WDBear-Manager Tutorial

Version: 1.2.0-W1.12.0

lousy.kizura@gmail.com

Please visit
http://www.wdbmanager.de.gg

Please:
If you like this program and find it usefull, donate money to a charity organization of your choice.
I recommend any organization that fights against cancer.

Bitte:
Wenn Dir das Programm gefällt und Du es regelmäßig verwendest, dann spende Geld für eine wohltätige Organisation Deiner Wahl.
Ich empfehle solche, die gegen den Krebs kämpfen.

I don't want no money. I did this to give the community a little back, what it gave to me.
English

WDBManager is designed to handle the information found inside the WDB files stored inside the client's cache.
WDB files are transferred during playing on a WoW server and are stored inside the WDB folder. Using this caching strategy helps blizzard to reduce the network traffic.

The WoW Client contains the whole world, but no quests, npcs or mobs. These information is transferred from the server to the client. And the client caches these information.

With WDBearManager you can:

- Create a CSV file from your WDB files
- Create a TXT file from your WDB files
- Store the WDB information in a SQL database (tested with Hypersonic, Oracle und MySQL4.x)
- Patch existing *.scp files with the information from your database
- Start the program using the console, gui or scripting version
- Enhance the program using Jython scripts
  (You have full access to the WDB database using Jython!)
- Store **any** WDB files you get your hands on. WDBearManager supports all formats from 1.2.4 to 1.11.1
- Supports localized WDB data.
  Read enUS, enGB, deDE; whatever WDB files you get you hands on. They can all be stored in one database (inf_locale, inf_version – locale information and what version the WDB was)

Kizura Zgabi
lousy.kizura@gmail.com

Installation

You need

Linux or Win32 environment (Others may work, too, but are not tested)
Java 1.4.2 installed

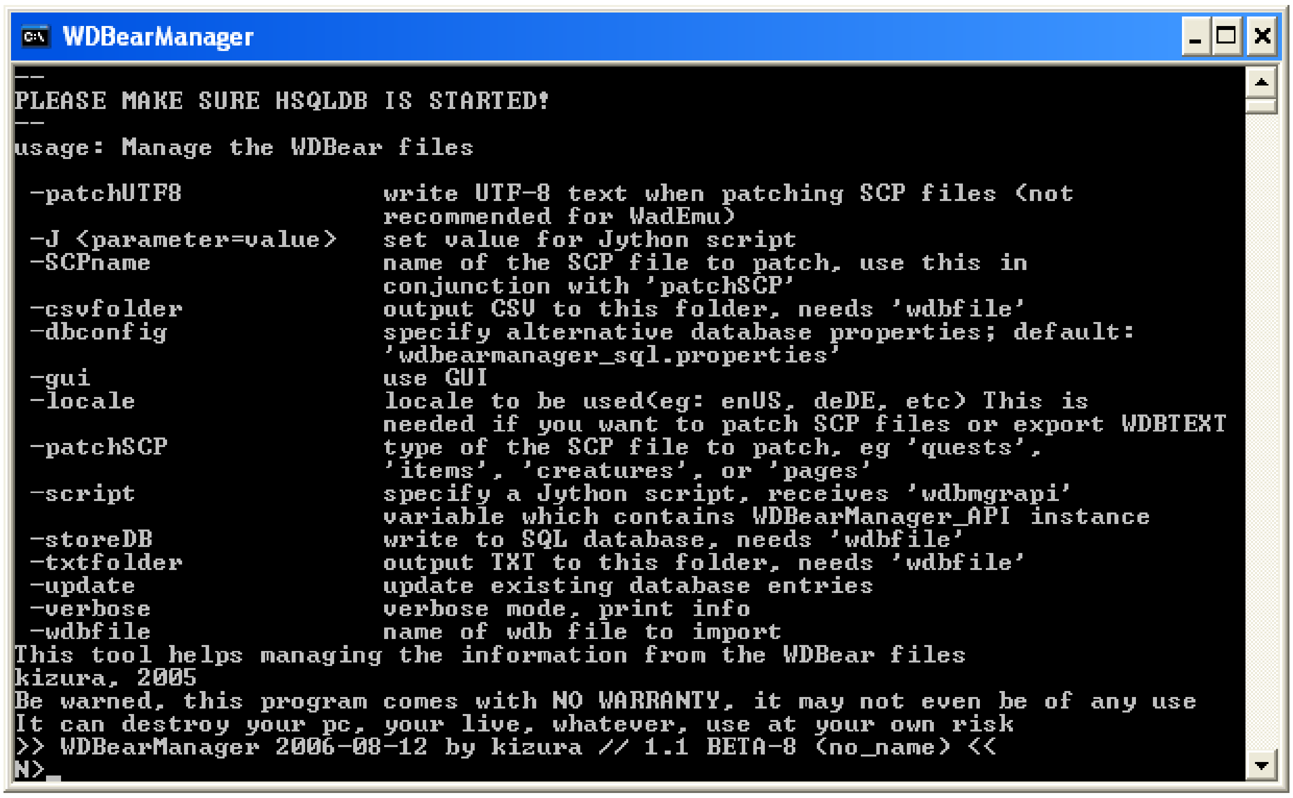
Steps

(Win32)
Unzip the archive
Open msdos prompt
Change to the installation folder

Invoke the program:

```
java -jar wdbearmanager.jar
```

Output:



