FoxInCloud

A-Z tutorial

Prepared for the 20TH VFP DevCon Germany, 11/2013

Requires FAA & FAS V2.0+

Agenda

Session	Subject	Slides
1	Understanding FoxInCloud	42
2	Adapting your Application to FoxInCloud	43
3	Publishing your Application on localhost	42
4	Deploying your Application for production	36

Each session breaks down into episodes

Prerequisites

Download and install:

- FoxInCloud Adaptation Assistant (FAA)

- FoxInCloud Application Server (FAS)

Both FAA and FAS are available for free at:
  http://foxincloud.com/download.php

FoxInCloud

A-Z tutorial, Session 1

Requires FAA & FAS V2.0+

Session 1
Understanding FoxInCloud

This session targets users who wish to see the big picture before starting hands-on experience…
it describes how FoxInCloud Application Server (FAS) operates your application on a Web server and interacts with the client browser.

Reproducing steps of this session requires setting break points at various locations of FAS source code (licensed users only).

Though you may follow the steps by browsing the coming slides and applying the steps on your own machine, you may get a better understanding by attending a FoxInCloud workshop or training session; please feel free to mail contact@foxincloud.com *for workshops and training sessions to come.*

FAS installs the files used in this session into `<VFP9>\Tools\AB\AW\Samples\FIC\FICtuto\`

Understanding FoxInCloud, Agenda

Episode	Subject	Slides
	FoxInCloud Architecture	1
1	Connecting Web Server and FAS	11
2	Displaying app's main form in Browser	17
3	Interacting with a form in Browser	12
		41

FoxInCloud Architecture

Web server (**IIS**, Apache, …):
Hosts Web site (i.e.
www.mysite.com)

FIC App. Server (FAS):
VFP/wConnect-based
FoxInCloud Application Server

App Host:
Private session where adapted
application runs

HTML/CSS/JS gen:
clones your app's forms into
HTML/CSS/JS which becomes
the designated website (i.e.
www.mysite.com)

