VALENTINA 1.2.1

written by Ruslan Zasukhin,
Paradigma (www.paradigmasoft.com)

(Tutorial)

Acknowledgments:

Brian Blood	**-** for help in development of this product.
Andy Bachorski	**-** consultant, Apple Computer, Inc.
Hideaki Iimori	- for help in internationalization

Example 1: Creation of database

This example demonstrates how you can create a new empty database, then add Tables to the database and define fields of the Tables. Notice, Valentina uses the terming ‘base object’ as ‘Table’.

To create a database you must specify its location on the hard disk. You can do this in 2 ways:
1) ask the user about location via standard dialog as shown in the example script;
2) specify some location directly in the script. The following line will create the database files at the location of the Valentina application:

```
     set theDB to make new database with data file “Customers db”
```

For string fields you can specify the language which will be used for the field.

The Table Customer is related to the Table Invoice as One to Many via field of special type tObjectPtr. For this relation we have choose deletion control as delete_many, i.e. if a record of Customer will be deleted then automatically will be deleted all Invoices related to it.

Obviously that function Create() will be called only once when you create a new database, later you will open the existing database.

```
-------------------------------
Create()

tell application «Valentina 1.2.1 (PPC)»
  close database «Customers.vdb»
end tell
-------------------------------
on Create()
  tell application «Valentina 1.2.1 (PPC)»
    activate
    set theSpec to new file with prompt «New database file» default name «Customers db»
    set theDB to make new database with data theSpec

    tell theDB
       set Customer to make new base object with properties {name:«Customer»} at end
       set Invoice to make new base object with properties {name:«Invoice»} at end

       tell Customer
         make new field with properties
              {name:«Name», type:tString, length:30, language:3} at end
         make new field with properties
              {name:«Address», type:tString, length:50, language:«German»} at end
         make new field with properties
              {name:«Photo», type:tBlob} at end
       end tell

      tell Invoice
         make new field with properties {name:«CustomerPTR», type:tObjectPtr,
    pointed object:Customer, deletion control:delete_many} at end
         make new field with properties {name:«Date», type:tDate} at end
         make new field with properties {name:«Total», type:tFloat} at end
      end tell
    end tell

  end tell
end Create
```

Example 2: Adding records

Now when we have Tables we can add records to it.

At first we open the database, we can again use 2 ways for this:
1) via standard dialog to allow user to choose the database to open;
2) specify location of the database directly in code;

Notice using of the timeout statement. It can be useful if you have a big database and operations takes long time.

In the following example we at first add a new record to the Table Customer then add several records to the Table Invoice relating them with current record of Table Customer. As a result, we create 5 records of Customers each of which is related with 3 records of Invoices.

As contents of the BLOB field “Photo” we use here the string of chars.

```
-------------------------------
property photo :
«klfdjghlkdfjglksdfjglskd;fjgls;kdfjgl;kdsfjglsdkf;gjlk;fsdjglnmvbld;skfjgsldkflnmdks»
-- this is psevdo-photo, of course, :-)

with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  AddRecords()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell
end timeout

-------------------------
on AddRecords()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Customer to base object «Customer» of DB
    set Invoice to base object «Invoice» of DB

    repeat with i from 1 to 5
        set field «Name» of Customer to i as string
        set field «Address» of Customer to «some address»
        set field «Photo» of Customer to photo
        make new record at end of Customer

        repeat with k from 1 to 3
            set field «CustomerPTR» of Invoice to current record of Customer
            set field «Date» of Invoice to date «3/3/99»
            set field «Total» of Invoice to k * i
            make new record at end of Invoice
        end repeat
    end repeat
  end tell
end AddRecords
```

Example 3: Faster adding

This example is just optimization of previous one.

We set the values of the fields not via separate AppleEvent but via single one: for this we use plural form ‘fields’ and pass the values of the fields as list.

