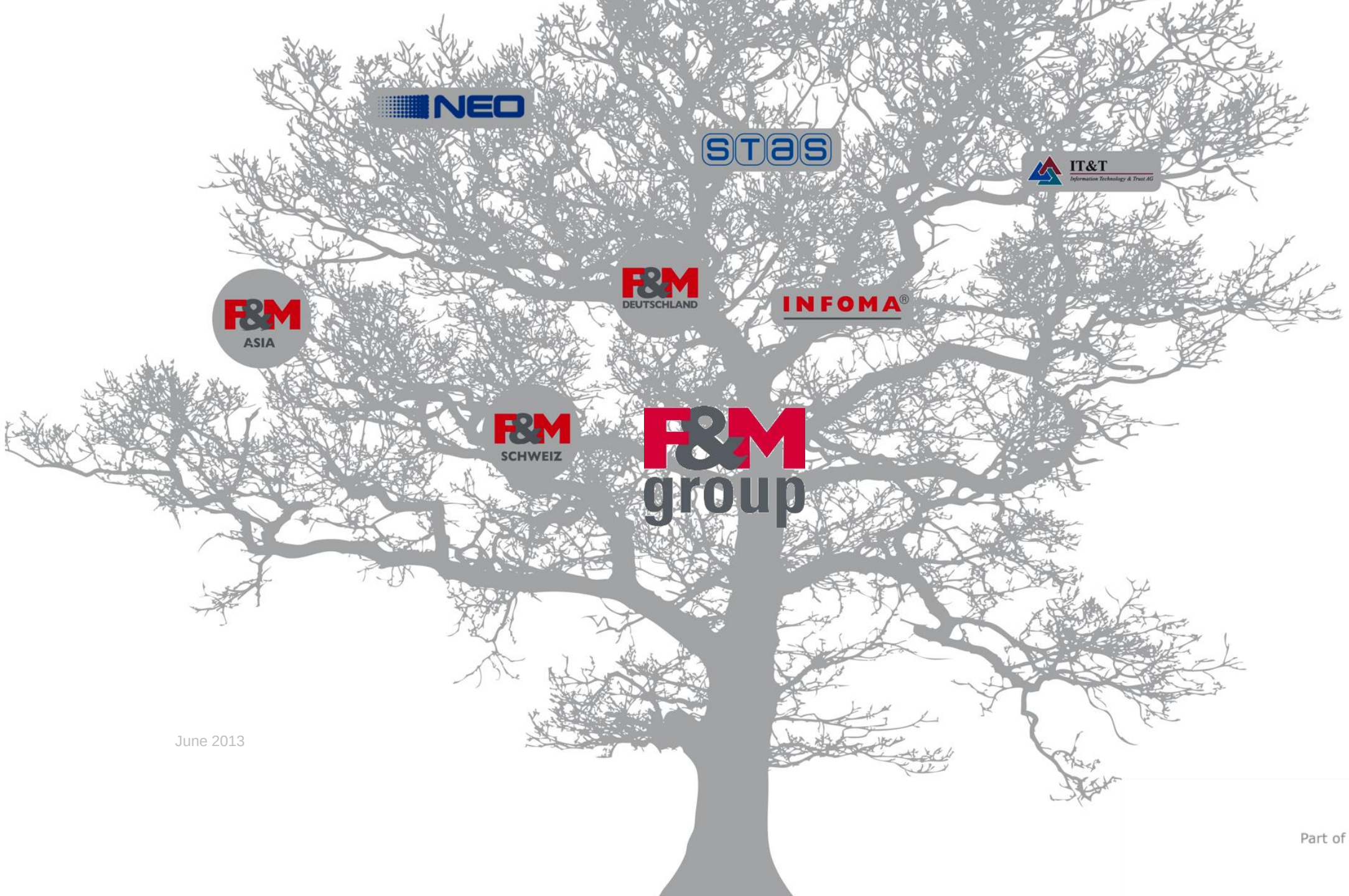
Sparen Sie Zeit bei der Administration von IBM Connections

Scripting für IBM WebSphere und DB2

Christoph Stöttner

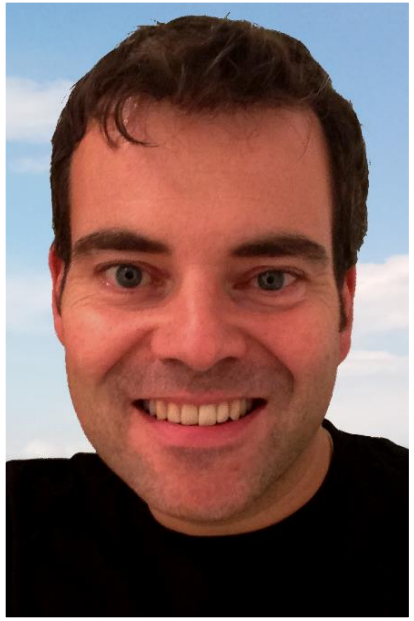
Fritz & Macziol Software & Computervertrieb GmbH
http://www.fum.de
cstoettner@fum.de
+49 89 4567858-90



June 2013

Part of Imtech

>> Wer bin ich?



Christoph Stöttner
IBM Software Consultant

- IBM Connections seit 2.5
- Domino / Windows / Linux Admin
- Mein Blog: http://www.stoeps.de
- Twitter: @stoeps
- Mail: cstoettner@fum.de
- mehr Informationen: http://about.me/stoeps

>> Agenda

1. Administration von IBM Connections

- Integrated Solution Console
- wsadmin

2. WebSphere Application Server

- Jython
- wsadmin properties
- praktische Scripte

3. DB2 – Backup

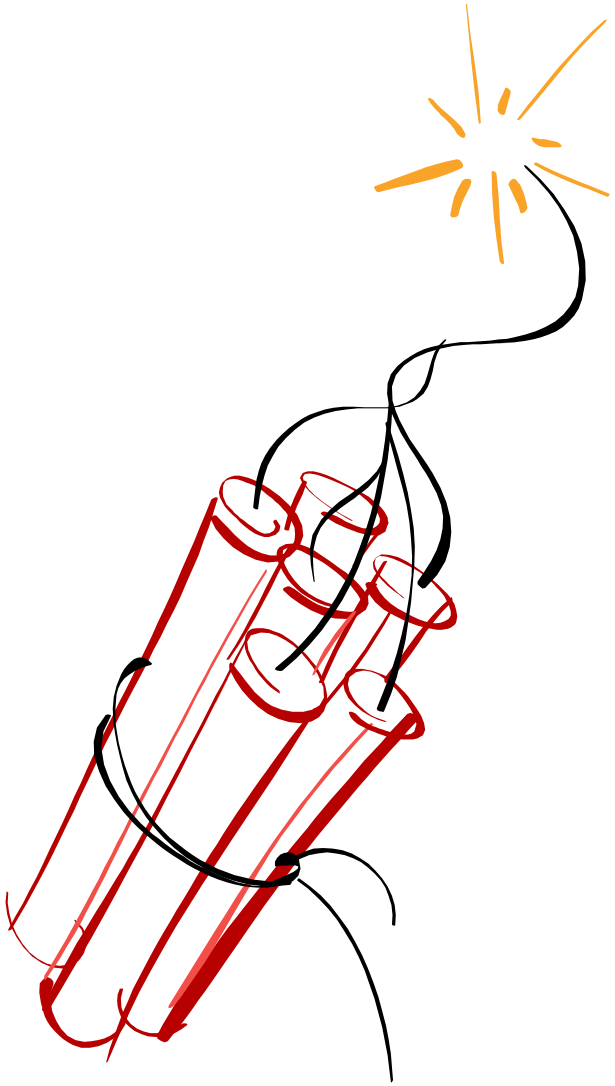
- Automatic Maintenance
- praktische Skripte