WDBearManager

```
--
PLEASE MAKE SURE HSQLDB IS STARTED!
--
usage: Manage the WDBear files

 -patchUTF8                write UTF-8 text when patching SCP files (not
                           recommended for WadEmu)
 -J <parameter=value>      set value for Jython script
 -SCPname                  name of the SCP file to patch, use this in
                           conjunction with 'patchSCP'
 -csvfolder                output CSV to this folder, needs 'wdbfile'
 -dbconfig                 specify alternative database properties; default:
                           'wdbearmanager_sql.properties'
 -gui                      use GUI
 -locale                   locale to be used(eg: enUS, deDE, etc) This is
                           needed if you want to patch SCP files or export WDBTEXT
 -patchSCP                 type of the SCP file to patch, eg 'quests',
                           'items', 'creatures', or 'pages'
 -script                   specify a Jython script, receives 'wdbmgrapi'
                           variable which contains WDBearManager_API instance
 -storeDB                  write to SQL database, needs 'wdbfile'
 -txtfolder                output TXT to this folder, needs 'wdbfile'
 -update                   update existing database entries
 -verbose                  verbose mode, print info
 -wdbfile                  name of wdb file to import
This tool helps managing the information from the WDBear files
kizura, 2005
Be warned, this program comes with NO WARRANTY, it may not even be of any use
It can destroy your pc, your live, whatever, use at your own risk
>> WDBearManager 2006-08-12 by kizura // 1.1 BETA-8 (no_name) <<
N>_
```

Usage

Parameter

- csvfolder <folder>
  Specifiy the folder for the CSV export.
  If you want to create a CSV file, you must provide the output directory.
  Needs "`wdbfile`" parameter.
- gui
  Start the graphical user interface, all other parameters are ignored.
- locale
  If you want to patch SCP files or export WDBTEXT, you should specify a locale.
  Remember that WDBEarManager supports multiple WDB formats and it also supports multiple languages. Common values are „enUS“, „enGB“, „deDE“, etc.
- J
  Define Parameters for a Jython script. Syntax: "-Jparam=value"
- script
  Call this Jython script. This script is invoked in the context of the WDBManager and receives the variable "`wdmgrapi`", which is an instance of "`WoWWDBManager_I`".
  (Please refer to JavaDocs)
- storeDB
  Save the values from the WDB file inside the database
  Needs "`wdbfile`" parameter.
- txtfolder <ordner>
  Specify the folder for TXT export..
  If you want to create a TXT file, you must provide the output directory.. The TXT file can be used for debugging.
  Needs "`wdbfile`" parameter.
- update
  Use "`update`" if you also want to update entries inside the database. Otherwise existing entries are ignored (insert only).
  Needs "`wdbfile`" parameter.
- verbose
  Switch to verbose mode, displays more info (console).
- wdbfile
  Specify the complete path to the WDB file.
- SCPName <myFile.scp>
  Name of the SCP file that should be patched.
  The information stored inside the database are used to replace the entries inside the SCP file.
  Creates:
  "<myFile.scp>_patch“ Contains the patched SCP file
  "<myFile.scp>_patch_info" Contains information about the patch process

Examples

Create CSV file from a WDB file

java -jar wdbearmanager.jar -wdbfile c:\temp\wdb\questcache.wdb -csvfolder c:\meine_csvs

Create TXT file from a WDB file

java -jar wdbearmanager.jar -wdbfile c:\temp\wdb\questcache.wdb -txtfolder c:\meine_txts

Create a CSV and a TXT file from a WDB file

java -jar wdbearmanager.jar -wdbfile c:\temp\wdb\questcache.wdb -csvfolder c:\meine_csvs -txtfolder c:\meine_txts

Store contents of a WDB file inside a database (incl. update)

