FoxInCloud

What are adaptations for?

Filling the gap between a VFP app running on Desktop –vs– FoxInCloud Web Application Server.

AT20 Development Workshop – Day 1, Activity 2

Agenda

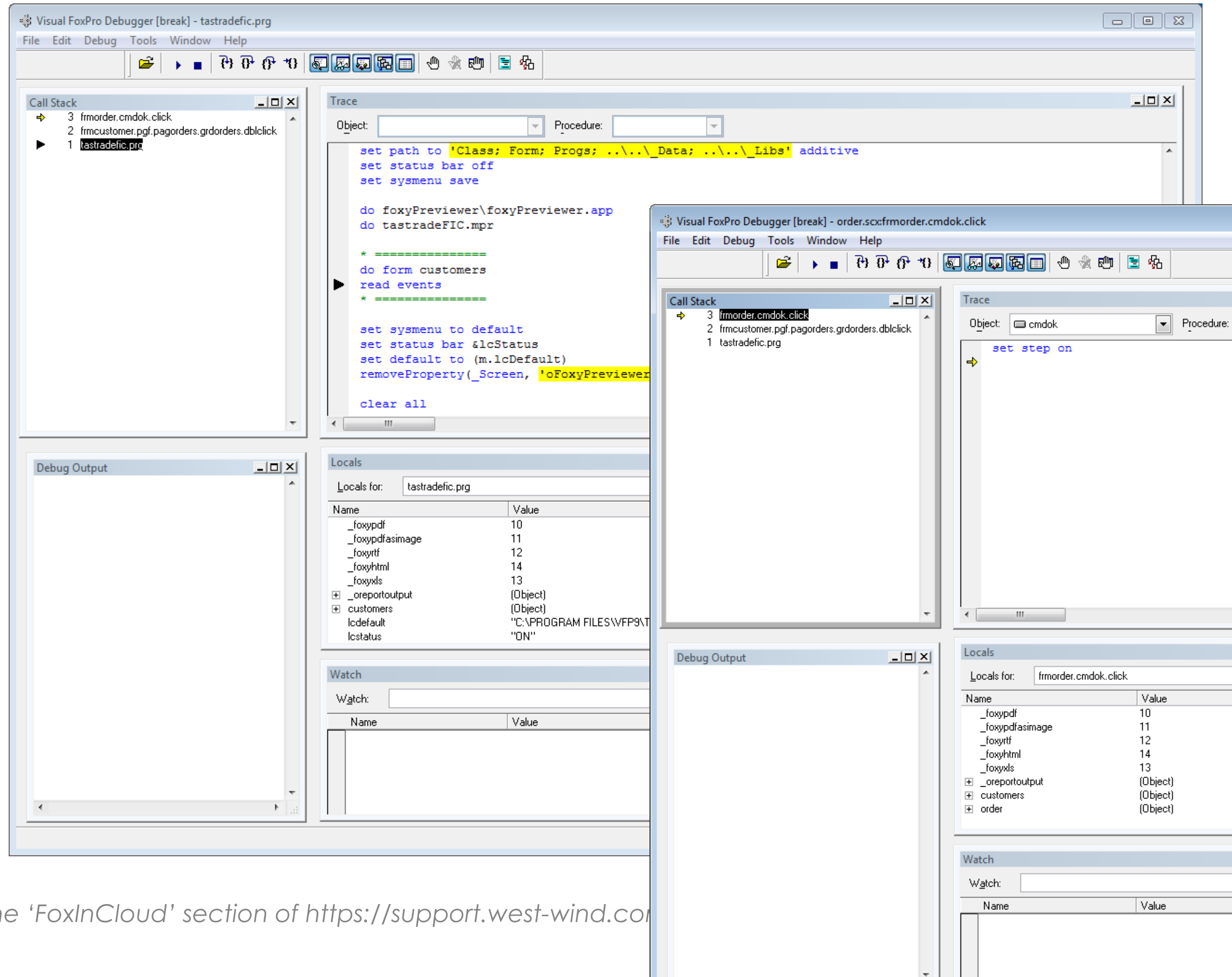
Section	Subject	Duration
1	A different process flow	15’
2	Access to user’s peripherals	3’
3	State factors	3’
4	Where user event are processed	3’
5	dodefault()	3’

Desktop: one execution thread / stack per user

1- User event code runs in the same call stack as the initial application startup.