Process:
Processes HTTP requests
received from the site

AJAX:
Processes user actions on forms
thru AJAX requests

Web Server → FIC App. Server — App Host

FIC App. Server → Process — AJAX

Process → HTML / CSS / JS on Web site ← AJAX

HTML / CSS / JS on Web site → End user on Web site → Web Server

Any question? Post with screenshot in the 'FoxInCloud' section of http://west-wind.com/wwThreads/ or http://www.universalthread.com/
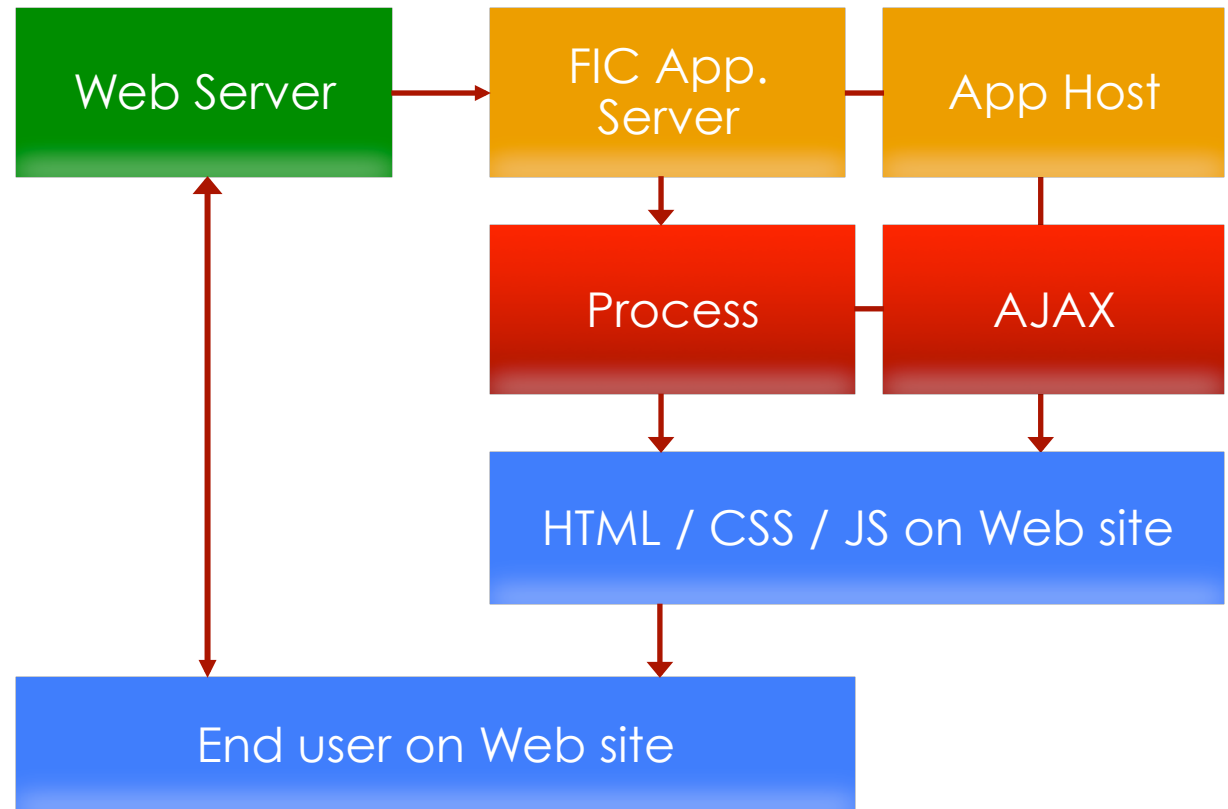
Session 1, episode 1
Connecting Web server to FAS
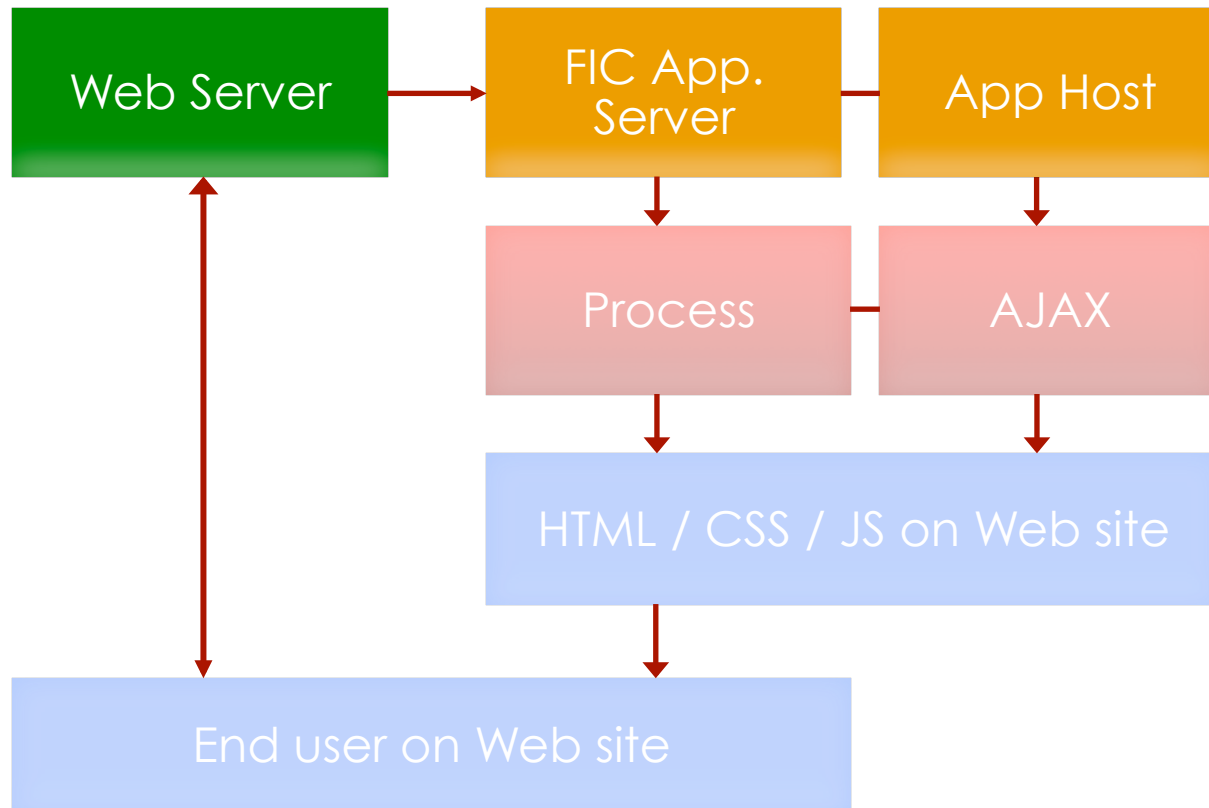
Grabbing requests from localhost web server