java -jar wdbearmanager.jar -wdbfile c:\temp\wdb\questcache.wdb -storeDB -update

Create a patched *.scp file

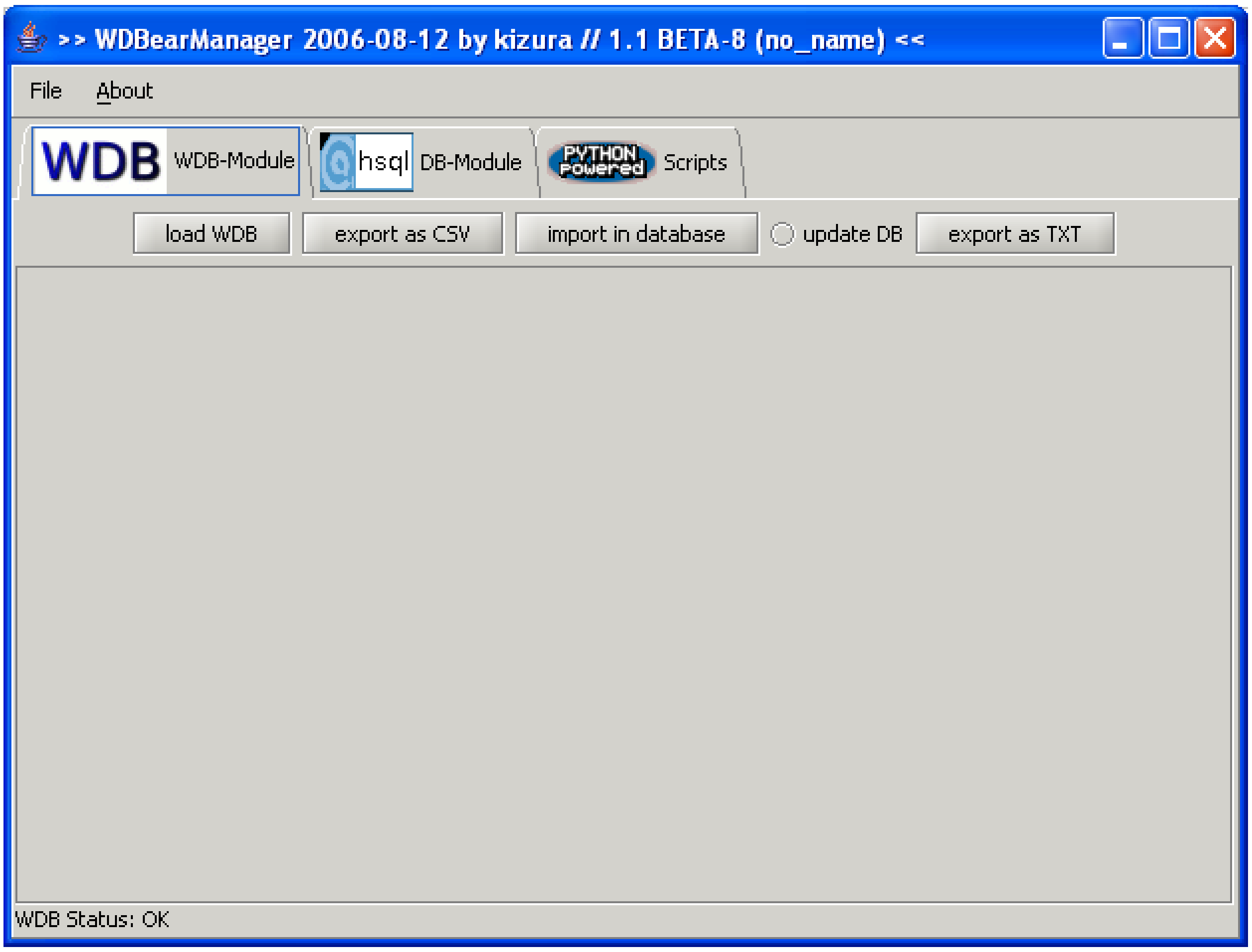
Replaces values inside an existing quests.scp, items.scp, etc. with values from your database. What should be replaced is configured inside "patchSCP.xml" and can be changed by the user. Please note that patching "multi value" attributes does not work (eg level=1 4)

Values for "`patchSCP`": quests, items, creatures, pages

```
java -jar wdbearmanager.jar -locale enUS -patchSCP quests -SCPname
/tmp/myQuests.scp
```

Starting the graphical user interface (gui)

```
java -jar wdbearmanager.jar -gui
```



Calling a Jython script

```
java -jar wdbearmanager.jar -script c:\temp\myScript.jy
-Jnose=bear
```

Call the script "`c:\temp\myScript.jy`" and pass the parameter "`nose`" with the value "`bear`". Now your Jython script has a variable called "`nose`" with the value "`bear`".

Database configuration

WDBManager supports 3 types of databases:

- Oracle (9i tested)
- MySQL
- Hypersonic SQL Datenbank 1.7.3 (included)

`wdbmanager_sql.properties`

```
#
# Config file for the SQL storeage module
# (Class implements I_StoreWDB)
#
#
# The ManagePP_SQL module stores/retrieves the data
# from a SQL database.
#
# jdbcurl    - JDBC URL to access the database
# jdbcdriver - Use this class to access the database
# username   - username to access the database
# password   - password to access the database
# dbprefix   - all created tables use this prefix (eg: myprefix_table)
#
# Important:
# The database user MUST have the right to create tables!
#
# kizura zgabi, march 2005

jdbcurl=jdbc:hsqldb:hsql://localhost
jdbcdriver=org.hsqldb.jdbcDriver
username=sa
password=wdbpasswd
dbprefix=wowwdb_

# mySQL
#jdbcurl=jdbc:mysql://localhost/meinedatenbank
#jdbcdriver=com.mysql.jdbc.Driver
#username=mysqlUser
#password=mysqlPass

#oracle
#jdbcurl=jdbc:oracle:thin:@mydbhost.somewhere.org:1521:SID
#jdbcdriver=oracle.jdbc.driver.OracleDriver
#username=oraUser
#password=oraPass
```