If in the previous example on one new records we send 4 AppleEvents:

```
set field «Name» of Customer to i as string
set field «Address» of Customer to «some address»
set field «Photo» of Customer to photo
make new record at end of Customer
```

then now only 2.:

```
set fields of Customer to {i as string, «some address», photo}
make new record at end of Customer
```

If a Table will have many fields, say 10-15, then you again will need only 2 AppleEvents instead of 11-16.

```
-------------------------------
property photo :
«klfdjghlkdfjglksdfjglskd;fjgls;kdfjgl;kdsfjglsdkf;gjlk;fsdjglnmvbld;skfjgsldkflnmdks»
— this is psevdo-photo, of course, :-)

with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  AddRecords()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell

end timeout

-------------------------------
on AddRecords()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Customer to base object «Customer» of DB
    set Invoice to base object «Invoice» of DB

    repeat with i from 1 to 5
         set fields of Customer to {i as string, «some address», photo}
         make new record at end of Customer

         repeat with k from 1 to 3
              set fields of Invoice to {current record of Customer, date «3/3/99», k * i}
              make new record at end of Invoice
         end repeat
    end repeat
  end tell
end AddRecords
```

Example 4: Search and sort

When you have records in the Table you must be able to select some of them which match to the conditions. The following example demonstrates how you can select records and then sort the selection on one or several fields.

Notice, as a search condition you must specify a string, for numeric fields you can write:
```
     theNumericValue as String
```

The routine RelatedSearch() demonstrates how you can find for a record its related records.

```
---------------------------------
with timeout of 300 seconds
   tell application «Valentina 1.2.1 (PPC)»
      set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
      set DB to open theFile
   end tell

   Search()
   RelatedSearch()

   tell application «Valentina 1.2.1 (PPC)»
      close DB
   end tell
end timeout
---------------------------------
on Search()
   tell application «Valentina 1.2.1 (PPC)»
      set DB to database «Customers.vdb»

      tell base object «Invoice» of DB
            set ResultSet to select records where { field «Total», «>=3 and <8»,
                                                    field «Date»,  «>01/01/1999»}

            set SortedSet to sort ResultSet by {field «Date», field «Total»}
            -- ATTENTION: “sort” delete the previous selection and returns a new sorted one.
            -- so you MUST get reference on it to later free memory.
      end tell

      get count of records in SortedSet
      delete SortedSet -- free memory
   end tell
end Search
-------------------------------------------
on RelatedSearch()
   tell application «Valentina 1.2.1 (PPC)»
      set DB to database «Customers.vdb»
      set Customer to base object «Customer» of DB

      set AllCustomers to select records of Customer – select all customers in Table

      -- now go to the record of first customer
      set current record of Customer to record 1 of AllCustomers

      -- select all invoices of the first Customer
      tell base object «Invoice» of DB
         -- note, for tObjectPtr field as search condition we pass not string,
         -- but record, for all other types of field we must pass a string condition.
         set InvoivesSet to select records where {field «CustomerPTR», current record of Customer}
      end tell

      get count of records in InvoivesSet
      delete InvoivesSet   -- free memory
      delete AllCustomers  -- free memory
   end tell
end RelatedSearch
```

Example 5: Iteration of selcted records

This example demonstrates how to iterate a selection of records.
At first we select all records in the Table Customer, then sort the selection on field “Name”.
In the loop we go through each selected record and get values of the fields.

When some record becomes current this means that its contents (exclude BLOB-fields) is loaded to the memory buffer of the Table.