2- When user closes a form, it releases completely from memory and these events fire: form.unload(), form and members .destroy()

3- Variables assigned at app startup without LOCAL are 'seen' by all called procedures and methods


Visual FoxPro Debugger [break] - tastradefic.prg
File Edit Debug Tools Window Help
Call Stack
Trace
Object:
Procedure:
Debug Output
Locals
Watch

Call Stack:

```
3  frmorder.cmdok.click
2  frmcustomer.pgf.pagorders.grdorders.dblclick
1  tastradefic.prg
```

Trace:

```
set path to 'Class; Form; Progs; ..\..\_Data; ..\..\_Libs' additive
set status bar off
set sysmenu save

do foxyPreviewer\foxyPreviewer.app
do tastradeFIC.mpr

* ===============
do form customers
read events
* ===============

set sysmenu to default
set status bar &lcStatus
set default to (m.lcDefault)
removeProperty(_Screen, 'oFoxyPreviewer

clear all
```

Locals for: tastradefic.prg

Name	Value
_foxypdf	10
_foxypdfasimage	11
_foxyrtf	12
_foxyhtml	14
_foxyxls	13
⊞ _oreportoutput	(Object)
⊞ customers	(Object)
lcdefault	"C:\PROGRAM FILES\VFP9\T
lcstatus	"ON"

Watch:

Name	Value

Visual FoxPro Debugger [break] - order.scx:frmorder.cmdok.click
File Edit Debug Tools Window Help
Call Stack
Trace
Object:
Procedure:
Debug Output
Locals
Watch

Call Stack:

```
3  frmorder.cmdok.click
2  frmcustomer.pgf.pagorders.grdorders.dblclick
1  tastradefic.prg
```

Trace — Object: cmdok

```
set step on
```

Locals for: frmorder.cmdok.click

Name	Value
_foxypdf	10
_foxypdfasimage	11
_foxyrtf	12
_foxyhtml	14
_foxyxls	13
⊞ _oreportoutput	(Object)
⊞ customers	(Object)
⊞ order	(Object)

Watch:

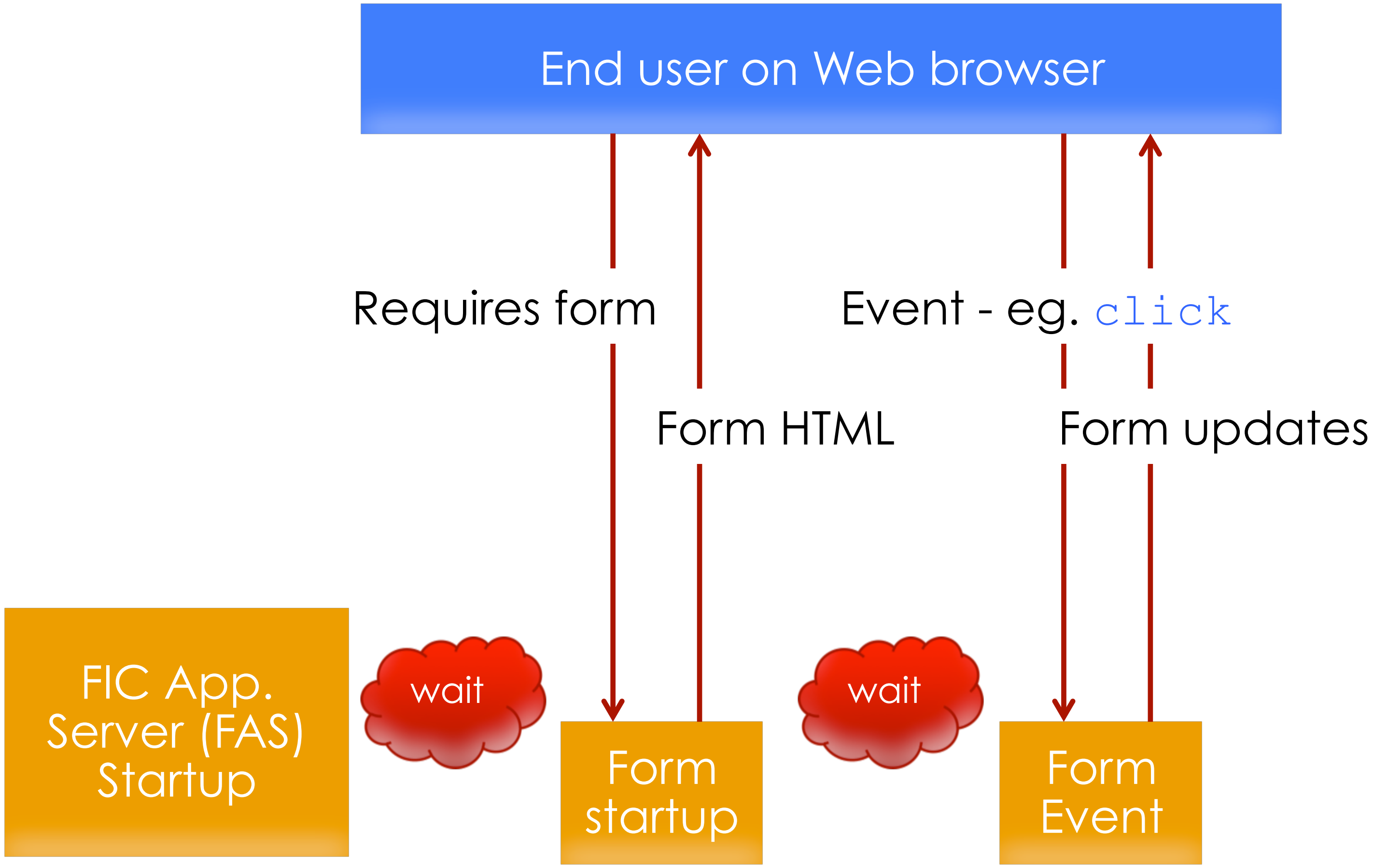
Name	Value

Any question? Post with screenshot in the 'FoxInCloud' section of https://support.west-wind.co