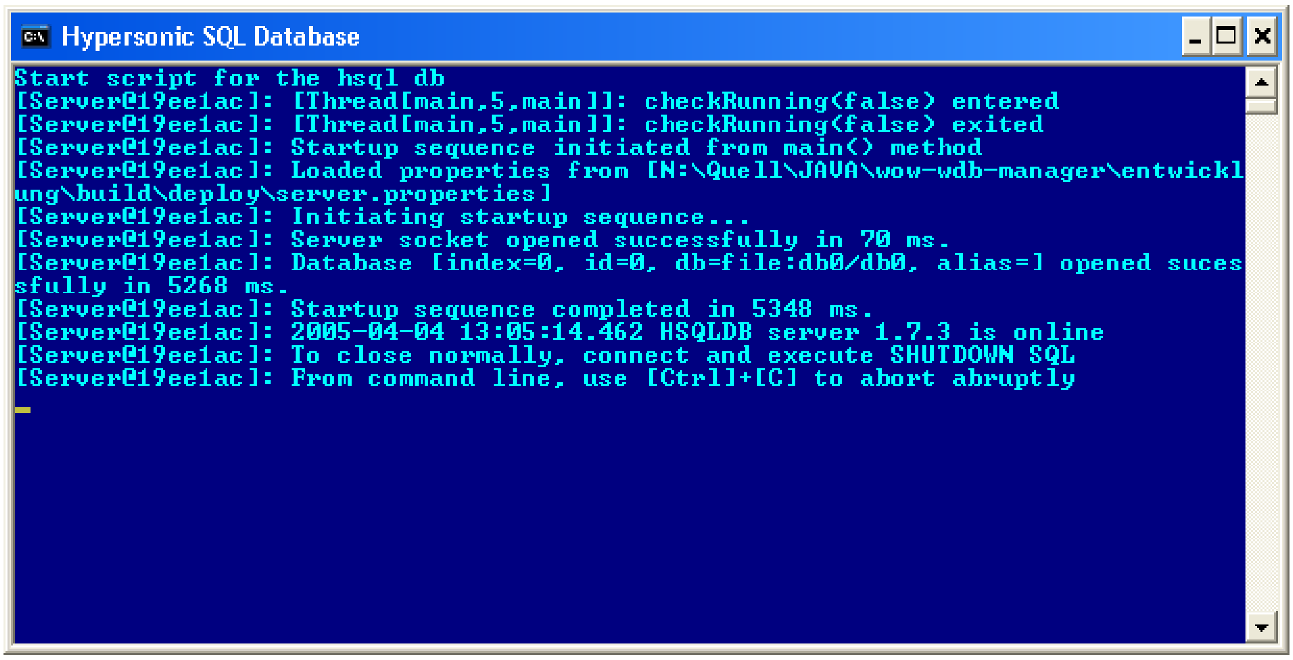
Using Hypersonic SQL database (included)

Pre:

Suffix .bat – Script for Win32
Suffix .sh – Script for Linux

You find the script „`start_hsqldb`".in your installation directory.
This script will start the hypersonic SQL database.
(You can access this database with JDBC, username and password can be found in the file
`wdbmanager_sql.properties` zu entnehmen.

If you want to access other databases, please adjust the script „`wdbmanager_sql.properties`".



```
Start script for the hsql db
[Server@19ee1ac]: [Thread[main,5,main]]: checkRunning(false) entered
[Server@19ee1ac]: [Thread[main,5,main]]: checkRunning(false) exited
[Server@19ee1ac]: Startup sequence initiated from main() method
[Server@19ee1ac]: Loaded properties from [N:\Quell\JAVA\wow-wdb-manager\entwickl
ung\build\deploy\server.properties]
[Server@19ee1ac]: Initiating startup sequence...
[Server@19ee1ac]: Server socket opened successfully in 70 ms.
[Server@19ee1ac]: Database [index=0, id=0, db=file:db0/db0, alias=] opened suces
sfully in 5268 ms.
[Server@19ee1ac]: Startup sequence completed in 5348 ms.
[Server@19ee1ac]: 2005-04-04 13:05:14.462 HSQLDB server 1.7.3 is online
[Server@19ee1ac]: To close normally, connect and execute SHUTDOWN SQL
[Server@19ee1ac]: From command line, use [Ctrl]+[C] to abort abruptly
```

Starting the hypersonic SQL database

Config-files

Description of the WDB files

- creaturecache.xml
- gameobjectcache.xml
- itemcache.xml
- itemnamecache.xml
- itemtextcaxhe.xml
- npccache.xml
- pagetextcache.xml
- questcache.xml

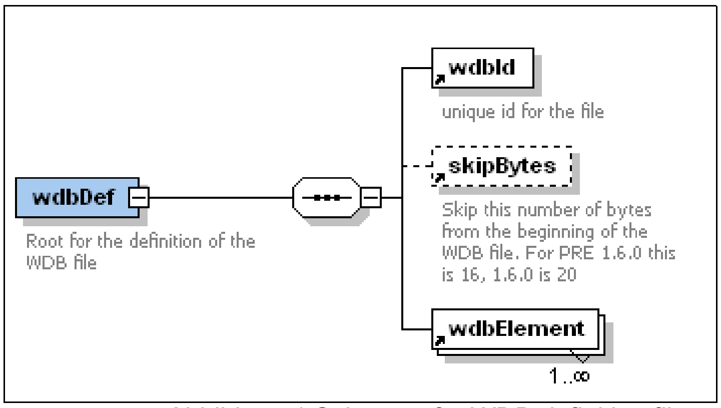

wdbDef
Root for the definition of the WDB file
wdbId
unique id for the file
skipBytes
Skip this number of bytes from the beginning of the WDB file. For PRE 1.6.0 this is 16, 1.6.0 is 20
wdbElement
1..∞

