FoxInCloud

Adaptation Tips and Tricks

Adapt your app safer and easier

AT20 Development Workshop – Day 1, Activity 9
## Agenda

<table>
<thead>
<tr>
<th>Section</th>
<th>Subject</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Global Stuff</td>
<td>3'</td>
</tr>
<tr>
<td>2</td>
<td>Callback</td>
<td>10'</td>
</tr>
<tr>
<td>3</td>
<td>Menu</td>
<td>3'</td>
</tr>
<tr>
<td>4</td>
<td>Security</td>
<td>6'</td>
</tr>
<tr>
<td>5</td>
<td>Tools</td>
<td>3'</td>
</tr>
</tbody>
</table>
Global stuff

- Move all application-wide command to your init procedure:
  ```
  xxxSets.Init():
  `PUBLIC
  `SET
  `OPEN DATABASE
  `USE <common table>
  `SQLconnect()
  `do menu.mpr
  ```

- Please get rid of full pathes! You can’t predict in which folder your production web app will live; in xxxSets.Init():
  ```
  set path to “relative to project home, all folders your app needs read from”
  eg.
  set path to “..\SharedLibs;..\Data;..\Forms;..\Images”
  
  do form “c:\my projects\project\Forms\form.scx” & no!
  thisForm.wForm(“form.scx”) & yes!
  ```
Instruction calling modal should be last in procedure:

```<code>
if condition1
    <code>
    if condition2
        <code>
            thisForm.wForm('some.scx', 'wFormCallBack')
        endif
    endif
endif
endproc
```

Any code after `thisForm.wForm()` executes right away in Web mode, not after `wFormCallBack` has executed.
Callback in independent procedure/function

Callback code can only live in a method of a form or form member

Independent procedures or functions are not supported (and will never be)

solutions here after…
Callback in independent procedure/function (…)

Solution 1: make it a form member

```plaintext
procedure independent(parm)
    if MessageBox(parms) = 6 && yes
        do stuff
    endif

define class variousProcs as xxxCst && (as custom)
    procedure independent(parm)
        thisForm.wMessageBox('wFormCallBack', parms)
        procedure wFormCallBack(userChoice)
            if m.userChoice = 6 && yes
                do stuff
            endif
        endif
    enddefine

procedure xxxFrm.Load && your form base class
    this.addObject('oProcs', 'variousProcs')

    Independent(parm)
    && to be replaced by
    thisForm.oProcs.independent(parm)
```

Question? Post with screenshot in the ‘FoxInCloud’ section of https://support.west-wind.com/ or http://www.universalthread.com/
Callback in independent procedure/function (...)

Solution 2: move call-back code to the calling object or one of its parent class

```
procedure independent (parm)
    if MessageBox(parms) = 6 && yes
        do stuff
    endif

procedure independent (parm, oCalling)
    oForm = oFormIn(m.oCalling) && in abOOP.prg
    oForm.wMessageBox(‘wFormCallBack’, parms)

procedure oCalling.wFormCallBack(userChoice)
    if m.userChoice = 6 && yes
        do stuff
    endif
```
Menu

- **Menu commands** must be passed as a string to `wMenu()` [awPublic.prg] which:
  - **Desktop mode**: executes via macro substitution
  - **Web mode**: turns into HTML using `jQueryUI`

- Any variable must be replaced by its literal value, eg using `Textmerge()`:

  ```pascal
  lcPrompt = "Display contents of myFile"
  DEFINE BAR 2 OF myPopup PROMPT m.lcPrompt && before

  wMenu(Textmerge('DEFINE BAR 2 OF myPopup PROMPT "<<m.lcPrompt>>"')) && after
  ```

- FoxInCloud’s menu generator (awMenuGen.prg, replacing MenuGen.prg as _genmenu) takes care of *.mnx
Security

- By default, anonymous access to Forms is disabled:
  \[ \text{aw Frm.wlAnonymousAllowed} = .F. \]

- Public app and/or development: set
  \[ \text{xxx Frm.wlAnonymousAllowed} = .T. \]
  at design time or run time:
  \[ \text{procedure xxx Frm.Load} \]
  \[ \text{if lDevMode()} \]
  \[ \text{this.wlAnonymousAllowed} = .T. \]
  \[ \text{endif} \]

- Private app in production:
  - Base class:
    \[ \text{xxx Frm.wlAnonymousAllowed} = .F. \]
  - splash and/or login form:
    \[ \text{splashLogin.wlAnonymousAllowed} = .T. \]
Security (...) user ID

- When user logs in, call
  ```javascript
  thisform.wUserLogin(<user ID for application>)
  ```
  Whatever the type of user ID

- If you store `<user ID>` in a global location like `public
variable, _screen.property, _vfp.property`,
make sure your server saves application environment:
  ```javascript
  xxxServer.lAppUserEnvSave = .T. && default
  ```

- If you store `<user ID>` in a form.property, make
sure it’s mentioned in `.wCPropSave`

- When user logs off, call
  ```javascript
  thisform.wUserLogoff()
  ```
Security (…) user session

- ‘User session’ is a place where FoxInCloud and you can store user-dependent data
  - eg. FoxInCloud stores the form dimension and position in the user’s session

- User session closes (expires) if:
  - User is idle during a delay that you can set remotely (default 30’) unless you choose permanent sessions
  - User logs in using the same account on a different machine-browser
Tools

- **GoFish**: [https://vfpx.codeplex.com/wikipage?title=GoFish](https://vfpx.codeplex.com/wikipage?title=GoFish)
  Useful to find occurrences of a given adaptation (besides FAA)

- **VFP debugger**:
  - FoxInCloud makes extensive use of **assert**
  - You can suspend the server execution anywhere to see what’s happening in your code

- **FoxInCloud utilities**
  - `abSet() / abOn()`: SETs and restore VFP setting
  - `wcPropSaveEdit()`: call in `form.Load()` or `member.init()` to **add/remove properties** to `.wcPropSave`
Free support on public forums below!

- Don’t stay stuck, you’re not alone!
- “does not work” is not enough; make sure to post:
  - Code
  - Screenshot
  - Description of what you’re trying to do
- Use FAA’s feedback procedure

Question? Post with screenshot in the ‘FoxInCloud’ section of https://support.west-wind.com/ or http://www.universalthread.com/