4. Und noch mehr

- Root CA

>> Achtung

- Mit Skripten
  - Shell / bash / zsh / ksh
  - Jython
  - Powershell / Batch
  - SQL
- kann man ...
  - Viel Zeit sparen!
  - **Sehr viel in sehr kurzer Zeit ändern!**



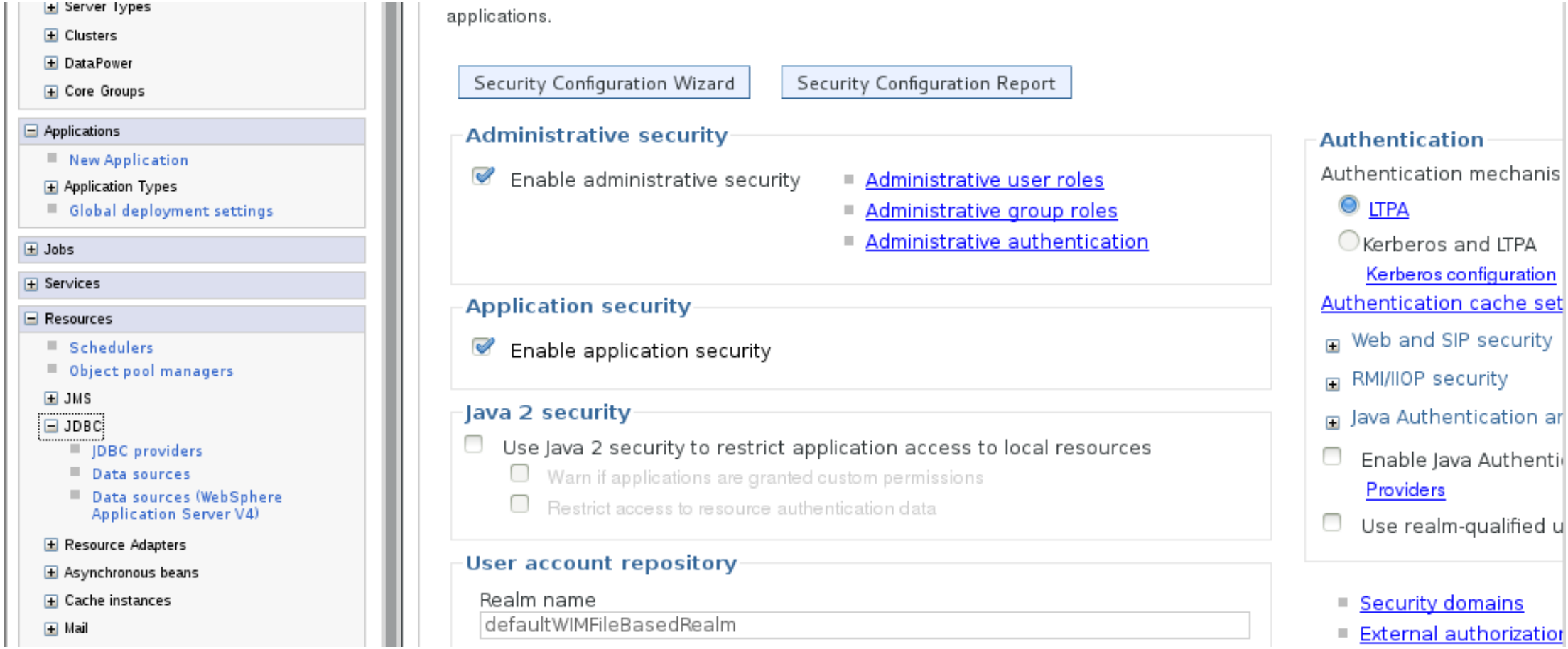
>> Disclaimer

Die Benutzung aller Skripte auf diesen Folien und in den angegebenen Download Repositories erfolgt auf eigene Gefahr und ohne Garantie!

- Tipps:
  - Seien Sie vorsichtig!
  - Lieber doppelt überprüfen!
  - Backups
  - Test- oder Entwicklungssystem
  - Dokumentation

>> IBM Connections Administration

>> Integrated Solutions Console



>> Save the mice!

>> Integrated Solution Console

- Browserbasierte Oberfläche für die Konfiguration von IBM WebSphere Application Server
- Während einer Connections Installation / Konfiguration legt mein Mauszeiger Meilen zurück
  - 90% davon als Postinstall Tasks innerhalb der ISC
- Einige Tätigkeiten nerven und sind langweilig!
  - Performance Tuning der DataSources
  - Konfiguration der Security / J2EE Rollen (Connections + FEB + Docs + CCM = 24 Apps)
    - Man übersieht schnell eine Applikation oder DB

☐	metrics-reader	All Authenticated in Application's Realm		
☐	community-creator	All Authenticated in Application's Realm		
☐	community-metrics-run	All Authenticated in Application's Realm		
☐	search-admin	None	wasadmin AConnections	CNXAdmins
☐	global-moderator	None	wasadmin Aconnections	CNXModerators

>> wsadmin: Konfiguration von Connections

- Export und Validierung der Konfigurationsdateien (Bsp. LotusConnections-config.xml)
  - Aufruf wsadmin

```
cd /opt/IBM/WebSphere/AppServer/profiles/Dmgr01/bin
[root@cnxwas1 bin]$ ./wsadmin.sh -lang jython -username wasadmin -password password
```

  - Name der Zelle, laden der Connections Befehle

```
WASX7209I: Connected to process "dmgr" on node cnxwas1CellManager01 using SOAP
connector; The type of process is: DeploymentManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>AdminControl.getCell()
'cnxwas1Cell01'
wsadmin>execfile("connectionsConfig.py")
Connections Administration initialized
wsadmin>LCConfigService.checkOutConfig('/tmp','cnxwas1Cell01')
# Edit /tmp/LotusConnections-config.xml and save your changes
wsadmin>LCConfigService.checkInConfig('/tmp','cnxwas1Cell01')
Loading schema file for validation: /tmp/LotusConnections-config.xsd
Loading schema file for validation: /tmp/service-location.xsd
/tmp/LotusConnections-config.xml is valid
Connections configuration file successfully checked in
```

>> wsadmin: ExID mit LDAP synchronisieren

- Synchronisierung der Benutzer über die Applikationen
- News

```
wsadmin>execfile("newsAdmin.py")
Connecting to NewsMemberServiceName: News Configuration Environment initialized
wsadmin>NewsMemberService.syncMemberExtIdByEmail("cstoettner@stoeps.local")
syncMemberExtIdByEmail request processed
```

- Blogs

```
wsadmin>execfile("blogsAdmin.py")
Connecting to {...} Blogs Administration initialized
wsadmin>BlogsMemberService.syncMemberExtIdByEmail("cstoettner@stoeps.local")
WASX70115E: Exception running command:
"BlogsMemberSErvice.syncMemberExtIdByEmail("cstoettner@stoeps.local")";
exception information:
There is no member associated with this email address or login name:
cstoettner@stoeps.local
wsadmin>BlogsMemberService.syncMemberExtIdByEmail("CStoettner@stoeps.local")
syncMemberExtIdByEmail request processed
```