IIS comes with most Windows editions

Note: the free FoxInCloud Adaptation Assistant (FAA) automates all IIS settings to come;

Session 3 demonstrates this automated installation.
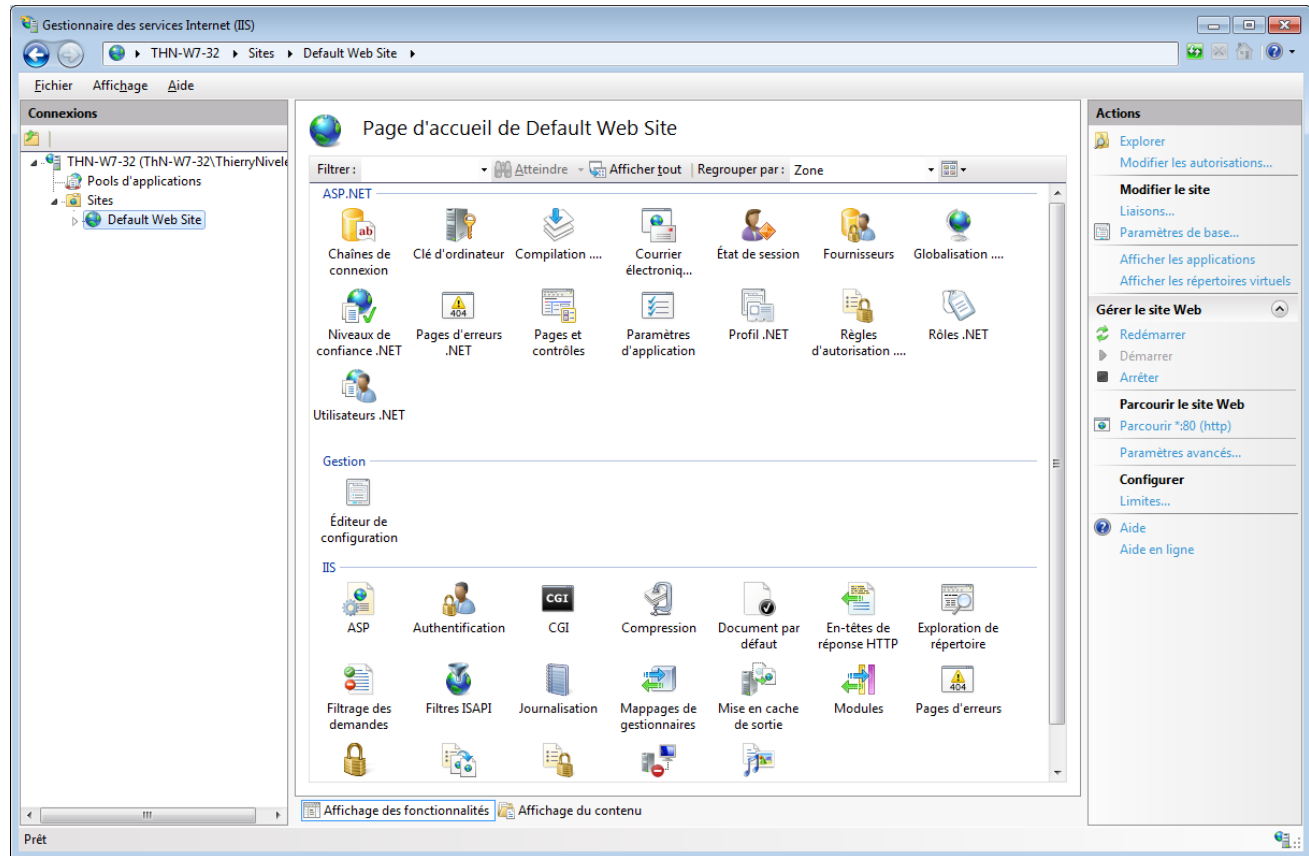
Adding an IIS application

