Kin DB Import

Kin DB import tool (*createdb*)

Universe definition

createdb tool creates all the storage files to hold a database defined on a definition text file. KinDB can then be started and will load the created universe and objects. Definition file has the following format (driven by the example):

```
##############################
#
# Example Record Definition
#
# This is definition for all the ExampleDB data base
#
# Keywords are case insensitive, field names are also case insensitive. By now,
# it is recommended that names you define to be standard ASCII (7bit) letters
#
# Remember also that field length is not length limited, but shorter names are
# easier to write and remember...
#
# Comments are allowed, from the pound sign to the end of line
#

Universe       ExampleDB

# First, we define a special database objects. These MUST exist in any universe,
# as it is used internally by some data types, and for DB login and client
# identification. You can discover them by the underscore that precedes the name.
#
# For the _USER object, compulsory fields are:
# - Usr, has to be sID, name is not relevant (so no underscore is used)
# - _lName, has to be some kind of text field, name has to be exactly this
# - _lPasswd, has to be some kind of text field, name has to be exactly this
# The rest of fields are just application fields, you define what you need
#

###############################
# User object

RECORD _User                    # Start object record definition, name: "_User"
  Usr       sID                 # Record ID, a field called "Usr"
  Dele      rsID        Empres  # This field is a short ID, and refers to another
                                #  object's records (Empres)
  Crea      uDTcrea             # Record creation timestamp (automatic, read only)
  Modi      uDTmodi             # Record modification timestamp (automatic, read only)
  Nom       String8b    Noms    # Variable length 8bit text, refers to object "Noms"
  Cog[2]    String8b    Cognoms # Idem, two fields referring to one object "Cognoms"
  Born      sDate
  _lName    fText8b(16)         # Fixed size embedded text, Login Name (predefined)
  _lPasswd  fText8b(8)          # Login Password (predefined)
/RECORD                         # End object definition
```

```
###############################
# Interface data definitions
# This is for the Web Client Interface, for this application

RECORD _WCIdd                   # Web Client Interface Dialog Definitions (predefined)
  _Name     fText8b(32)         # Application name (Predefined)
  *St       Byte                # Historic field with status
  _XMLdef   Block       XMLdd   # XML source for this application (predefined)
/RECORD

OBJECT XMLdd                    # Web Client Interface XML Dialog definitions
  Block                         # Variable length (0 bytes to 4 GBytes) data
  Base8                         # Minimum granularity is 8bit (stores bytes - UTF8 text)
/OBJECT

###############################
# Particular universe database definitions

### Empr object (Points of Sale, etc.)

RECORD Empr                     # Start object record definition, name: "Empr"
  Dele      sID                 # Record ID (automatic)
  Crea      uDTcrea             # Record creation timestamp (automatic, read only)
  Modi      uDTmodi             # Record modification timestamp (automatic, read only)
  nEmp      String8b    NomE    # Name of the company
  nDel      String8b    NomE    # Name of the branch, local name or site name
/RECORD                         # End object definition

### Cli object (customers)

RECORD Cli
  ID        rID                 # Record ID
  +Dele     rsID        Empres  # Company or branch that got this client in. A view (+).
  Crea      uDTcrea             # Record creation timestamp (automatic, read only)
  Modi      uDTmodi             # Record modification timestamp (automatic, read only)
  uMod      muID                # Automatic Last user that modified this record (read only)
  uCre      cuID                # Automatic Last user that created this record (read only)
  Nom       String8b    Noms    # Customer Name
  Cog[2]    String8b    Cogs    # Customer Surnames
  Tel[4]    PhoneStr    Telefs  # Phone numbers
/RECORD

# Now we define all the variable size objects.
# They only require a simple declaration, with some attributes

OBJECT Noms                     # Names
  String8b                      # Short 8bit variable length text
  CaseInsensitive               # Do not take care about case when matching
/OBJECT

OBJECT Cogs                     # Surnames
  String8b                      # Short 8bit variable length text
  CaseInsensitive               # Do not take care about case when matching
/OBJECT
```

```
OBJECT Telefs                   # Telephone numbers
  PhoneStr                      # Short 8bit variable length text processed as only numbers
/OBJECT

OBJECT Addr                     # Addresses
  String8b                      # Short 8bit variable length text
  CaseInsensitive               # Do not take care about case when matching
/OBJECT
```