Web: Request – Response

For each request, Server **must deliver a response**

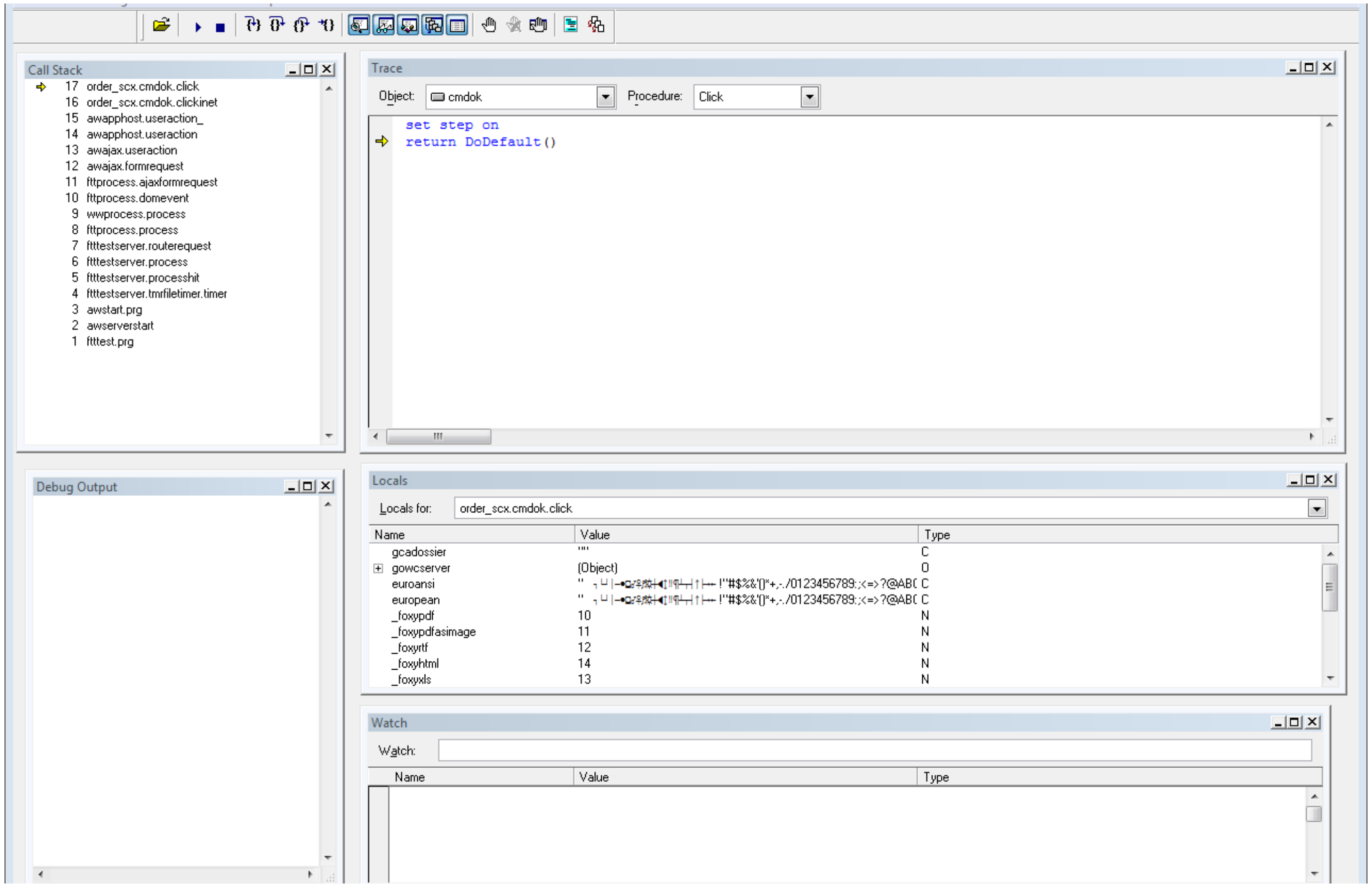
Server can't stay waiting for requests from a single user.



Web: User events run in server context

Each user request runs "Stateless",

IOW, out of the context of startup and previous user action.


Call Stack

```
17 order_scx.cmdok.click
16 order_scx.cmdok.clickinet
15 awapphost.useraction_
14 awapphost.useraction
13 awajax.useraction
12 awajax.formrequest
11 fttprocess.ajaxformrequest
10 fttprocess.domevent
 9 wwprocess.process
 8 fttprocess.process
 7 ftttestserver.routerequest
 6 ftttestserver.process
 5 ftttestserver.processhit
 4 ftttestserver.tmrfiletimer.timer
 3 awstart.prg
 2 awserverstart
 1 ftttest.prg
```

Trace — Object: cmdok — Procedure: Click

```
set step on
return DoDefault()
```

Debug Output

Locals — Locals for: order_scx.cmdok.click

Name	Value	Type
gcadossier	""	C
gowcserver	(Object)	O
euroansi	" ┐└┘│─•◘♂♀♫☼►◄↕‼¶╌┤↑├── !"#$%&'()*+,-./0123456789:;<=>?@AB(	C
european	" ┐└┘│─•◘♂♀♫☼►◄↕‼¶╌┤↑├── !"#$%&'()*+,-./0123456789:;<=>?@AB(	C
_foxypdf	10	N
_foxypdfasimage	11	N
_foxyrtf	12	N
_foxyhtml	14	N
_foxyxls	13	N

Watch — Watch:

Name	Value	Type

Adaptations

- Private variables must either:
  - Become global: PUBLIC, _Screen.Property or _VFP.Property,
  - Be passed to form as parameters (up to 20 supported).
- Minimum dependencies between forms:
  - Private datasession preferred
  - Restore Settings
  - Avoid mutual references
- Callback (…)
- App Startup & Exit (…)

Adaptations (…) Callback

In Web mode, calling form and modal child form run in different threads;

Server can't wait (suspend) until user replies.