An "IIS application" is a virtual directory where an application can run
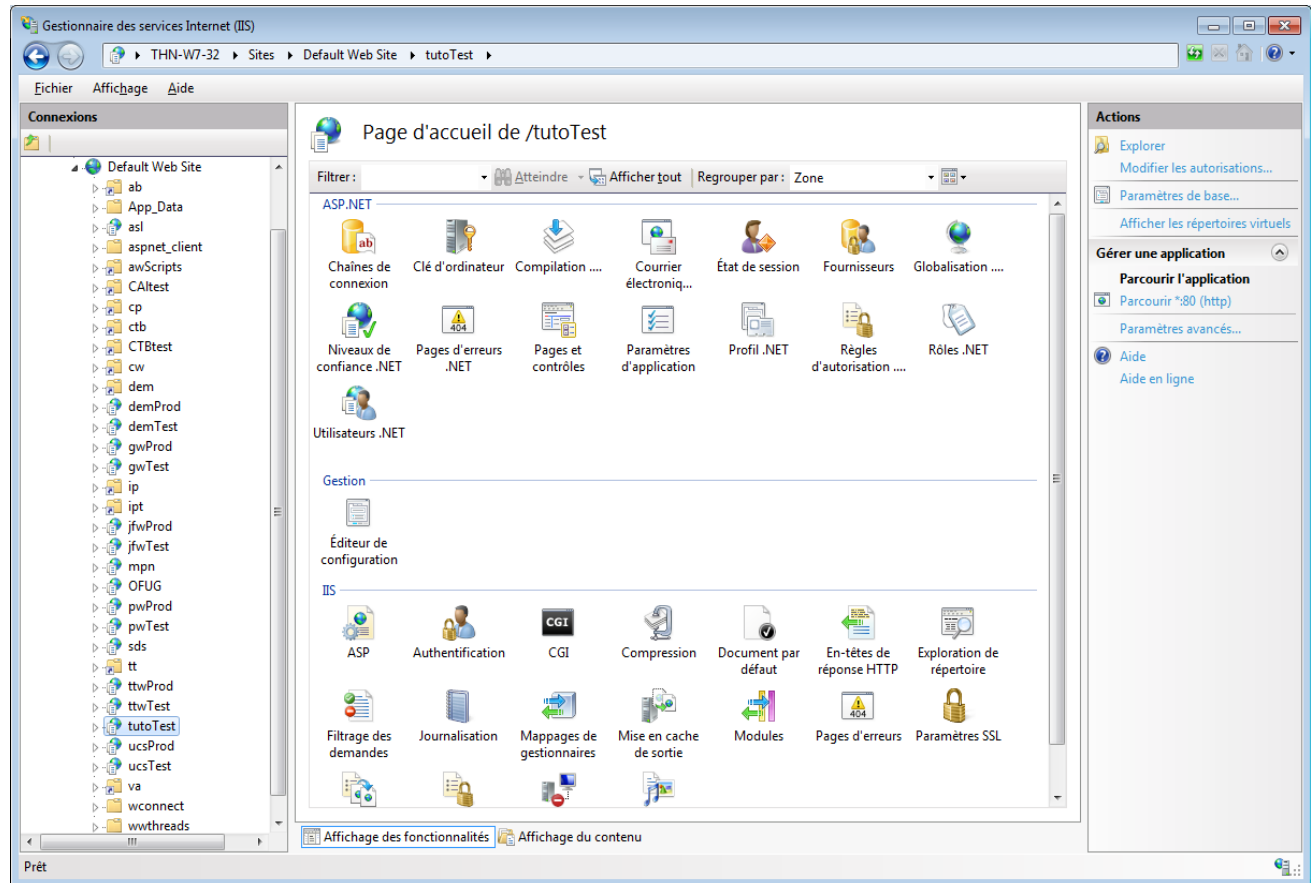
Mapping an URL extension to your FAS