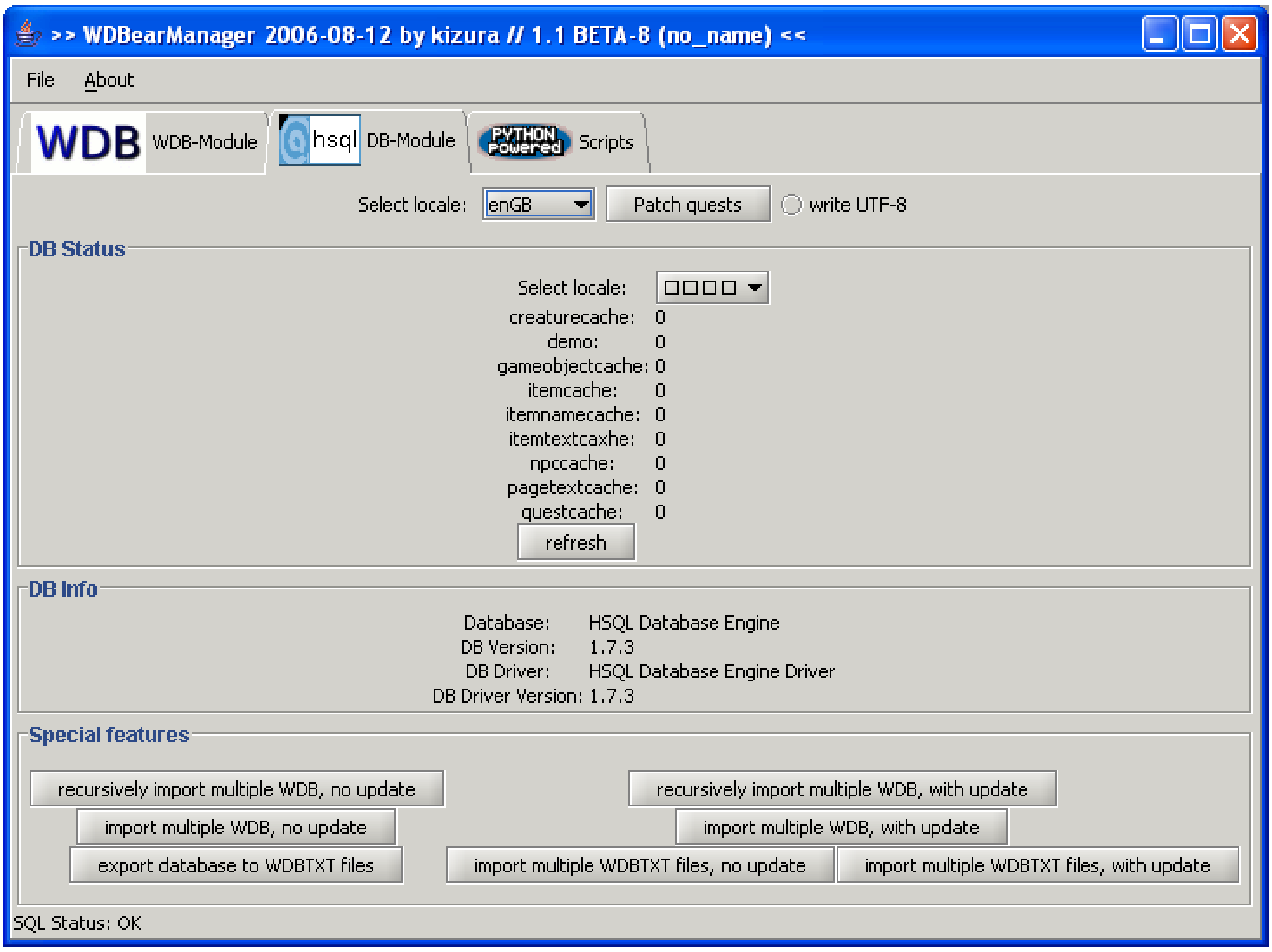
Abbildung 1 Schema of a WDB definition file

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Config file, that describes the format of a WDB file.<br/>

  Version: 1.11
  Same as 1.10

  A lot of the description concerning the columns was taken from:
  http://www.sourcepeek.com/wiki/Main_Page

  Changes to this file will affect the application!
  !Beware!