Desktop: 1 method

```
LOCAL userChoice, nTimeOut
nTimeOut = 5000 && 5 secs

userChoice = MessageBox(;
  ''; && (eMessageText) MessageBox() parameter # 1
  + 'MessageBox called by "' + m.this.Name + '"' + Chr(13);
  + '(auto closes after ' + Transform(m.nTimeOut/1000) + ' secs)';
 , 3+64+512; && (nDialogBoxType) MessageBox() parameter # 2
 , 'VFP MessageBox()'; && (cTitleBarText) MessageBox() parameter #
 , m.nTimeOut; && (nTimeout) MessageBox() parameter # 4
 )

thisForm.tutoLblInfos.Caption = '';
 + 'MessageBox() returned ';
 + ICase(;
  m.userChoice = IDABORT,; && see FoxPro.h
   'IDABORT',;
  m.userChoice = IDCANCEL,; && see FoxPro.h
   'IDCANCEL',;
  m.userChoice = IDIGNORE,; && see FoxPro.h
   'IDIGNORE',;
  m.userChoice = IDNO,; && see FoxPro.h
   'IDNO',;
  m.userChoice = IDOK,; && see FoxPro.h
   'IDOK',;
  m.userChoice = IDRETRY,; && see FoxPro.h
   'IDRETRY',;
  m.userChoice = IDTIMEOUT,; && see FoxPro.h
   'IDTIMEOUT',;
  m.userChoice = IDYES,; && see FoxPro.h
   'IDYES',;
   '???';
  );
 + ' (as of FoxPro.h)'
```

Web: 2 methods

```
local nTimeOut
nTimeOut = 5000 && 5 secs

thisForm.wMessageBox(;
  'wFormCallBack'; && call back method (in this object)
 , ''; && (eMessageText) MessageBox() parameter # 1
  + 'MessageBox called by "' + m.this.Name + '"' + Chr(13);
  + '(auto closes after ' + Transform(m.nTimeOut/1000) + ' secs)';
 , 3+64+512; && (nDialogBoxType) MessageBox() parameter # 2
 , 'FoxInCloud wMessageBox()'; && (cTitleBarText) MessageBox() parameter # 3
 , m.nTimeOut; && (nTimeout) MessageBox() parameter # 4
 )
ENDPROC


* ______________________________________________________________________________
PROCEDURE wFormCallBack && Standard method for processing value returned by modal forms
LPARAMETERS userChoice && @ User's choice in modal form

&& Source code for processing the value returned by MessageBox()
&& was moved from this.Click() to here

thisForm.tutoLblInfos.Caption = '';
 + 'MessageBox() returned ';
 + ICase(;
  m.userChoice = IDABORT,; && see FoxPro.h
   'IDABORT',;
  m.userChoice = IDCANCEL,; && see FoxPro.h
   'IDCANCEL',;
  m.userChoice = IDIGNORE,; && see FoxPro.h
   'IDIGNORE',;
  m.userChoice = IDNO,; && see FoxPro.h
   'IDNO',;
  m.userChoice = IDOK,; && see FoxPro.h
   'IDOK',;
  m.userChoice = IDRETRY,; && see FoxPro.h
   'IDRETRY',;
  m.userChoice = IDTIMEOUT,; && see FoxPro.h
   'IDTIMEOUT',;
  m.userChoice = IDYES,; && see FoxPro.h
   'IDYES',;
   '???';
  );
 + ' (as of FoxPro.h)'
```

Adaptations (…)
App Startup & Exit

Because of differences in process flow, app startup program should run differently in desktop and web mode

