Miscellaneous Notes

Abbreviations

• aka Also Known As

• DBMS DataBase Management System
Administrivia

Please read and highlight BEFORE lab

• Assignment 11
• Lecture Notes

This week

• Scripting in FileMaker
• Regular Expressions
• Project Pearl

Please ask questions if there’s something you don’t understand
Assumptions

• You have completed Assignment 9 and have an understanding of FileMaker.
• You have an understanding of scripting and automating tasks in an application.
Things to Think about

How do we manipulate the data objects with a script?
Is there more than one way to create a script?
What are the deficiencies of the scripting interface?
What are the efficiencies of the scripting interface?
Application Scripting

Application scripting is a way

• to automate lengthy manual procedures
• to automate error prone manual procedures
  eg sort & enter Preview Mode to display/print a report in FileMaker
• to customize / extend an application
• to create a different interface
• for an application designer to hide details from an application user

Using a “textual representation” of every possible user action

• or at least most user actions...

“Digital player pianos”

Before we look at scripting in Filemaker...

• “why scripting again?” and
• “what do we expect?”
What do we expect to find? (“What’s our model of scripting?”)

Commands (statements) equivalent to user actions

Variables

Loops & Conditionals
• loops so we can do something repeatedly
• if-stmts so we can make decisions about what to do

The ability to invoke one script from another, so we can
• easily reuse scripts
• write small scripts to do simple things, & combine them

The ability to (manually) create and edit scripts

Recordability

A way to attach scripts to buttons, menus, and/or keystrokes

Debugging support

The ability to customize the user interface
### FileMaker Script Steps (= “statements”)

#### Control
- Perform Script
- Pause/Resume Script
- Exit Script
- Halt Script
- If
- Else If
- Else
- End If
- Loop
- Exit Loop If
- End Loop
- Allow User to Abort
- Set Error Capture
- Set Variable
- Install OnTimer Script

#### Navigation
- Go to Layout
- Go to Record/Request/Page
- Go to Related Record
- Go to Portal Row
- Go to Field
- Go to Object
- Go to Next Field
- Go to Previous Field
- Enter Browse Mode
- Enter Find Mode
- Enter Preview Mode

#### Editing
- Undo/Redo
- Cut
- Copy
- Paste
- Clear
- Set Selection
- Select All
- Perform Find/Replace

#### Fields
- Set Field
- Set Field by Name
- Set Next Serial Value
- Insert Text
- Insert Calculated Result
- Insert from Index
- Insert from Last Visited
- Insert from URL
- Insert Current Date
- Insert Current Time
- Insert Current User Name
- Insert Picture
- Insert QuickTime
- Insert Audio/Video
- Insert PDF
- Insert File
- Replace Field Contents
- Relookup Field Contents
- Export Field Contents

#### Records
- New Record/Request
- Duplicate Record/Request
- Delete Record/Request
- Delete Portal Row
- Delete All Records
- Open Record/Request
- Revert Record/Request
- Commit Records/Requests
- Copy Record/Request
- Copy All Records/Requests
- Import Records
- Export Records
- Save Records As Excel
- Save Records As PDF
- Save Records as Snapshot Link

#### Found Sets
- Perform Find
- Constrain Found Set
- Extend Found Set
- Modify Last Find
- Show All Records
- Show Omitted Only
- Omit Record
- Omit Multiple Records
- Sort Records
- Sort Records by Field
- Unsort Records
- Find Matching Records

#### Windows
- New Window
- Select Window
- Close Window
- Adjust Window
- Move/Resize Window
- Arrange All Windows
- Freeze Window
- Refresh Window
- Scroll Window
- Show/Hide Toolbars
- Show/Hide Text Ruler
- Set Window Title
- Set Zoom Level
- View As

#### Files
- New File
- Open File
- Close File
- Convert File
- Set Multi-User
- Set Use System Formats
- Save a Copy as
- Recover File
- Print Setup
- Print

#### Accounts
- Add Account
- Delete Account
- Reset Account Password
- Change Password
- Enable Account
- Re-Login

#### Spelling
- Check Selection
- Check Record
- Check Found Set
- Correct Word
- Spelling Options
- Select Dictionaries
- Edit user Dictionary

#### Open Menu Item
- Open Edit Saved Finds
- Open Preferences
- Open File Options
- Open Manage Containers
- Open Manage Database
- Open Manage Data Sources
- Open Manage Layouts
- Open Manage Value Lists
- Open Find/Replace
- Open Help
- Open Remote
- Open Sharing

#### Miscellaneous
- Show Custom Dialog
- Allow Formatting Bar
- Beep
- Speak
- Dial Phone
- Install Plug-In File
- Install Menu Set
- Set Web Viewer
- Open URL
- Send Mail
- Perform AppleScript
- Execute SQL
- Send Event
- Comment
- Flush Cache to Disk
- Exit Application
Evaluation

Variables

the clipboard

“Global Fields” (see last week’s assignment)

“local variables” ($name)

“global variables” ($$name)

Loops & Conditionals

The ability to invoke one script from another

“Perform Script” does the trick

The ability to create and edit scripts

✔

Recordability

no (boo....)