Case sensitiv

>> wsadmin

- Unterstützt JACL und JYTHON
- lange Befehle
- Groß-/Kleinschreibung
  - Jython / JACL Befehle
  - Parameter
- Linux: keine Historie oder Wiederaufruf von Eingaben
- Vorsicht bei Copy & Paste aus Webseiten
  - " oft falsch formatiert
  - Securityproblem format c:
- Tipp: "Cheatsheet" erstellen
  - häufig benutzte Befehle in einer Textdatei vorbereiten

WebSphere Application Server Scripting

>> wsadmin Aufruf und Einstellungen

- wsadmin immer im Ordner "bin" des Deployment Manager Profils ausführen!
  - `cd {WAS_HOME}/profiles/Dmgr01/bin`
- Linux | AIX
  - `./wsadmin.sh -lang {jython | jacl} -username wasadmin -password password`
- Windows
  - `wsadmin.bat -lang {jython | jacl} -username wasadmin -password password`
- Alias oder Shell Variable erstellen
  - schnellerer Zugriff
  - erspart Tippfehler
  - `alias wsadmin='cd {WAS_HOME}/profiles/Dmgr01/bin;./wsadmin.sh -lang jython'`

>> Beispiel .bashrc

```
PATH=$PATH:/opt/IBM/WebSphere/AppServer/java/jre/bin/
WAS_HOME=/opt/IBM/WebSphere/AppServer
DMGR=Dmgr01
APPSRV=AppSrv01

alias dmgrBin='cd $WAS_HOME/profiles/$DMGR/bin'
alias wsadmin='cd $WAS_HOME/profiles/$DMGR/bin;./wsadmin.sh -lang jython'
alias nodeBin='cd $WAS_HOME/profiles/$APPSRV/bin'
alias startNode='$WAS_HOME/profiles/$APPSRV/bin/startNode.sh'
alias startDmgr='$WAS_HOME/bin/startManager.sh'
alias stopNode='$WAS_HOME/profiles/$APPSRV/bin/stopNode.sh'
alias stopDmgr='$WAS_HOME/bin/stopManager.sh'
alias nodeLog='tail -f $WAS_HOME/profiles/$APPSRV/logs/nodeagent/SystemOut.log'
alias InfraClusterLog='tail -f $WAS_HOME/profiles/$APPSRV/logs/InfraCluster_server1/SystemOut.log'
alias Cluster1Log='tail -f $WAS_HOME/profiles/$APPSRV/logs/Cluster1_server1/SystemOut.log'
alias Cluster2Log='tail -f $WAS_HOME/profiles/$APPSRV/logs/Cluster2_server1/SystemOut.log'
```

>> wsadmin: Standardsprache ändern

`edit {WAS_HOME}/profiles/Dmgr01/properties/wsadmin.properties`

- Default:
  - `com.ibm.ws.scripting.defaultLang=jacl`
- Ändern zu:
  - `com.ibm.ws.scripting.defaultLang=jython`

```
#---------------------------------------------------------------------
# The defaultLang property determines what scripting language to use.
# Supported values are jacl and jython.
# The default value is jacl.
#---------------------------------------------------------------------
com.ibm.ws.scripting.defaultLang=jython
```

>> wsadmin – Login

WAS_HOME}/profiles/Dmgr01/properties/soap.client.props

- **Verringert die Sicherheit!**
  - com.ibm.SOAP.securityEnabled=true
  - com.ibm.SOAP.loginUserid=wasadmin
  - com.ibm.SOAP.loginPassword=password
- PropFilePasswordEncoder.sh soap.client.props com.ibm.SOAP.loginPassword

  → com.ibm.SOAP.loginPassword={xor}Lz4sLCgwLTs=

```
com.ibm.SOAP.securityEnabled=true

#-----------------------------------------------------------------------
# - authenticationTarget     ( BasicAuth[default], KRB5. These are the only sup
#                              on a pure client for JMX SOAP Connector Client.
#-----------------------------------------------------------------------
com.ibm.SOAP.authenticationTarget=BasicAuth

com.ibm.SOAP.loginUserid=wasadmin
com.ibm.SOAP.loginPassword={xor}Lz4sLCgwLTs=
```

>> WebSphere Password Decoding

- KEINE Passworte für produktive Umgebungen speichern!
- WebSphere Hashes sind xor mit "_" und dann base64 kodiert
- Verschlüsselung ist umkehrbar.
  - Verschiedene Webseiten: z.B. http://www.sysman.nl/wasdecoder

```
cd WAS_HOME

java/jre/bin/java \
-Djava.ext.dirs=deploytool/itp/plugins/com.ibm.websphere.v8_1.0.201.v20111031_1843/wasJars \
-cp securityimpl.jar:iwsorb.jar com.ibm.ws.security.util.PasswordDecoder \
"{xor}Lz4sLCgwLTs="

encoded password == "{xor}Lz4sLCgwLTs=", decoded password == "password"
```

>> Connections Administration mit wsadmin

- Jede Applikation benötigt eigene Befehle
- Diese werden innerhalb von wsadmin über execfile geladen

```
[root@cnxwas1 bin]# ./wsadmin.sh -lang jython -username admin -password passwor
WASX7209I: Connected to process "dmgr" on node cnxwas1CellManager01 using SOAP c
WASX7031I: For help, enter: "print Help.help()"
wsadmin>synchAllNodes()
WASX7015E: Exception running command: "synchAllNodes()"; exception information:
 com.ibm.bsf.BSFException: exception from Jython:
 Traceback (innermost last):
   File "<input>", line 1, in ?
   NameError: synchAllNodes
   wsadmin>execfile("connectionsConfig.py")
   Connections Administration initialized
   wsadmin>synchAllNodes()
   Nodes synchronized
```

>> Laden aller Connections Befehle

- Erstellen Sie ein Skript, das alle Befehle lädt
- Aufruf dieses Skripts:
  - Innerhalb von wsadmin:

    ```
    execfile("loadAll.py")
    ```
- Skript in com.ibm.ws.scripting.profiles aufrufen:

  ```
  {WAS_HOME}/profiles/Dmgr01/properties/wsadmin.properties
  ```
- Beim wsadmin Aufruf ausführen lassen:

  ```
  ./wsadmin.sh -lang jython -profile loadAll.py
  ```

>> loadAll.py

- Speichern in
  - `{WAS_HOME}/profiles/Dmgr01/bin`

```
execfile('connectionsConfig.py')
execfile("activitiesAdmin.py")
execfile("blogsAdmin.py")
execfile("communitiesAdmin.py")
execfile("dogearAdmin.py")
execfile("filesAdmin.py")
execfile("forumsAdmin.py")
execfile("homepageAdmin.py")
execfile("newsAdmin.py")
execfile("profilesAdmin.py")
execfile("wikisAdmin.py")
```

- **Achtung:**

  In Multinode Umgebungen wird beim Aufruf abgefragt, auf welchem Node dieser ausgeführt werden soll.