Desktop	Web
`* set up app environment`	OK

`do form login`
`read events`
	KO
	• displaying a form out of a user context makes no sense
	• can't enter a modal state

`* clean up app environment`	KO – no prior modal wait state: this code won't execute

Recommended: move app environment set up code into a standard env. class (`xxxSets as awSets of awPublic.prg`) that cleans up environment automatically

Desktop: main.prg	Web: xxxServer.prg

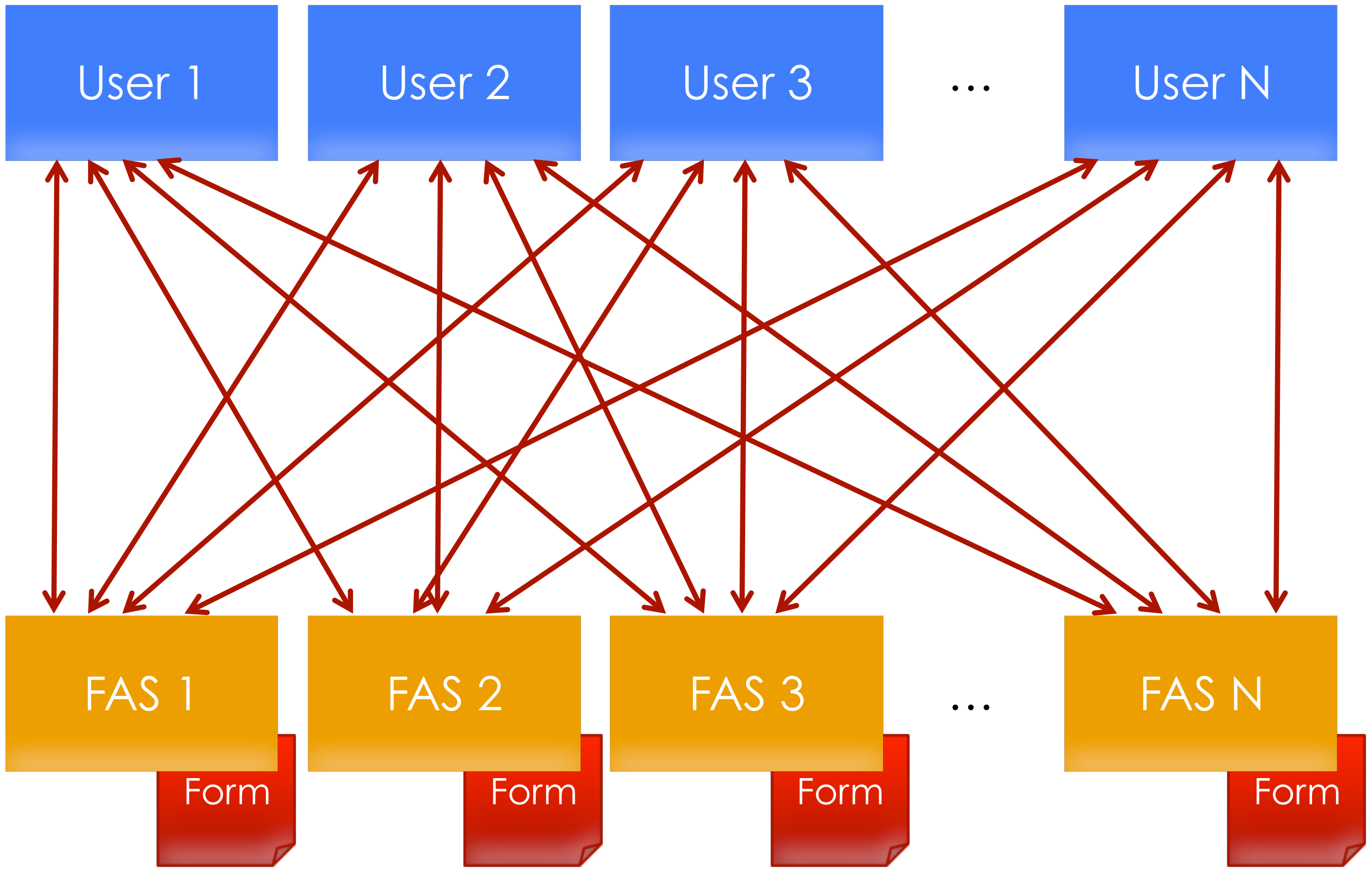
`o = NewObject('xxxSets', 'xxxSets.prg')`
`do form login`
`read events`
	`cAppSetsLib = 'xxxSets.prg'`
	`cAppSets = 'xxxSets'`
	`lAppSetsClass = .T.`