  possible types - internal mapping        - comment
  -
  tinyInt        - java.sql.Types.TinyInt    1 byte signed int
  smallInt       - java.sql.Types.SmallInt   Short (2 byte signed int)
  integer        - java.sql.Types.Int        Integer (4 byte signed int)
  varChar        - java.sql.Types.VARCHAR    string, ends with NULL
  bigIntHex      - java.sql.Types.NUMERIC    4 byte int value, gets hex encoded
                                             after read from WDB (Stored as
                                             HEX inside database)
  single         - java.sql.Types.FLOAT      4 byte float, single precision
                                             Stored inside database as
                                             Varchar()
                                             to preserve precision.
  char, size     - java.sql.Types.CHAR       string, fixed width
                   eg:   <wdbElement name="wdb_Text" type="char" size="20"/>
                   BEWARE:
                   Internal encoding of WDB is UTF-8 *but* you can only specify
                   the number of bytes here. This can significantly differ
                   (eg german "ß" is 2 bytes)
  -

  For PRE-1.6.0, the WDB files contain an ID of 16 bytes
  1.6.0 - ID is 20 bytes

  Please use "skipBytes" to set the length of the ID
  (Thanks to andrikos for this hint.)

  kizura zgabi, >= 03/2005
-->
<wdbDef>
  <wdbId name="pagetextcache" />
  <skipBytes numBytes="20" />
  <wdbElement name="wdb_ID" type="integer" />
  <wdbElement name="wdb_RecLen" type="integer" />
  <wdbElement name="wdb_Text" type="varChar" />
  <wdbElement name="wdb_nextPageId" type="integer" />
</wdbDef>
```

Patching SCP files – PatchSCP.xml

Maybe you want to patch some existing SCP files with the information you gathered from the WDB files.

All patches you defined are displayed in the „HSQL tab“.


>> WDBearManager 2006-08-12 by kizura // 1.1 BETA-8 (no_name) <<
File
About
WDB WDB-Module
hsql DB-Module
PYTHON Powered Scripts
Patch quests
refresh
recursively import multiple WDB, no update
recursively import multiple WDB, with update
import multiple WDB, no update
import multiple WDB, with update
export database to WDBTXT files
import multiple WDBTXT files, no update
import multiple WDBTXT files, with update

Select locale: enGB
write UTF-8

DB Status

```
Select locale:      □□□□
creaturecache:      0
demo:               0
gameobjectcache:    0
itemcache:          0
itemnamecache:      0
itemtextcaxhe:      0
npccache:           0
pagetextcache:      0
questcache:         0
```

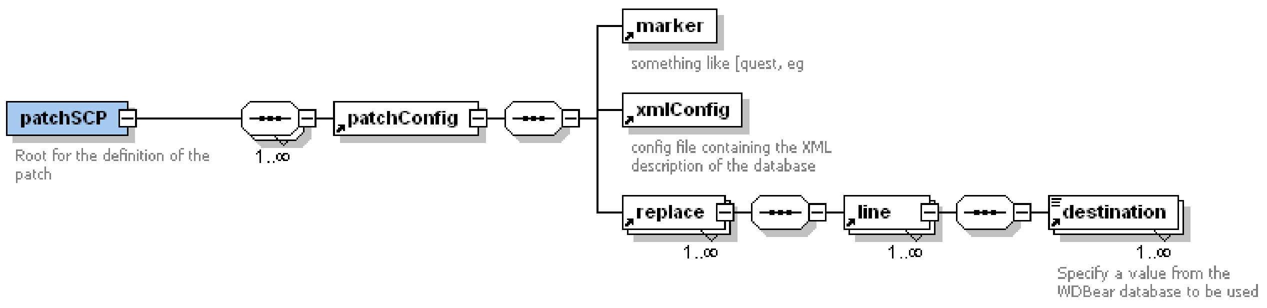
DB Info