IIS allows URLs with a given extension to be processed by a given module (.exe or .dll)

Target module is wConnect's "wc.dll"

FAS relies on "West-Wind web connect" (wConnect)

`wc.dll` is the IIS extension that wConnect provides.
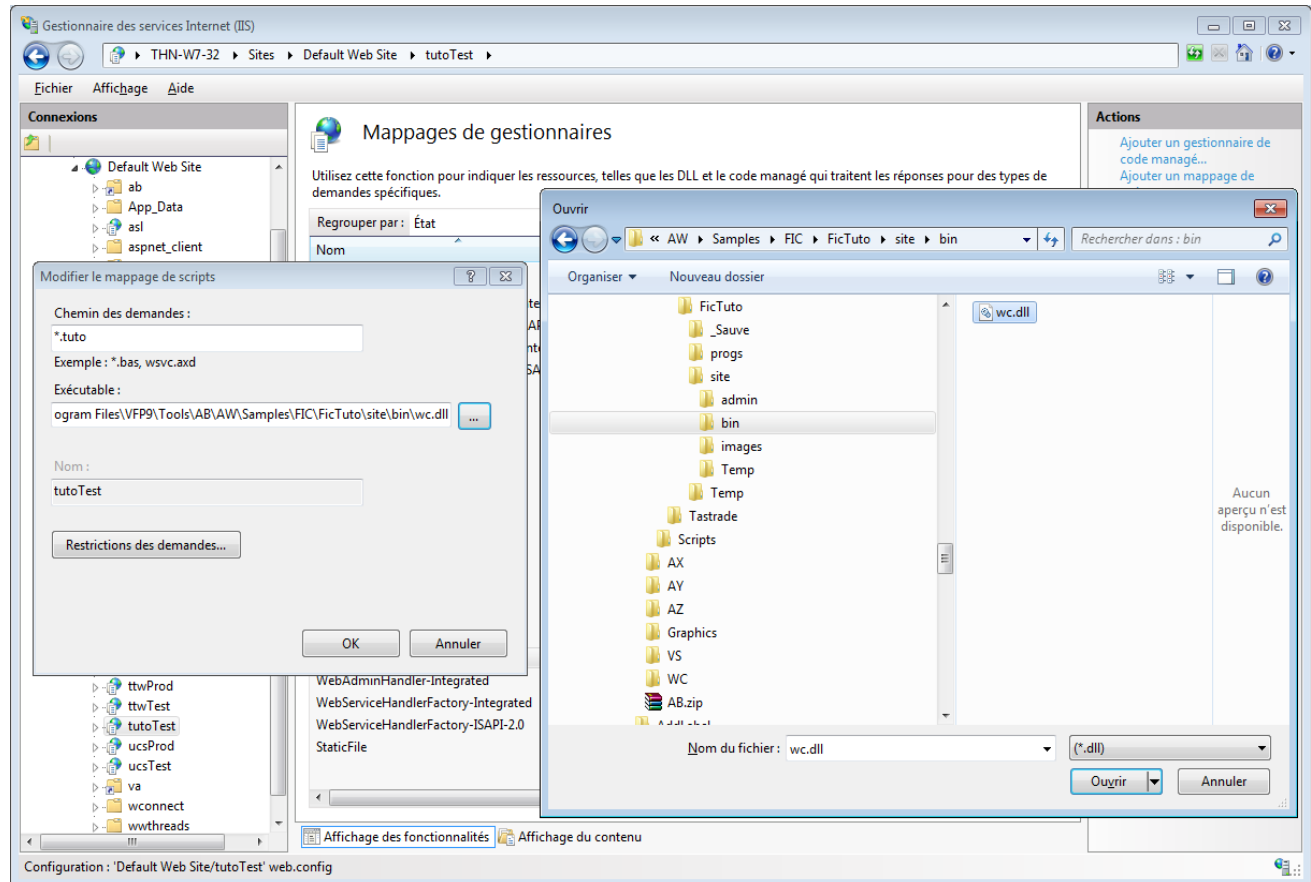
wc.dll reads settings from wc.ini