On the command ‘get fields’ your script get values of all fields of the Table including value of BLOB field as list. The values in the list has type corresponded to type of the field, i.e. value of the string field is returned as string, values of numeric fields are returned as numbers, values of date/time as date, ...

```
with timeout of 300 seconds

   tell application «Valentina 1.2.1 (PPC)»
      set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
      set DB to open theFile
   end tell

   Iterate()

   tell application «Valentina 1.2.1 (PPC)»
      close DB
   end tell

end timeout
```

```
on Iterate()
   tell application «Valentina 1.2.1 (PPC)»
      set DB to database «Customers.vdb»
      set Customer to base object «Customer» of DB

      tell Customer
            set ResultSet to select records — select all records
            set SortedSet to sort ResultSet by {field «Name»}
      end tell

      set theCount to count of records in SortedSet
      repeat with i from 1 to theCount
            tell Customer
                  — perform «go to» record
                  set current record to record i of SortedSet

                  set FieldsList to fields
                        -> { “Piter”, “his adress”, photo }

                  -- assign values of the field to script variable
                  set theName to item 1 of FieldsList
                  set theAdress to item 2 of FieldsList
                  set thePhoto to item 3 of FieldsList
            end tell
      end repeat

      delete SortedSet -- free memory
   end tell
end Iterate
```

Example 6: Faster iteration

This example is optimization of the previous one. It shows how you can get many records via single AppleEvent. For this you should get range of records from the selection. As a result you get list of lists – list of records, where each record is represented as list of field values.
To get a record from this list you can write:
        get item N from List
In the same way you can get values of the fields from the list.

This technic can be very useful for WEB: as a result of the search you return first 10 records, then next 10, ...

If the result of the search is many records (thousands) then you should not use range 1..Count because you can get out of memory error.

```
with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  Iterate()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell

end timeout
_______________________________
on Iterate()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Customer to base object «Customer» of DB

    tell Customer
        set ResultSet to select records -- select all records
        set SortedSet to sort ResultSet by {field «Name»}
    end tell

    set theCount to count of records in SortedSet
    tell Customer
        set RecordsSet to records 1 thru theCount of SortedSet
            -> { {“John”, “adress1”, Photo1 },
                 {“Kee”, “adress2”, Photo2 },
                 {“Brian”, “address3”, Photo3}}
    end tell

    set theRecord to item 1 of RecordsSet
        -> {“John”, “adress1”, Photo1 }

    delete SortedSet -- free memory
  end tell
end Iterate
```

Example 7: Updating of records

This example shows how update values of the fields for existing records.

At first we must make a record current, so Valentina loads it to the memory buffer.
Then we assign new values of the fields using the standard AppleScript command ‘set’.
Now new values are in the memory buffer.
To save this changes on the disk we must make the command ‘update’.

This example also demonstrates using of the command ‘replace value’.
This command is useful if you need to change values of one field in many records.
At first you must get selection of records, then put a new value of the field in the memory buffer and make the command ‘replace value’.

```
-------------------------------
with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  UpdateTotal()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell

end timeout

_______________________________
on UpdateTotal()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Invoice to base object «Invoice» of DB

    tell Invoice
         set S1 to select records where {field «Total», «<5»}
    end tell

    (*set theCount to count of records in S1
    repeat with i from 1 to theCount
         set current record of Invoice to record i of S1
         set field «Total» of Invoice to 12
         update Invoice
    end repeat*)

    -- this loop will be better write via command “replace value”:
    tell Invoice
         set field «Total» to 12
         replace value for field «Total» in S1
    end tell

    delete S1 -- free memory
  end tell
end UpdateTotal
```

Example 8: Updating of BLOB fields

This example shows how update the contents of the BLOB fields.

The deal is that contents of the BLOB field (pict, text, sound...) is stored in the separate file, in the records of the Table is stored only reference (4 byte) on the corresponded contents. If a record have not contents of a BLOB field (for example Customer still have not photo) then the reference is NULL. You never will work with the value of this reference – this is responsibility of Valentina.

So to update the BLOB field you need 2 steps:
1) by command 'update' change the contents of the BLOB field;
2) by command 'update' save into the Table new value of the reference.

```
------------------------------
property photo :
«klfdjghlkdfjglksdfjglskd;fjgls;kdfjgl;kdsfjglsdkf;gjlk;fsdjglnmvbld;skfjgsldkflnmdks»

property photo2 :
«oiuetryopeiuopierwupoiweuo[poreitpw[oity[peorty[peorty[porjkhlhkljkhjkhj;kh;h;;etyopertyi[peortyie[porty»

with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  UpdatePhotos()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell

end timeout
——————————————————————————————-
on UpdatePhotos()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Customer to base object «Customer» of DB

    set AllCustomers to select records of Customer   -- select all customers in Table
    set theCount to count of records in AllCustomers -- how much customers in Table ?

    repeat with i from 1 to theCount
         set current record of Customer to record i of AllCustomers

         update field «Photo» of Customer with photo2
         update Customer
    end repeat

    delete AllCustomers -- free memory
  end tell
end
```

Example 9: Deletion of records

This example demonstrate how you can delete records of the Table.
You can delete the current record of the Table or some record of the selection.