A way to attach scripts to buttons, menus, and/or keystrokes

✔, ✔, ✔

Debugging support

only in FileMaker Pro Advanced (sigh...)

Customize the interface

✔

A local variable can only be used in script steps in the currently executing script. The value in a local variable is cleared when the script exits.

A global variable can be used in a calculation or script anywhere in a file— for example, in other scripts or file paths. The value of a global variable is not cleared until the file is closed.

Control

Perform Script
Pause/Resume Script
Exit Script
Halt Script
If
Else If
Else
End If
Loop
Exit Loop If
End Loop
Allow User Abort
Set Error Capture
Set Variable
Let’s look at a script that
• starting in the Album table
• and assuming you’ve selected a song,
• opens the Song table and
• displays detailed information about that selected song

To see the script, select Manage Scripts from the Scripts menu...
It’s straightforward to deduce

• how to edit an existing script
• how to create & start editing a new script
• how to delete a script

What’s the basic “interaction model” for this dialog?

Is there anything on this dialog that’s particularly worth remembering?
What’s the basic “interaction model” for this dialog?
### Script Step Parameters

**“Local Variables”**
- last only until the script returns
- are usable only in the same script
- start with $`

**“Global Variables”**
- last until the file is closed
- are usable only in the same file
- start with $$

### “Set Variable” Options

Names prefixed by "$" are local variables available only within the current script. Prefix the name with "$$" to make the variable available throughout the current file (global).

- **Name:** $SomeName variable
- **Value:** `Get(LayoutName)`
- **Repetition:** 1
Running A Script

Remember the menu checkbox in the Define Scripts dialog box?

Or, attach a script to a button

A useful button trick

- make the button transparent

- overlay it on another item
  eg a picture, or the top portal row

- or place the button *behind* fields for which editing has been turned off in Browse mode, making it easier to manipulate those fields in layout mode
Using the Transparent Button Trick in the Albums & Songs

The selected rectangle shown below is actually a transparent button

In Browse mode, the icon shown while hovering over a portal row now indicates you’re over a button

• for this to work, however, the button must be in front of the portal fields, or editing of the portal fields must be turned off (Inspector > Data > Behavior)
Returning to the Correct Layout

We now have two layouts that use the DisplayClickedSongRecord script to display Song info.

Here’s how we arrange to return easily to the Album layout whence we came:

```
1. Set Variable [ $cameFromLayout ; Value: Get(layoutName) ]
2. Set Variable [ $songID ; Value: Song Table::Song ID ]
3. Go to Layout [ "Song Table" (Song Table); Animation: None ]
4. Enter Find Mode [ Pause: Off ]
5. Set Field [ Song Table::Song ID ; $songID ]
6. Perform Find [ ]
```

Note that we’ve had to use a global field because we need to save a value from this script and make it available in another (namely ReturnToAlbum).
Debugging Techniques in FileMaker

“Print Statements”

• the Show Message script step

• the Beep script step

• create a global text field M on a visible layout & use Set Field to write msgs / values into M
  both append and replace are supported
  eg in the Marks Demo’s Choose layout

“Breakpoints”

• the Pause/Resume Script script step

• the Show Message script step

• the Show Custom Dialog step

• or use FileMaker Pro Advanced
FileMaker’s Scripting As Compared to Excel’s

**Similarities**

- you can automate most manual manipulations
- you can attach scripts to buttons & put them in a menu
- so you can extend both applications

**Differences**

- in FM recording is non-existent
- in FM editing is very constrained
  - is this good or bad?
- in FM you *can* pass parameters explicitly to user-defined scripts
  but the mechanism is clumsy
  
  see online help for the Get(ScriptParameter), Let(...) and Evaluate(...) functions
  if you’re interested

- Only FileMaker Pro Advanced has a step-by-step debugger like Excel’s
- Excel has a larger set of script steps & functions
Other Important DBMS Features
(Not relevant to this week’s assignment)

Network access

Field- or record-locking
  • prevent simultaneous modification by two users

Transaction support
  • several steps treated as an “atomic action”
  • eg: – $1000 from this account, + $1000 into that account
  • — all are executed (or not) as if a single step

Large amounts of data
  • eg gigabytes (billions) or terabytes (trillions)
The Project Pearl

It’s boring (and often overwhelming) to “sit down and learn a new application”

So a good strategy is to

• find a project you need to do for some (unrelated) reason,
• for which you can use the app (or feature) you want to learn,
• that’s of suitable size and
• for which you can allocate a bit of extra time.

It helps if the project is small, so you can accomplish it quickly, before losing interest or enthusiasm

(This is how many of the CS 200 lab exams get set 😊.)