>> Jython

- Einfach zu lernen und leistungsfähig
- Python für Java
  - http://www.jython.org/jythonbook/en/1.0/
  - http://www.jython.org/docs/index.html
- Bücher
  - **WebSphere Application Server Administration Using Jython** (2009)
    Autoren: Robert A. Gibson, Arthur Kevin McGrath und Noel J. Bergman
  - **The Definitive Guide to Jython: Python for the Java Platform** (2010)
    Autoren: Josh Juneau, Frank Wierzbicki, Leo Soto und Victor Ng
- Web Ressourcen
  - mehrere kostenlose Pythonkurse (sehr ähnlich zu Jython)
  - http://www.codecademy.com und http://learnpythonthehardway.org/book/

Jython Grundlagen

>> Jython Grundlagen

- Gut lesbarer Code
- Shell und Command Line Interpreter
- Shell sehr praktisch zum Testen

```
[root@cnxwas1 bin]# wsadmin
WASX7209I: Connected to process "dmgr" on node cn
entManager
WASX7031I: For help, enter: "print Help.help()"
wsadmin>int2=10
wsadmin>str2="Chris"
wsadmin>print str2
Chris
wsadmin>10+int2
20
```

```
[root@cnxwas1 ~]# python
Python 2.6.6 (r266:84292, Feb 22 2013, 00:00:18)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-3)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> int1=10
>>> str1="Chris"
>>> print str1
Chris
>>> 10*int1
100
>>>
```

>> Jython Grundlagen: Variablen

- Typzuweisung unnötig
- Integer mit " oder '

```
# Defining a String
x = 'Hello World'
x = "Hello World Two"

#  Defining an integer
y = 10

#  Float
z = 8.75

# Complex
i = 1 + 8.07j

# Multiple assignment
x, y, z = 1, 2, 3
```

>> Jython Grundlagen: Range

- praktisch für die Benutzung in Schleifen
- range startet mit 0!

```
wsadmin>range(7)
[0, 1, 2, 3, 4, 5, 6]

# Include a step in the range
wsadmin>range(0,10,3)
[0, 3, 6, 9]

# Good base for loops
wsadmin>range(1,11)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

wsadmin>range(20,27)
[20, 21, 22, 23, 24, 25, 26]
```

>> Jython Grundlagen: Lists und Dictionaries

- Liste

```
#List
wsadmin>dbs = ['activities','blogs','communities','dogear','files','forum']
wsadmin>dbs[1]
'blogs'
```

- Dictionary

```
# Dictionary with Performance Data
wsadmin>minConnections = {'activities':1,'blogs':1,'communities':10,'dogear':1}
wsadmin>maxConnections = {'activities':50,'blogs':250,'communities':200}
wsadmin>maxConnections
{'communities': 200, 'activities': 50, 'blogs': 250}
wsadmin>maxConnections.keys()
['communities', 'activities', 'blogs']
wsadmin>maxConnections.values()
[200, 50, 250]
wsadmin>maxConnections['blogs']
250
```

>> Jython Grundlagen: if – elif - else

- Code mit 4 Leerzeichen einrücken, um diesen zu gruppieren

```
# Basic
if condition :
    # print or do something
elif other condition :
    # print or do something other
else :
    # print or do completely different
```

- Beispiel

```
# Example
if value.find( 'CLFWY0217E' ) > -1 :
    print "\t\tuser already converted"
elif value.find( 'CLFWY0212E' ) > -1 :
    print "\t\tuser not found in database"
elif value.find( ' CLFWY0209E' ) > -1 :
    print "\t\tnew identifier '" + data[1] + "' does not exist."
else :
    print '\t\tException value: ' + value
```

>> Jython Grundlagen: Schleifen

- For Schleife

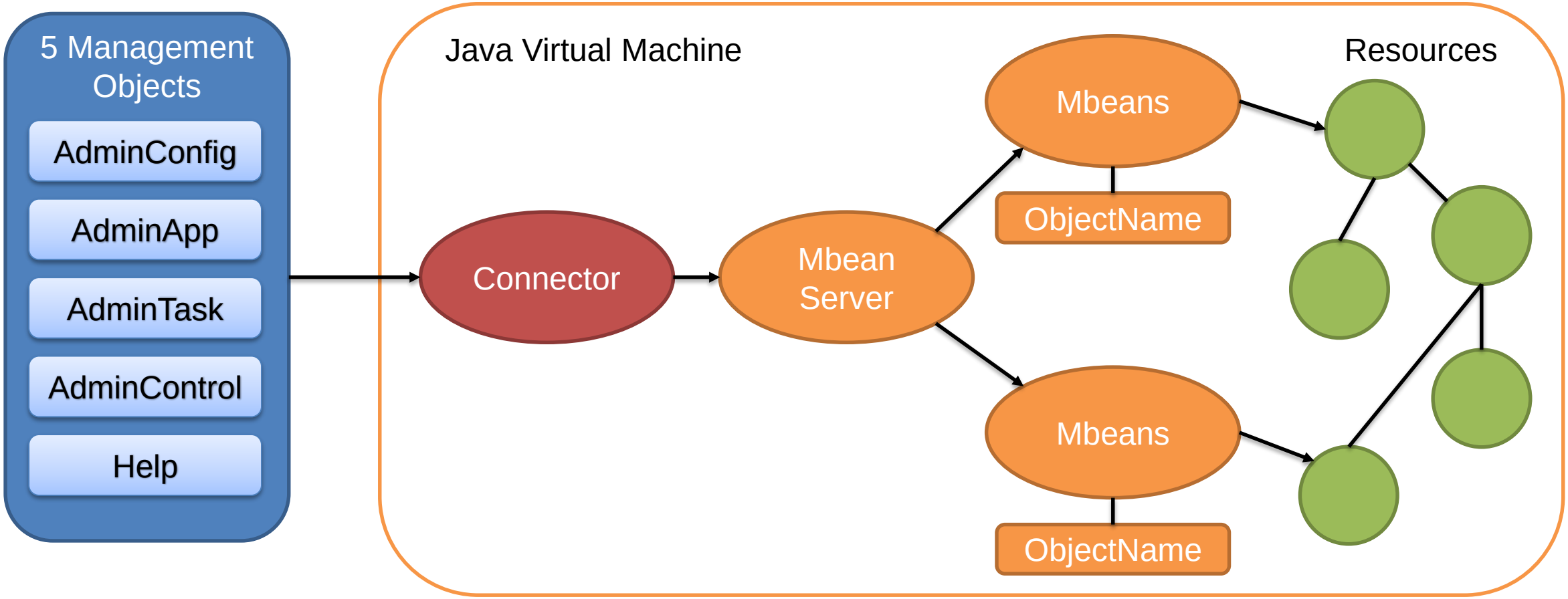
```
# For Loops
dbs = ['activities','blogs','communities','dogear','files','forum','homepage']
for db in dbs: #loop through databases
    print "Database %s" % db
```

- While

```
# While
x = 0
y = 3
while x <= y :
    print 'Value of x is: %d' %(x)
    x += 1
else:
    print 'Completed!'

Value of x is: 0
Value of x is: 1
Value of x is: 2
Value of x is: 3
Completed!
```

>> wsadmin Befehle



>> Fünf Management Objects

- AdminApp:
  - Anwendung installieren, ändern und verwalten
- AdminConfig:
  - Konfiguration des Websphere Application Servers ändern
  - Elemente in der Konfiguration erstellen (Datenquelle, J2C Alias etc)
- AdminControl:
  - Verwaltung der Anwendungsserverobjekte
- AdminTask:
  - Verwaltungsbefehle, Befehlsgruppen
- Help:
  - Hilfe zu den benutzten Objekten aufrufen

