FoxInCloud

Adaptation Tips and Tricks

Adapt your app safer and easier

AT20 Development Workshop – Day 1, Activity 9

Agenda

Section	Subject	Duration
1	Global Stuff	3’
2	Callback	10’
3	Menu	3’
4	Security	6’
5	Tools	3’

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!
```

Callback

- Instruction calling modal should be last in procedure:

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

```
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
enddefine

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

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

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

For a complete documentation about `wMenu()`, read or execute `wMenu_cDoc()` in `awPublic.prg`

- 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()`:

```
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:
  `awFrm.wlAnonymousAllowed = .F.`

- Public app and/or development: set
  `xxxFrm.wlAnonymousAllowed = .T.`

  at design time or run time:
  ```
  procedure xxxFrm.Load
  if lDevMode()
    this.wlAnonymousAllowed = .T.
  endif
  ```

- Private app in production:
  - Base class:
    `xxxFrm.wlAnonymousAllowed = .F.`
  - splash and/or login form:
    `splashLogin.wlAnonymousAllowed = .T.`

Security (…) user ID

- When user logs in, call
  `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:
  `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
  `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 an set remotely (default 30’) unless you choose permanent sessions
  - User logs in using the same account on a different couple machine-browser

Tools

Make sure to get used to tools that will facilitate your adaptations and daily development with FoxInCloud!

- 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()`: `SET`s 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