While Tables Customer and Invoices are related and we have specify deletion control for the field Invoice.CustomerPTR as ‘delete_many’ (see Example 1) when we delete a record of Customer, Valentina automatically deletes all its related Invoice records.

```
-------------------------------
with timeout of 300 seconds

   tell application «Valentina 1.2.1 (PPC)»
      set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
      set DB to open theFile
   end tell

   DeleteRecords()

   tell application «Valentina 1.2.1 (PPC)»
      close database «Customers.vdb»
   end tell

end timeout
———————————————————————————————
on DeleteRecords()
   tell application «Valentina 1.2.1 (PPC)»
      set DB to database «Customers.vdb»
      set Customer to base object «Customer» of DB

      get count of records in Customer -- see how much records are now ?
      get count of records in base object «Invoice» of DB

      set AllCustomers to select records of Customer -- select all customers in Table
      set theCount to count of records in AllCustomers -- how much customers in Table now?

       -- delete half of records
      repeat with i from 1 to theCount div 2
            -- we can write:
            -- set current record of Customer to record i of AllCustomers
            -- delete current record of Customer

            -- this is more short code:
            delete record i of AllCustomers
      end repeat

      -- see how much records now in db
      get count of records in Customer -- see how much records are left ?
      get count of records in base object «Invoice» of DB

      -- delete all rest records
      delete records of Customer

      delete AllCustomers -- free memory
   end tell
end DeleteRecords
```

Example 10: Get info

This example shows how you can get information about structure of the database:
- number of Tables;
- names of the Tables;
- number of the fields in the Tables;
- names and flags of the fields.

```
-------------------------------
with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  GetInfo()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell

end timeout
———————————————————————————————
on GetInfo()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set ObjectsCount to count of base objects in DB

    repeat with i from 1 to ObjectsCount
        set theObject to base object i of DB
        tell theObject
            get name of theObject
            get count of records in theObject

            set FieldsCount to count of fields in theObject
            repeat with k from 1 to FieldsCount
                get name of field i of theObject
                get type of field i of theObject
                get indexed of field i of theObject
                get unique of field i of theObject
                -- note, we can’t get reference theField like theObject,
                -- because “get field i of theObject” returns contents of the field.
            end repeat
        end tell

    end repeat

  end tell
end GetInfo
```

Example 11: Export/Import

This example shows how you can export selected records to the ASCII file or import new records to the Table.

Here we at first ask the user where must be located an exported file then perform export of the selected fields. Note records are exported with keeping of sort order of the Selection, so you, can for example, export records of Customer sorted by the Name.

```
-------------------------------
with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  ExportCustomers()
  ImportCustomers()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell

end timeout
───────────────────────────────
on ImportCustomers()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Customer to base object «Customer» of DB

    set theFile to choose file with prompt «File to import» of type {«TEXT»}

    tell Customer
         import from ascii file theFile to {field «Name», field «Address»}
    end tell

  end tell
end ImportCustomers
───────────────────────────────
on ExportCustomers()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Customer to base object «Customer» of DB

    set theSpec to new file with prompt «Export to» default name «Customers backup»

    tell Customer
         set S1 to select records
         set S1 to sort S1 by { field "Name" }
         export from {field «Name», field «Address»} using S1 to ascii file theSpec
    end tell

  end tell
end ExportCustomers
```

Example 12: Changing of database structure

Here you can see how is possible to change the structure of the database.
Valentina give you full control on the database structure via AppleScript.
You can add new Tables, remove old, add/remove fields to the Tables, change any parameter of the fields.