```
Database:           HSQL Database Engine
DB Version:         1.7.3
DB Driver:          HSQL Database Engine Driver
DB Driver Version:  1.7.3
```

Special features

SQL Status: OK

This can easily be done using the WDBearManager and the file „PatchSCP.xml“.


patchSCP
Root for the definition of the patch
1..∞
patchConfig
marker
something like [quest, eg
xmlConfig
config file containing the XML description of the database
replace
1..∞
line
1..∞
destination
1..∞
Specify a value from the WDBear database to be used

While reading this, please refer to patchSCP.xml, which serves as an example concerning patching SCP files.

patchSCP
Root node

patchConfig

You write a patchConfig entry for any SCP file you want to patch.
Attribute:

- name
  Set the name of the SCP file.

marker
Attribute:

- name
  Name to be displayed

xmlConfig

Attribute:
- name
  Name of the XML config file to be used for patching.

replace

Attribute:
- name
  Name of the key inside the SCP file that should be replaced

line

Tag that describes the start/end of a single line to be patched. A line consists of one or more destinations. A destination is used to set the values inside a line.

destination

Attribute:
- name
  Name of the column inside the WDBear database. The value inside the database is used for patching. You can specify one or more destination values.
- notNull
  Ignore the line, if the value inside the WDBear database is null (numeric ZERO). This is very handy if you for example want to replace item_choice, item_reward or anything like this.
  These keys can occure more than one time and they are only „valid“, if the first value is not null (not zero), since this defines the „amount“ and the second value defines the „object“.

Example for patching quests.scp

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
If you want to add your own patches, just edit this file.

It is used for the dynamic configuration of all patch algorithms.

syntax:
source = text in scp file
destination = name in wdbmanager database
line = how a line is to be built inside the SCP file
       (Here you specify one or more entry that constructs a line inside the SCP file)

convertTo = manipulate data read from the database
Valid settings are:
bin, hex, octal - convert value to binary, hex or octal
            (Warning: Make sure your input value is an integer, otherwise the patch process
            will stop with an error!)
upper, lower - convert String to upper/lower case
int - convert value to an integer


If you want a text across mulitple lines, use
multiline="1"

(As used in pages.scp)

eg
this will do:

text=blah blah
text=blah2 blah2

etc.

Hint:
- A line is only executed, if the first entry inside the database is not 0
  (Numeric value for zero).
- First specify all "replace" settings, THEN specify all "replaceMulti" settings

replace -> reward_item example (quests.scp)

    <replace source="desc">
      <line>
        <destination name="wdb_Desc1"/>
      </line>
    </replace>
    <replace source="reward_item">
      <line>
        <destination name="wdb_RewardItem1a" notNull="1"/>
        <destination name="wdb_RewardItem1b"/>
      </line>
      <line>
        <destination name="wdb_RewardItem2a" notNull="1"/>
        <destination name="wdb_RewardItem2b" notNull="1"/>
      </line>
      <line>
              <destination name="wdb_Unknown11" notNull="1"/>
              <destination name="wdb_Unknown12" notNull="1">
                          </destination>
      </line>
      <line>
        <destination name="wdb_Unknown13" notNull="1"/>
        <destination name="wdb_Unknown14" notNull="1"/>
      </line>
    </replace>

If you want to manipulate the value from the WDBear database before
it is written to the SCP, use the <destination> TAG.
Just write your Jython code between <destination> and
</destination>.

eg: add "Hurz" to the "name"

    <replace source="name">
      <line>
```

```
        <destination name="wdb_Name"><![CDATA[
print wdb_Name
retValue = wdb_Name + "Hurz"
]]></destination>

You just provide a Jython script, which is executed each time.
The Jython script receives the name of the WDBear column as a
variable ("wdb_Name", as specified in <destination>)
To manipulate the value, set the variable "retValue".
This will replace the original value from the database.


kizura zgabi, 2005/2006

History:
1.1 - Aug 2006
            Added convertTo and destination2... destination10
            Thanks to Hochelf for the inspiration

version   $Rev: 170 $

-->
<patchSCP>
  <!-- config name is displayed in GUI with "patch XYZ
       It is also used as parameter for the "patchSCP" command line
       argument.
       eg
       java -jar obf_wowwdbmanager -patchSCP quests -SCPname myQuests.scp
   Hint: Do NOT change the "name" attribute!
   -->
  <patchConfig name="quests">
    <marker name="quest" />
    <xmlConfig name="questcache.xml" />
    <replace source="name">
      <line>
        <destination name="wdb_Name"><![CDATA[
print wdb_Name
retValue = wdb_Name
]]></destination>
            </line>
    </replace>
    <replace source="desc">
      <line>
        <destination name="wdb_Desc1"/>
      </line>
    </replace>
    <replace source="second_desc">
      <line>
        <destination name="wdb_Desc2"/>
      </line>
    </replace>
    <replace source="details">
      <line>
        <destination name="wdb_Desc3"/>
      </line>
    </replace>
    <replace source="reward_item">
      <line>
        <destination name="wdb_RewardItem1a" notNull="1"/>
        <destination name="wdb_RewardItem1b"/>
      </line>
      <line>
        <destination name="wdb_RewardItem2a" notNull="1"/>
        <destination name="wdb_RewardItem2b" notNull="1"/>
      </line>
      <line>
              <destination name="wdb_Unknown11" notNull="1"/>
              <destination name="wdb_Unknown12" notNull="1">
                          </destination>
      </line>
      <line>
        <destination name="wdb_Unknown13" notNull="1"/>
        <destination name="wdb_Unknown14" notNull="1"/>
      </line>
    </replace>
  </patchConfig>
</patchSCP>
```

Appendix

1.1 (2006-Aug-14)

- Added new methods to the API
  getVersion – Returns the version of the application
-
- More support for inf_locale added.
- Completeley reworked "PatchSCP.xml"
  Now the system supports inf_locale and more complex situations (Jython support added).
- Reworked database update scripting
  Please refer to "db-update"

1.0 (2006-May-07)

- Program now supports **any** WDB format
  Just edit "wdb_compatibility.properties" and add a folder to "wdbear-config". Doing this the application can determine the version of the WDB file and load the appropriate XML config.
- Added support for locale
  Now the program also supports multiple language files. Added "inf_locale" and "inf_version" to the database tables.
  A primary key for an item inside the database is defined by "inf_locale" and "wdb_ID".
- Program supports **all** WDB versions till 1.11

PRE-8 (2004-Apr-04)

- Enhanced Scripting support
- Implemented Plugins
  (See folder "`plugins`" for details, thanks to Yodan for pushing this forward)
- Fixed bug in XML concerning gameobjectcache
  (Thanks to Hochelf from WDDG-GER for finding the bug)
- Enhanced WBDManager_I (API)