Main setting is the folder where wc.dll and FAS trade request and response files

When getting an HTTP request from IIS, wc.dll writes request details into a file in this folder.

C:\Program Files\VFP9\Tools\AB\AW\Samples\FIC\FicTuto\site\bin\wc.ini - Sublime Text 2 (UNREGISTERED)

File   Edit   Selection   Find   View   Goto   Tools   Project   Preferences   Help

wc.ini

```
1   [wwcgi]
2   ; Directory where wc.dll and FAS trades files
3   Path=c:\Temp\FICtuto\
4
5   ;*** Time to allow request to finish
6   ;*** Process will be terminated after number of secs
7   ;*** specified here.           REQUIRED
8   Timeout=600
9
10  ;*** Specify how often wc.dll polls for 'completion' message
11  ;*** Specify in milliseconds.  REQUIRED FOR FILE BASED
12  PollTime=100
13
14  ;*** Message File Template (1st 3 letters)
15  ;*** Default is "wc_"
16  ;*** Only needed if using a different template
17  Template=wc_
18
19  ;*** Messaging Mechanism  of the DLL:    REQUIRED
20  ;*** File       -   Original WEB Connection Logic of
21  ;                   file based messaging
22  ;*** Automation  -  Use OLE Automation Server Interface
23  ;*** Interactive -  Call up the VFP development environment via Automation
24  Mechanism=File
25  ;Mechanism=Automation
26
27  ; PostMethod   -  URLEncoded or INI  -  Match POSTMETHOD in WCONNECT.H
28  PostMethod=UrlEncoded
29
30  ;*** Account for Admin tasks    REQUIRED FOR ADMIN TASKS
31  ;***      NT User Account   -  The specified user must log in
32  ;***      Any               -  Any logged in user
33  ;***                        -  Blank - no Authentication
34  AdminAccount=
```