>> Beispiele wsadmin Objekte

- AdminControl.getCell()
  - Ausgabe des Cellnames
- AdminConfig.list('Server')
  - alle Application Server der Zelle
- AdminControl.getNode()
  - aktueller Nodename (DMGR)
- AdminConfig.list('Node')

```
[root@cnxwas1 ~]# wsadmin
WASX7209I: Connected to process "dmgr" on node cn:
WASX7031I: For help, enter: "print Help.help()"
wsadmin>AdminControl.getCell()
'cnxwas1Cell01'
wsadmin>AdminControl.getNode()
'cnxwas1CellManager01'
wsadmin>
```

```
wsadmin>nodes = AdminConfig.list( 'Node' ).splitlines()
wsadmin>for node in nodes:
wsadmin>    print node
wsadmin>
cnxdocsNode01(cells/cnxwas1Cell01/nodes/cnxdocsNode01|node.xml#Node_1)
cnxwas1CellManager01(cells/cnxwas1Cell01/nodes/cnxwas1CellManager01|node.xml#Node_1)
cnxwas1Node01(cells/cnxwas1Cell01/nodes/cnxwas1Node01|node.xml#Node_1)
webserverNode(cells/cnxwas1Cell01/nodes/webserverNode|node.xml#Node_1371545639179)
```

>> AdminApp

- Die folgenden Befehle sind für das Objekt AdminApp verfügbar:
  - deleteUserAndGroupEntries
  - edit
  - editInteractive
  - export
  - exportDDL
  - exportFile
  - getDeployStatus
  - install
  - installInteractive
  - isAppReady
  - list
  - listModules
  - Optionen
  - publishWSDL
  - searchJNDIReferences
  - taskInfo
  - uninstall
  - update
  - updateAccessIDs
  - updateInteractive
  - view

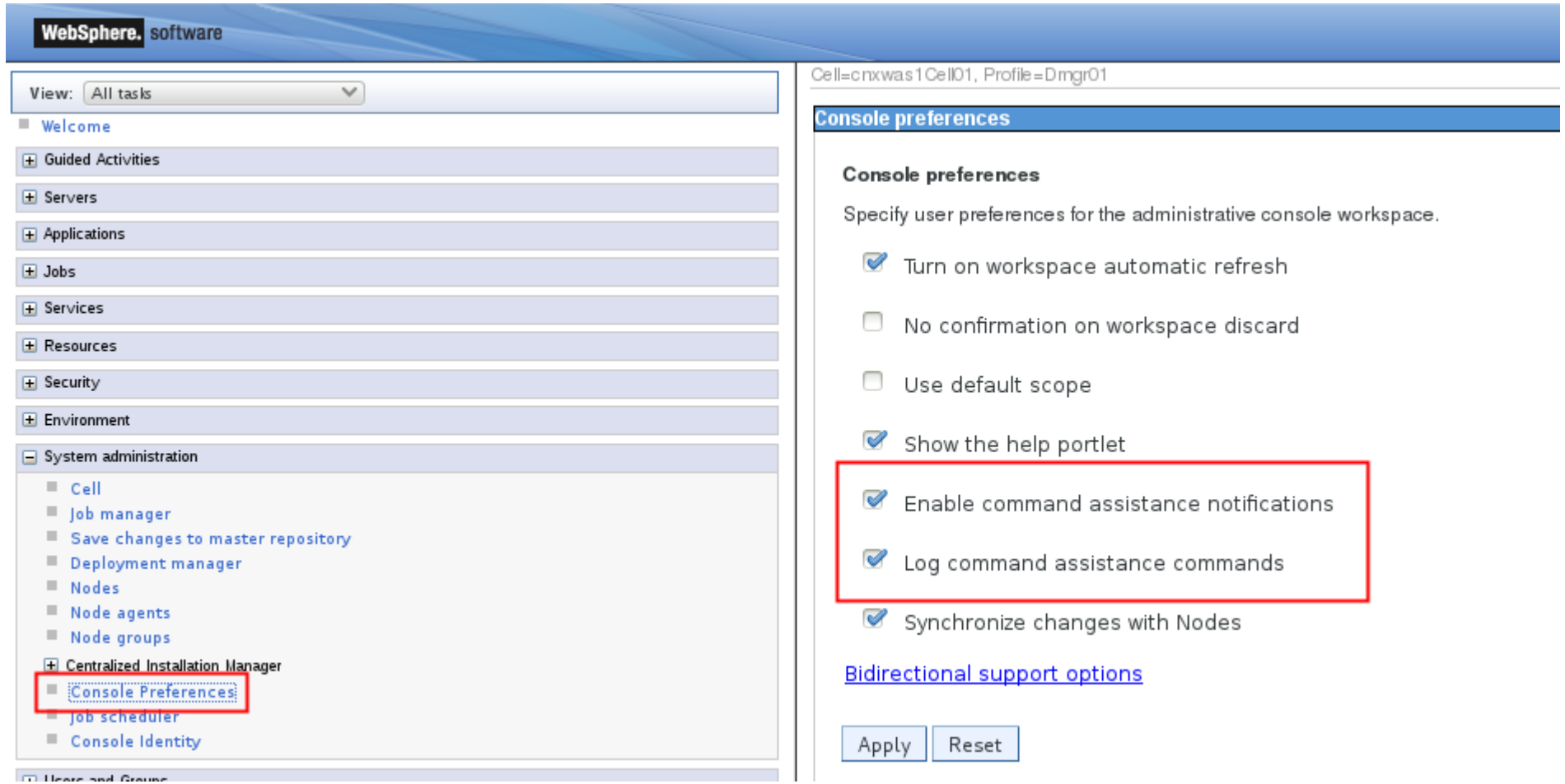
>> AdminConfig (Auszug)

- attributes
- create
- createDocument
- createUsingTemplate
- deleteDocument
- existsDocument
- getCrossDocumentValidationEnabled
- getid
- getObjectName
- getObjectType
- installResourceAdapter
- list
- modify
- queryChanges
- remove
- reset
- resetAttributes
- save
- show
- showall
- showAttribute
- validate

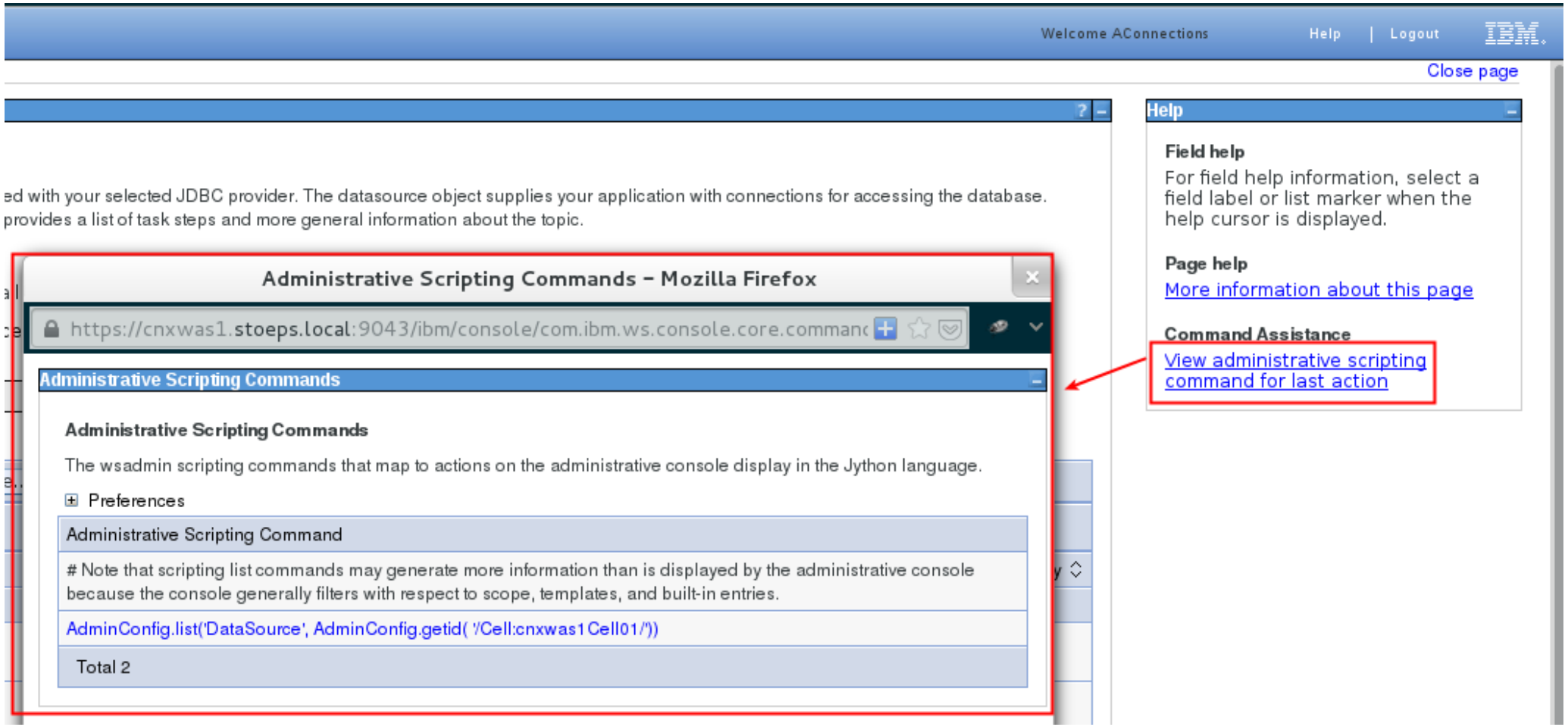
>> AdminControl (Auszug)