Jython Scripts

Display number of quests in database

```
# Jython example to access WDBManager
# kizura, March 2005
# Needs at least WDBManager PRE4
#
# All rights reserved.
#
# This script is used for WDBManager
#
# It reads the entries from table "questcache" and
# prints out the quest_id and the quest's name
# -> Please check "questcache.xml" for all attributes! <-
#
# Every script gets "wdbmgrapi" variable
# wdbmgrapi is of type "WoWWDBManager_API"
# Please refer to Java Docs for more information
#
#
from java.util import *

# Connect to database
print 'Number of entries in table "QuestCache"'
print wdbmgrapi.getCountOf('questcache')
objQuestcache = wdbmgrapi.getAllObjectsTable('questcache')
print "Count: %d" % objQuestcache.size()
itObjs = objQuestcache.iterator()
for i in itObjs:
  print "QuestID: %d, QuestNName: %s" %
(i.getColumnValue("wdb_QuestId"),i.getColumnValue("wdb_name"))
```

Query database and generate TXT file

```
# Jython example to access WoWWDBManager
# kizura, March 2005
# Needs at least WowWDBManager PRE7
#
# All rights reserved.
#
# This script is used for WoWWDBManager
#
# Allow the user to query the WDB database.
#
# Generates a TXT file with the found entries from the
# database.
#
# params (eg: tabName=questcache):
# tabName - Name of the database table
#           creaturecache
#           gameobjectcache
#           itemcache
#           itemtextcaxhe
#           npccache
#           pagetextcache
#           questcache
#
# colName - Use this column for searching
#
# colValue - Compare all columns with this value
#
# colOrder - Name of the column to sort the result (ASC)
#
#
# Every script gets "wdbmgrapi" variable
# wdbmgrapi is of type "WoWWDBManager_API"
# Please refer to Java Docs for more information
#
#
from java.util import *
from java.io import *

# Create a DTO from the name of the table
# (Make sure the XML file exists)
dto = wdbmgrapi.createDTOfromTable( tabName )
# Prepare the query to the database
dto.setColumnValue( colName, colValue );

# Order BY
orderDTO = wdbmgrapi.createDTOfromTable( tabName )
# Prepare the query to the database
orderDTO.setColumnValue( colOrder, "ASC" );

# Connect to database
print "Searching for entries in table %s where %s = %s" % (tabName, colName,
colValue)
print 'Number of entries in table "%s": %d' %(tabName, wdbmgrapi.getCountOf
dto))
```

```
try:
  objWDBCache = wdbmgrapi.getAllObjects(dto, orderDTO)
  print "Count: %d" % objWDBCache.size()
  txtFile = File(".")
  # Use API method (we are lazy, aren't we)
  print wdbmgrapi.writeTXT( txtFile, objWDBCache )
  print "TXT file written: %s" %(txtFile.getAbsolutePath())
  itObjs = objWDBCache.iterator()
#  for i in itObjs:
#    print "QuestID: %d, QuestNName: %s" %
(i.getColumnValue("wdb_QuestId"),i.getColumnValue("wdb_name"))

except:
  print "No data found"
```

Thanks to

Andrikos for the 1.6.x specs of the WDB files
Pyro's WDBViewer
WDDG Forum
blizzhackers
www.wddg-ger-translators.de.vu
etc etc

Special Thanks to:
Yodan, fritzpille, pifnet-pif, Ogite, hardboil, lilly, biene and lillyhunter

This program uses:

Kunststoff LF 2.0
Hypersonic SQL database 1.7.3
Apache Log4J, Xerces
Jakarta Commons Logging and CLI\n" + "Castor from ExoLab
MySQL JDBC connector 3.1.7
Oracle 9.2.0.3 JDBC connector for jdk_1.4
Jython 2.1

FAQ

I get an Error, when I try to open itemcache.wdb (quest.wdb, etc).

OutOfMemoryError

Use

```
java -Xmx512M -jar wdbearmanager.jar ...
```

This adds 512M of RAM to the Java virtual machine.

I try to load a WDB file, but the file seems corrupted. Maybe WDBManager is broken?

Since 1.6.0 the WDB format has changed. Before there was an ID of 16 bytes, but with 1.6.0 the ID is 20 bytes. Please check your WDB files and your XML files.
You can specify the "skipBytes" parameter inside the XML files. It should be 16 for pre 1.6.0 and 20 for 1.6.x and above.

I am using MySQL 5 and the WDBearManager always quits with an error.
(I am using version 1.x or a PRE version)

MySQL:
If you are using MySQL 5, then you must download the new JDBC driver from

http://dev.mysql.com/downloads/connector/j/3.1.html

Please unzip the package and rename the JAR to "mysql-connector-java-3.0.8.jar" and copy it to "deploy/jars".

This replaces the original 4.x driver with a new version, since MySQL
5.x changed the authoriziation procedure.

btw:
Use MySQL 4.x and the app will run w/o changes.

Since version 2.0 the manager supports MySQL 4 and 5.