3 lines, 68 characters selected   Tab Size: 4   Plain Text

Starting FoxInCloud Application Server (FAS)…

Open

`ficTuto.pjx`
and run
main
program
`tutoTest.prg`
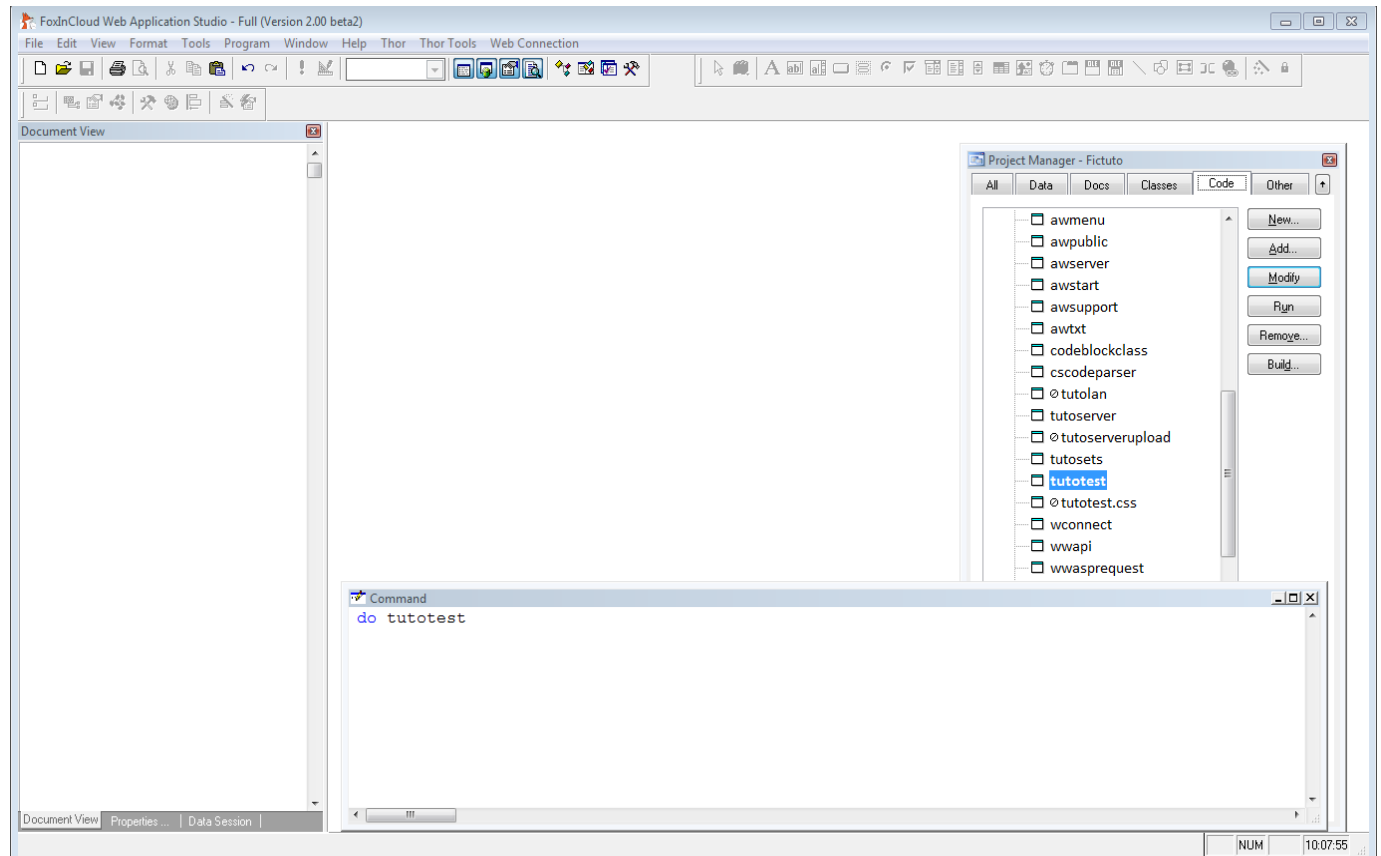
FAS initializes application

awAppHost is app's session object.

Both desktop and Web versions of your app share the xxxSets Settings class.