Stateless and Scalable: N Users <–> N Servers

Any user can hit any server at any time.

Each server (FAS) uses a single instance of each form for any user.

FAS saves/ restore the state of form for each user.



Form instances stay alive until server stops

Because each server keeps an instance of each requested form, some events fire slightly differently.

User	Desktop	Web	Adaptation

opens a form	`.Load()`
	`.Init()`
		`.Load()`: once at first user request
		
		`.Init()`:
		• Once at first request with `.wlInitFirst`
		• Once per user
			Move **user-dependent** code from `.Load()` to `.Init()`
			
			Move code within `.Init()`

exits a form	`.Release()`
	`.Destroy()`
	`.Unload()`
		`.Release()` only	Move **user-dependent** code to `.Release()`

Commands acting on user's peripherals

These commands require access to the user's peripherals, which web server can't access.

do form / form.show() / report form
MessageBox() / InputBox() / WAIT
LocFile / PutFile / GetFile / GetDir / GetColor()
Menu commands

To be replaced by procedures and methods supporting either desktop or web:

Original	Adapted
do form/form.show()	[.]wForm*()
report form	PDF generation
MessageBox()	[.]wMessageBox()
WAIT	wWait()
PutFile()	.wFileSaveAs()
Menu commands	wMenu()

Adaptation: Tell FoxInCloud the state factors

To avoid saving properties that never change, FoxInCloud needs to know elements that user can change at runtime.

.wcPropSave

- Each object inherits a 'wcPropSave' property holding a list of properties that user action can change (mostly automated)

.wContentDynamic

- Tells FoxInCloud that the members of a form/ container/page can change at runtime

.wViewParmSet()

- Tells FoxInCloud the name and value of parameters when querying the views

Events: process on server, browser, both, or ignore

In most cases, events are processed on server using the existing event code.

You can also implement this process on the client browser using JavaScript.

- Any user event method must **begin** with this code :

```
if thisForm.wlHTMLgen
  return <some value>
endif
```

where `<some value>` tells FoxInCloud how to process the Event:

- .T.: use existing VFP code on server;
- 'string': JavaScript to be executed in browser; eg. MouseMove();
- `thisForm.wcScriptEventClientServer()` : first execute on browser, then on server;
- .F.: ignore event in web mode.

Add dodefault() to your code

Make sure to let FoxInCloud code run by calling dodefault() where appropriate

(soon automated by FAA)

- `.Init(), .Destroy()`
- `.Load(), .UnLoad()`
- `.Release()`
- `.AddObject(), .NewObject(), .RemoveObject()`
- `.Requery()`
- `.SetFocus()`
- `.*_assign()`
- `.Autofit()`

Look at the code inherited from aw.vcx!aw* to know where to add `dodefault()` in your code.

That’s about it

This presentation has covered the main differences between desktop and web modes, requiring developer’s understanding and attention.

FAA provides an educated, in-depth list of all adaptations you need to care about.

Once your application is adapted and you’ve practiced the adaptation process for a while, you’ll naturally develop ‘the FoxInCloud way’.

One last word about FAA FoxInCloud Adaptation Assistant

- Free: http://foxincloud.com/download.php
- Copy (test) mode / Source mode
- Assistant, not magician!
- Adapts 99% of your code (avg)
- Spots adaptation needing your attention
- Provides guidance, documentation and code samples, together with FoxInCloud Live Tutorial: http://foxincloud.com/tutotest/
- Helps you manage your adaptation project.