Notice, you can do changes runtime when a database have data already.

```
-------------------------------
with timeout of 300 seconds

  tell application «Valentina 1.2.1 (PPC)»
    set theFile to choose file with prompt «Choose database file» of type {«Vdsc»}
    set DB to open theFile
  end tell

  ChangeStructure()

  tell application «Valentina 1.2.1 (PPC)»
    close DB
  end tell

end timeout
———————————————————————————————
on ChangeStructure()
  tell application «Valentina 1.2.1 (PPC)»
    set DB to database «Customers.vdb»
    set Customer to base object «Customer» of DB
    set Invoice to base object «Invoice» of DB

    tell Customer
        get length of field «Name»
        set length of field «Name» to 55 -- change length of the string field.

        get length of field «Name»
    end tell

    tell Customer
        delete field «Address»  -- delete the field from the table
    end tell

    delete Invoice -- delete table Invoice.

  end tell
end ChangeStructure
```

Example 13: Multiple open database

This example demonstrates how you can work with multiple open database.
Here we create 2 databases with the same structure, add records to the first database then copy that records to the second database.

```
----------------------------------------------------
property photo :
«klfdjghlkdfjglksdfjglskd;fjgls;kdfjgl;kdsfjglsdkf;gjlk;fsdjglnmvbld;skfjgsldkflnmdks»

with timeout of 3000 seconds
  Create(«Customers»)
  AddRecords()

  Create(«Archive»)
  CopyRecords()

  tell application «Valentina 1.2.1 (PPC)»
    close database «Customers.vdb»
    close database «Archive.vdb»
  end tell
end timeout

______________________________________

on Create(DataBaseName)
  tell application «Valentina 1.2.1 (PPC)»
    activate
    set theSpec to new file with prompt «New database file» default name DataBaseName
    set theDB to make new database with data theSpec

    tell theDB
         set Customer to make new base object with properties {name:«Customer»} at end
         set Invoice to make new base object with properties {name:«Invoice»} at end

         tell Customer
              make new field with properties
                   {name:«Name», type:tString, length:30} at end
              make new field with properties
                   {name:«Address», type:tString, length:50} at end
              make new field with properties
                   {name:«Photo», type:tBlob} at end
         end tell

         tell Invoice
              make new field with properties
                   {name:«CustomerPTR», type:tObjectPtr, pointed object:Customer} at end
              make new field with properties
                   {name:«Date», type:tDate} at end
              make new field with properties
                   {name:«Total», type:tFloat} at end
         end tell
    end tell
  end tell
end Create
```

```
on AddRecords()
  tell application «Valentina 1.2.1 (PPC)»
    set Customer to base object «Customer» of database «Customers.vdb»
    set Invoice to base object «Invoice» of database «Customers.vdb»

    repeat with i from 1 to 10
        set fields of Customer to {i as string, «some address», photo}
        make new record at end of Customer

        repeat with k from 1 to 3
            set fields of Invoice to {current record of Customer, date 3/3/99, k * i}
            make new record at end of Invoice
        end repeat
    end repeat

    -- after adding of many records we must flush database
    flush database «Customers.vdb»

  end tell
end AddRecords
```

```
on CopyRecords()
  tell application «Valentina 1.2.1 (PPC)»
    set Customer to base object «Customer» of database «Customers.vdb»
    set ArchiveCustomer to base object «Customer» of database «Archive.vdb»

    set S to select records of Customer
    set theCount to count of records in S

    -- delete all records in table ArchiveCustomer
    delete records of ArchiveCustomer

    -- now copy records:
    repeat with i from 1 to theCount
        set current record of Customer to record i of S
        set theValues to fields of Customer
        set fields of ArchiveCustomer to theValues
        make new record at end of ArchiveCustomer
    end repeat


    flush Customer     -- after adding of many records we must flush base object

  end tell
end CopyRecords
```