- completeObjectName
- getAttribute
- getAttributes
- getCell
- getConfigId
- getHost
- getNode
- getObjectInstance
- getPort
- getType
- isRegistered
- makeObjectName
- queryMBeans
- queryNames
- reconnect
- setAttribute
- setAttributes
- startServer
- stopServer
- testConnection
- trace

>> Finden der notwendigen Befehle



>> Enable command assistance notifications



>> Log command assistance commands

- {WAS_HOME}/profiles/Dmgr01/logs/dmgr/commandAssistanceJythonCommands_AConnections.log

```
AdminConfig.list('ServerCluster', AdminConfig.getid( '/Cell:cnxwas1Cell01/'))

# [9/20/13 12:45:36:204 CEST] WebSphere application server clusters
AdminControl.invoke('WebSphere:name=InfraCluster,process=dmgr,
platform=common,node=cnxwas1CellManager01,version=8.0.0.5,type=Cluster,
mbeanIdentifier=InfraCluster,cell=cnxwas1Cell01,spec=1.0', 'rippleStart')

# Note that scripting list commands may generate more information than is
# displayed by the administrative console because the console generally filters
# with respect to scope, templates, and built-in entries.

# [9/22/13 19:09:43:718 CEST] DataSource
AdminConfig.list('DataSource', AdminConfig.getid( '/Cell:cnxwas1Cell01/'))
```

Beispielskripte

>> Test Database Connection

- checkDataSource.py

```
# List of databases you want to check:
dbs = ['activities','blogs','communities','dogear','files','forum','homepage']
for db in dbs: #loop through databases
    ds = AdminConfig.getid('/DataSource:' + db + '/')
    checkDS = AdminControl.testConnection(ds)
    if checkDS == "WASX7217I: Connection to provided datasource was successful." :
        print 'Connect to %s was successful' % db
    else :
        print 'Error: %s is not available' % db
```

- Ausgabe:

Connect to activities was successful
Connect to blogs was successful
Connect to communities was successful
[...]

>> Change DataSource Parameters

- Setzt die Parameter minConnections und maxConnections für die Connections DataSources
  - Werte wie im Performance Tuning Guide Connections 4.0 erwähnt als Startparameter
- Dictionary mit Datenbankname und empfohlenen Parametern:

```
perf = {'activities':{'minConnections':1,'maxConnections':50},
        'blogs':{'minConnections':1,'maxConnections':250},
        'communities':{'minConnections':10,'maxConnections':200},
        'dogear':{'minConnections':1,'maxConnections':150},
        'files':{'minConnections':10,'maxConnections':100},
        'forum':{'minConnections':50,'maxConnections':100},
        'homepage':{'minConnections':20,'maxConnections':100},
        ...
        'wikis':{'minConnections':1,'maxConnections':100}}
```

- statementCacheSize=100 (für DB2) bzw. 50 (für Oracle)

>> cfgDataSource.py

```
perf = {'activities':{'minConnections':1,'maxConnections':50},
        'blogs':{'minConnections':1,'maxConnections':250},
        'communities':{'minConnections':10,'maxConnections':200},
        'dogear':{'minConnections':1,'maxConnections':150},
        'files':{'minConnections':10,'maxConnections':100},
        'forum':{'minConnections':50,'maxConnections':100},
        'homepage':{'minConnections':20,'maxConnections':100},
        'metrics':{'minConnections':1,'maxConnections':75},
        'mobile':{'minConnections':1,'maxConnections':100},
        'news':{'minConnections':50,'maxConnections':75},
        'profiles':{'minConnections':1,'maxConnections':100},
        'search':{'minConnections':50,'maxConnections':75},
        'wikis':{'minConnections':1,'maxConnections':100}}

statementCacheSize = 100 #change to 50 for oracle

for db in perf.keys(): # Looping through databases
    print 'Change DataSource parameters for: %s' % db.upper()

    t1=AdminConfig.getid('/DataSource:' + db + '/')
    print '  statementCacheSize: ' + str(statementCacheSize)
    print '  minConnections: ' + str(perf[db]['minConnections'])
    print '  maxConnections: ' + str(perf[db]['maxConnections'])
    AdminConfig.modify(t1,'[[statementCacheSize "' + str(statementCacheSize) + '"]]')
    AdminConfig.modify(t1,'[[connectionPool [[minConnections "' + str(perf[db]['minConnections']) + '"]
[maxConnections "' + str(perf[db]['maxConnections']) + '"]]]]')
    AdminConfig.save()
```

>> MemberService für einen Benutzer in allen Apps aufrufen

- memberSyncByEmail.py
- ./wsadmin.sh –lang jython –f "memberSyncByEmail.py" cstoettner@stoeps.local
  - Mailadresse als Parameter an wsadmin übergeben

```
MAILADDRESS = sys.argv[0]
print "Syncing MemberService for " + MAILADDRESS
execfile("/opt/install/scripts/loadAll.py")
ActivitiesMemberService.syncMemberExtIdByEmail(MAILADDRESS)
BlogsMemberService.syncMemberExtIdByEmail(MAILADDRESS)
CommunitiesMemberService.syncMemberExtIdByEmail(MAILADDRESS)
DogearMemberService.syncMemberExtIdByEmail(MAILADDRESS)
FilesMemberService.syncMemberExtIdByEmail(MAILADDRESS)
ForumsMemberService.syncMemberExtIdByEmail(MAILADDRESS)
NewsMemberService.syncMemberExtIdByEmail(MAILADDRESS)
WikisMemberService.syncMemberExtIdByEmail(MAILADDRESS)
```

- besser:

```
try:
   print "Sync Dogear"
   DogearMemberService.syncMemberExtIdByEmail(MAILADDRESS)
except:
   print 'No user with Email ' + MAILADDRESS +' found'
```