Source data definition

createdb tool also can import data into those created objects. Most of the data to import is read using XML laws, with the exception of binary objects. Document Type Definitions (DTD) for the different data types that can be imported follow:

- String8b

  creadb_string8b.dtd:

```
<?xml version="1.0" ?>
<!ELEMENT String8b (t+)>
 <!ATTLIST String8b name CDATA #REQUIRED>
<!ELEMENT t (#PCDATA)>
 <!ATTLIST t ID CDATA #REQUIRED>
```

- oRec

  creadb_orec.dtd:

```
<?xml version="1.0" ?>
<!ELEMENT orec (fd*,r+)>
 <!ATTLIST orec name CDATA #REQUIRED>
<!ELEMENT fd (#PCDATA)>
 <!ATTLIST fd name CDATA #REQUIRED>
 <!ATTLIST fd size CDATA "">
 <!ATTLIST fd type (trID|tsID|trrID|trsID|tmuID|tcuID|tsByte|tsWord|tsInt|tsLong|tByte|tWord|
tInt|tLong|tBitMap|tuDateTime|txDateTime|tsDate|tTime|tsTIme|tuDTmodi|tuDTcrea|tfText8b|
tString8b|tlString8b|tPhoneStr|tImage|tBlock)>
 <!ATTLIST fd attr (none|historical) "none">
 <!ATTLIST fd ref CDATA #REQUIRED>
<!ELEMENT r (f+) (#PCDATA)>
 <!ATTLIST r rID CDATA>
<!ELEMENT f (#PCDATA)>
 <!ATTLIST f name CDATA #REQUIRED>
<!ELEMENT fh (#PCDATA)>
 <!ATTLIST fh name CDATA #REQUIRED>
 <!ATTLIST fh ts CDATA #REQUIRED>
 <!ATTLIST fh uid CDATA #REQUIRED>
```

- Block

Some examples follow, for every kind of data type:

- String8b:

  Object name: noms

Filename: noms.xml

```
<?xml version="1.0" encoding="iso8859-1" ?>
<!DOCTYPE String8b SYSTEM "creadb_String8b.dtd">
<String8b name="noms">
<t ID="1">Mª Lourdes</t>
<t ID="2">Francisco</t>
<t ID="3">Juan</t>
  (...)
</String8b>
```

- oRec:

Object name: _WCIdd

Filename: _WCIdd.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE orec SYSTEM "createdb_orec.dtd">
<orec name="_WCIdd">
 <fd name="_Name" type="fText8b(32)"></fd>
 <fd name="_XMLdef" type="Block" ref="XMLdd">0</fd>
 <fd name="St" type="Byte" attr="historic"></fd>

 <r rID="1">
  <f name="_Name">base</f>
  <f name="_XMLdef">1</f>
  <f name="St">5</f>
   <fh name="St" ts="20070801000000" uid="1">4</f>
   <fh name="St" ts="20070401000001" uid="2">3</f>
 </r>

</orec>
```

<fd> is used for Field Definition. A default value may be supplied as the content of the tag. *ref* tag is for referencing fields (like tString8b), naming the referenced object.

<r> is used to define values for one record each, being <f> its fields values, and *rID* its record ID (might be omitted, being a serial number then)

Historic fields can get their values using <fh> tags, like in <fh name="_Name" ts="20070920153059" uid="1">This is _Name value until user '1' changed it on September 20th, 2007</fh>

- Block:

As this items are binary data, it can't be expressed in XML documents, at least with no encoding. For easy manual creation of import files, *createdb* uses two different approaches for import (you may use the one that is easier to manage for you):

1. A file system approach: instead of reading an xml file named as the object to import, it looks for a directory of that name, and then imports every single file in it, as items. Thus, files should be named as rID, e.g. there will be files named “1”, “2”, etc. containing item value for rID=1,2,etc. So in the example, we have:

Object name: XMLdd

Directory name: XMLdd/

File Names: 1, 2, 3, 4, 5... (with no extension at all)

2. A single big tagged binary file. Every block is encapsulated into a tag, or preceeded by a header. Format is simple, just a 32bit value with the data size of the block, and another 32bit value with its ID, then raw data for it, and finally, 0 to 3 bytes of padding (next block must start on a 32bit boudary). It would be something like this:

Sz (5)	ID (1)	Block 1 Data (5 bytes)	Padding (3 B)	Sz (28)	ID (2)	...

In the example, the file would be XMLdd.dta