>> Alle Benutzer in allen Apps gegen LDAP synchronisieren

- ./wsadmin.sh –f memberSyncAllByEXID.py "true|false"
  - Beispiel: `ActivitiesMemberService.syncAllMembersByExtId( {"updateOnEmailLoginMatch": true|false} )`

```
EmailMatch = sys.argv[0]
# Loading Connections Administration Commands
execfile("loadAll.py")

apps = ['Activities','Blogs','Communities','Dogear','Files','Forums','News','Wikis']

def memService(appname,EmailMatch):
    if("Activities" == appname) :
        print "\tActivitiesMemberService.syncAllMembersByExtId"
        ActivitiesMemberService.syncAllMembersByExtId( {"updateOnEmailLoginMatch": EmailMatch } )
    elif("Blogs" == appname) :
        print "\tBlogsMemberService.syncAllMembersByExtId"
        BlogsMemberService.syncAllMembersByExtId( {"updateOnEmailLoginMatch": EmailMatch } )
    ...
    else :
        print "\tUnknown application name '" + appname + "'"

for app in apps:
    print "Sync all Members by EXTID for " + app
    memService(app, EmailMatch)
```

>> Backup Security/J2EE Rollen

- Skript erstellt jeweils ein Textfile mit den Security Rollen aller installierten Applications
  - `./wsadmin.sh -lang jython -f "{path}/securityrolebackup.py" 'backuppath'`

```
apps = AdminApp.list()
appsList = apps.split(lineSeparator) #List of all applications
path = '/opt/install/backup' # must exist!
for app in appsList:
    filename = app + ".txt"
    my_file = open(path + '/' + filename,'w')
    my_file.write (AdminApp.view(app,"-MapRolesToUsers"))
    my_file.flush
    my_file.close()
```

>> Restore Security/J2EE Rollen

- Skript liest die Rollen aus den Backup Textdateien
- Praktisch nach Updates, CR/Fixpack Installationen
- Dateien werden in Dictionaries zerlegt und über die Funktion setSecurityRoles gesetzt

```
def setSecurityRoles(dictionary,appName):
    strRoleChange = '['
    for role in dictionary.keys():
        # Loop through Roles
        strRoleChange += '[\"' + role + '\" '
        strRoleChange += dictionary[role]['Everyone?'] + ' '
        strRoleChange += dictionary[role]['All authenticated?'] + ' '
        strRoleChange += '\"' + dictionary[role]['Mapped users'] + '\" '
        strRoleChange += '\"' + dictionary[role]['Mapped groups'] + '\"] '
    strRoleChange += ']]'
    AdminApp.edit(appName, '[-MapRolesToUsers' + strRoleChange +']')
    print "Setting Roles and Users for %s" % appName
    AdminConfig.save()
```

>> Alternative für J2EE Rollen

- Skript von http://kbild.ch (Blog von Klaus Bild)
- Set Default Security Roles and Restrict Roles to Administrators
- Praktisch für konsistente Rollen bzw. initiales Einrichten der Rollen
- Orginalskript erweitert um Gruppen:

```
connwasadmin = 'wasadmin'
connadmin = 'Admin1|Admin2'
connmoderators = 'Moderator1|Moderator2'
connmetrics = 'Metrics1|Metrics2'
connmobile = 'Mobile1|Mobile2'
```

>> Initiales setzen der J2EE Rollen

```
# Variables for Groupmapping
connadmingroup = 'CNXAdmins'
connmoderatorgroup = 'CNXModerators'
connmetricsgroup = 'CNXMetricsAdmins'
connmobilegroup = 'CNXMobileAdmins'

appName = 'Activities'
# "role" Yes No = everyone
# "role" No Yes = All Authenticated
# "role" No No  = None
AdminApp.edit( appName, '[-MapRolesToUsers [["person" No Yes "" ""]
  ["everyone" Yes No "" ""] ["reader" Yes No "" ""]
  ["metrics-reader" No Yes "" ""]
  ["search-admin" No No "' + connwasadmin + '|' + connadmin + '" "' + connadmingroup + '"]
  ["widget-admin" No No "' + connwasadmin + '|' + connadmin + '" "' + connadmingroup + '"]
  ["admin" No No "' + connwasadmin + '|' + connadmin + '" "' + connadmingroup + '"]
  ["bss-provisioning-admin" No No "" ""] ]]' )
print "Setting Roles and Users for Activities"
AdminConfig.save()
...
```

IBM DB2 / SQL

>> DB2 praktische Kommandos

- Liste aller Datenbanken einer Instanz
  - Linux
    - `db2 list database directory | grep alias | awk '{print $4}' | sort`
  - Windows (Powershell)
    - `db2cmd -i -c -w "db2 list database directory | where {$_ -match "alias"} | %{ $_.Split('=')[1]; }"`
- Liste aktiver Datenbanken anzeigen
  - `db2 list active databases`

>> Automatische Maintenance

- DB2 9.7 Export der Policy Files und anschließender Import in andere DB
- Skripting DB2 Automatic Maintenance | Stoeps
- IBM Data Studio
  - Maintenance, Statistiken, Backup, Zeitfenster für eine DB konfigurieren
  - Kommando als SQL Skript speichern

Edit Run Save...

```
CONNECT TO "homepage";
UPDATE DATABASE CONFIGURATION USING auto_db_backup ON auto_reorg ON auto_runstats ON auto_prof_upd ON auto_stats_prof ON;
CALL SYSPROC.AUTOMAINT_SET_POLICY ('MAINTENANCE_WINDOW', BLOB('<?xml version="1.0" encoding="UTF-8"?><DB2MaintenanceWindows xmlns="http:
CALL SYSPROC.AUTOMAINT_SET_POLICY ( 'AUTO_BACKUP', BLOB('<?xml version="1.0" encoding="UTF-8"?><DB2AutoBackupPolicy xmlns="http://www.ibm.co
CALL SYSPROC.AUTOMAINT_SET_POLICY ( 'AUTO_REORG', BLOB('<?xml version="1.0" encoding="UTF-8"?><DB2AutoReorgPolicy xmlns="http://www.ibm.com/
CALL SYSPROC.AUTOMAINT_SET_POLICY ( 'AUTO_RUNSTATS', BLOB('<?xml version="1.0" encoding="UTF-8"?><DB2AutoRunstatsPolicy xmlns="http://www.ibr
CONNECT RESET;
```

>> Automatische Maintenance (2)

- automaint.sql
  - kopieren der update Zeile und der 4 CALL statements
- setmaintenance.sh

```
#!/bin/bash

databases=$(db2 list database directory | grep alias | awk '{print $4}' | sort)
for database in ${databases[@]}
do
 echo $database
 db2 "connect to $database"
 db2 -tvf automaint.sql
 db2 "connect reset"
done
```

>> ExId in Connections Apps prüfen

- Benutzer in peopleDB nachschlagen
- Suche in den restlichen Applikationen

```
MAIL=$1

db2 -x "connect to peopledb" | grep alias | awk '{print $5}'
db2 -x "SELECT PROF_GUID, PROF_MAIL FROM EMPINST.EMPLOYEE WHERE PROF_MAIL_LOWER = '${MAIL,,}'"
db2 -x "connect reset" > /dev/null
while true; do
```

```
[db2inst1@cnxdb2 ~]$ ./checkExID.sh cstoettner@stoeps.local
PEOPLEDB
84A543FE-A27D-395A-C125-7B8F00665563
                                                                    CStoettner@stoeps.local
                                                                                                  MAIL'"
Which email address should be used for Lookup?
CStoettner@stoeps.local
OPNACT                                                                                            _'"
84A543FE-A27D-395A-C125-7B8F00665563

BLOGS                                                                                              '${MAIL,,}'"
84A543FE-A27D-395A-C125-7B8F00665563

                                                                                                  '
SNCOMM
84A543FE-A27D-395A-C125-7B8F00665563
```

```
db2 -x "select DIRECTORY_ID from FILES.USER where email = '$MAIL'"
db2 -x "connect reset" > /dev/null
```

FRITZ&MACZIOL group

Und noch mehr

>> SSL Root Zertifikat abrufen

- Export Root Zertifikat von selfsigned Zertifikaten oft aufwändig
- Verwendung z.B.:
  - TDI und LDAPS
  - Domino und Embedded Experience Config
- Prerequist: openssl, java (keytool)
- Aufruf:
  - `./create_cacerts.sh -h hostname -p port -f path/filename`
- TDIPopulation/solution.properties:
  - javax.net.ssl.trustStore=/opt/install/keystore

>> create_cacerts.sh

```
TMP1=`mktemp -d`
trap "rm -rf $TMP1" EXIT

while getopts h:p:f:?: option
do
    case "${option}"
        in
        h) SERVERNAME=${OPTARG};;
        p) SERVERPORT=${OPTARG};;
        f) STORECACERTS=${OPTARG};;
        ?) echo "USAGE: `basename $0` -h hostname -p port -f Certfile\n"
        echo "You have to type a password for keyfile twice and set\n"
        echo "the key to trusted!"
        exit
        ;;
    esac
done

if [ -z "$SERVERNAME" ] | [ -z "$SERVERPORT" ] | [ -z "$STORECACERTS" ] ; then
    echo "USAGE: `basename $0` -h hostname -p port -f Certfile"
    echo "You have to type a password for keyfile twice and set\n"
    echo "the key to trusted!"
    exit
fi
```

```
openssl s_client -showcerts -connect $SERVERNAME:$SERVERPORT < /dev/null > $TMP1/cst-key.out
openssl x509 -outform DER < $TMP1/cst-key.out > $TMP1/cst-key.der
openssl x509 -inform der -in $TMP1/cst-key.der -out $TMP1/cst-key.pem
keytool -import -alias Selfsigned -keystore $STORECACERTS -file $TMP1/cst-key.pem
```

>> TDI und Javascript

- Feldwerte bei der Synchronisation mit LDAP zusammenführen
- Felder auf null setzen
- Zeitzone vorbelegen
- Funktionen können im TDISOL-Verzeichnis in die profiles_functions.js angehängt werden
- Verwendung in der map_dbrepos_from_source.properties
  - z.B. displayname={functionsname}

>> Fullname / Displayname im LDAP nicht vorhanden

- AD:
  - nur Vor- und Nachname gepflegt
  - Nachname, Vorname usw.

```
function func_compute_CN(fieldname) {
    var givenName = work.getAttribute("givenName");
    var sn = work.getAttribute("sn");

    if(sn != null) {
        var result = givenName + ' ' + sn;
    }
    return result;
}
```

- Verwendung in map_dbrepos_from_source.properties
  - displayName={func_compute_CN}

>> Zeitzone setzen

- Zeitzone im LDAP oft nicht vorhanden oder nicht gepflegt
- User setzen dann beim editieren des Profils die Zeitzone auf -12
- profiles_functions.js:

```
function function_settimezone(fieldname){
    var timeZone = 'Europe/Amsterdam';
    result = timeZone;
    return result;
}
```

- Verwendung in map_dbrepos_from_source.properties
  - timezone={function_settimezone}

>> Felder in peopleDB auf null setzen

- Felder werden manchmal falsch gemappt
- Zurücksetzen über Skript in profiles_functions.js

```
function function_setnull(fieldname){
    var result = '';
    return result;
}
```

- Verwendung in map_dbrepos_from_source.properties
  - `mobile={function_setnull}`
- Achtung! Alle Daten in Beschreibung werden überschrieben

Ressourcen

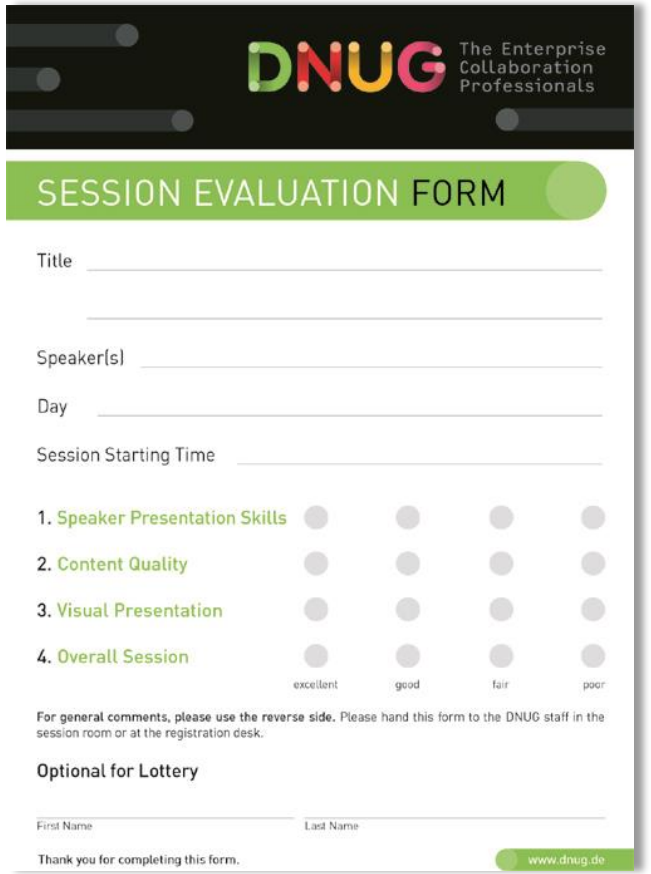
>> Download der gezeigten Skripte

- Sämtliche Skripte können OHNE GARANTIE und auf eigenes Risiko von folgendem GIT Repository heruntergeladen werden:
  https://github.com/stoeps13/ibmcnxscripting
- Da gibt es mehr als hier gezeigt wurde:
  - Benutzer inaktiv und aktiv setzen
  - druckbare Version der Connections Doku erstellen
  - DB2 Backup, Lastlogon
  - DB2 init Skript für CentOS / Red Hat
  - Change Monitoring Policy
  - usw.

>> Ausblick

- Powershell und Windows Batches
- Skripte für Troubleshooting
- Dokumentation
- Sie sind herzlich eingeladen die Skripte zu benutzen und Änderungen hochzuladen
- Diskussionen über
  - Skype: christophstoettner
  - Twitter: @stoeps
  - G+, Facebook, LinkedIn ...

Danke fürs Zuhören



DNUG The Enterprise Collaboration Professionals

SESSION EVALUATION FORM

Title

Speaker(s)

Day

Session Starting Time

	excellent	good	fair	poor
1. Speaker Presentation Skills				
2. Content Quality				
3. Visual Presentation				
4. Overall Session				

For general comments, please use the reverse side. Please hand this form to the DNUG staff in the session room or at the registration desk.

Optional for Lottery

First Name Last Name

Thank you for completing this form. www.dnug.de

Bitte nehmen Sie sich die Zeit, um diesen Vortrag zu bewerten

(A6-Block in Ihren Konferenzunterlagen)

Rückgabe

Geben Sie das ausgefüllte Bewertungsblatt bei dem Moderator/Betreuer Ihres Vortrages bzw. am Tagungscounter der DNUG ab.

Verlosung

Unter allen Teilnehmern wird ein ***iPod nano*** verlost.