FAS enters wait state

FAS waits for timer events

FAS timer fires

In dev. mode, FAS waits for files written by the Web server, containing HTTP request details

FAS: ready to serve requests!
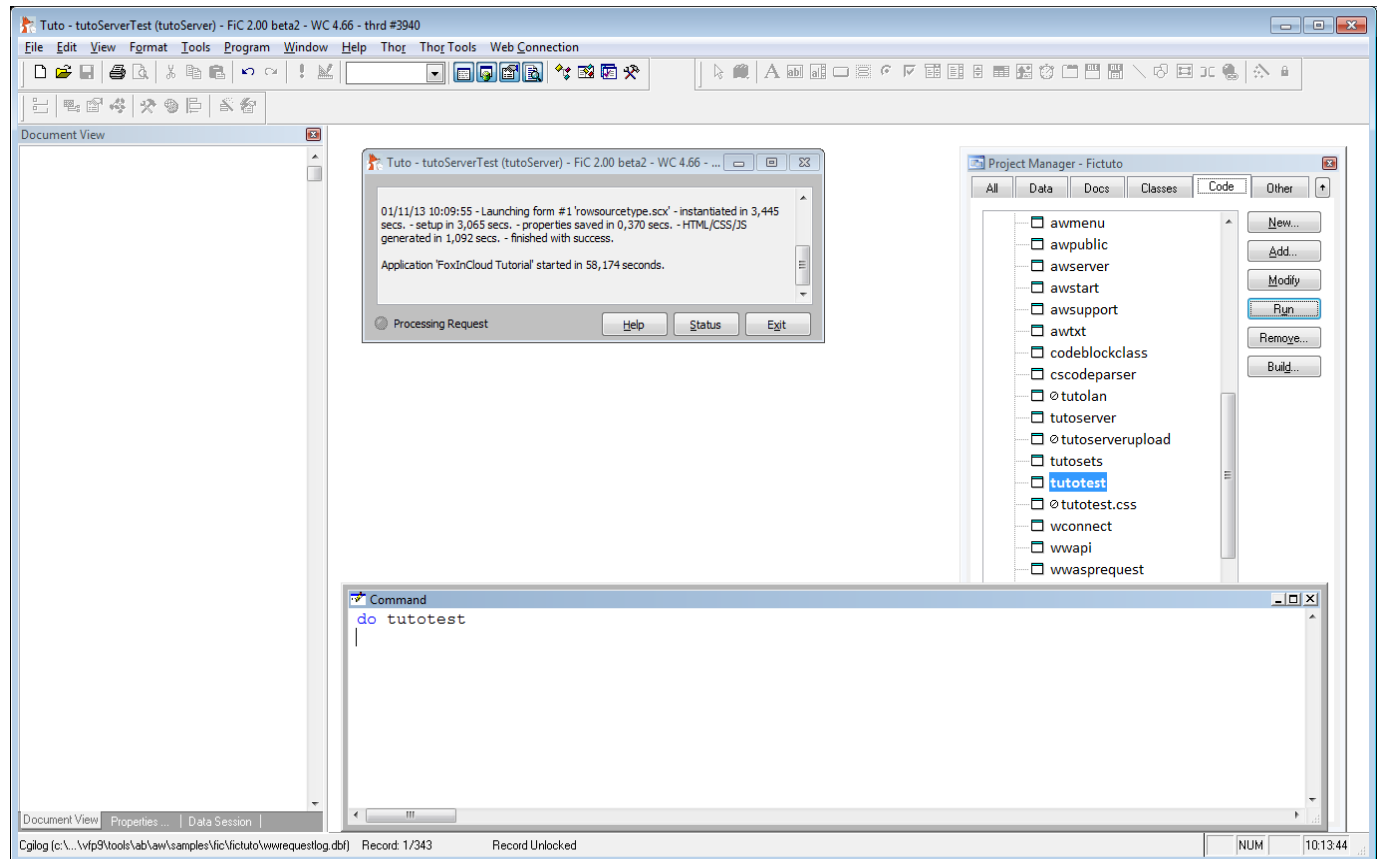
Session 1, episode 2
Displaying main form in browser

```
Web Server  →  Fic App.     →  App Host
                Server
                  ↓               ↓
                Process   —     AJAX
                  ↓               ↓
                HTML / CSS / JS on Web site
                       ↓
                End user on Web site
```
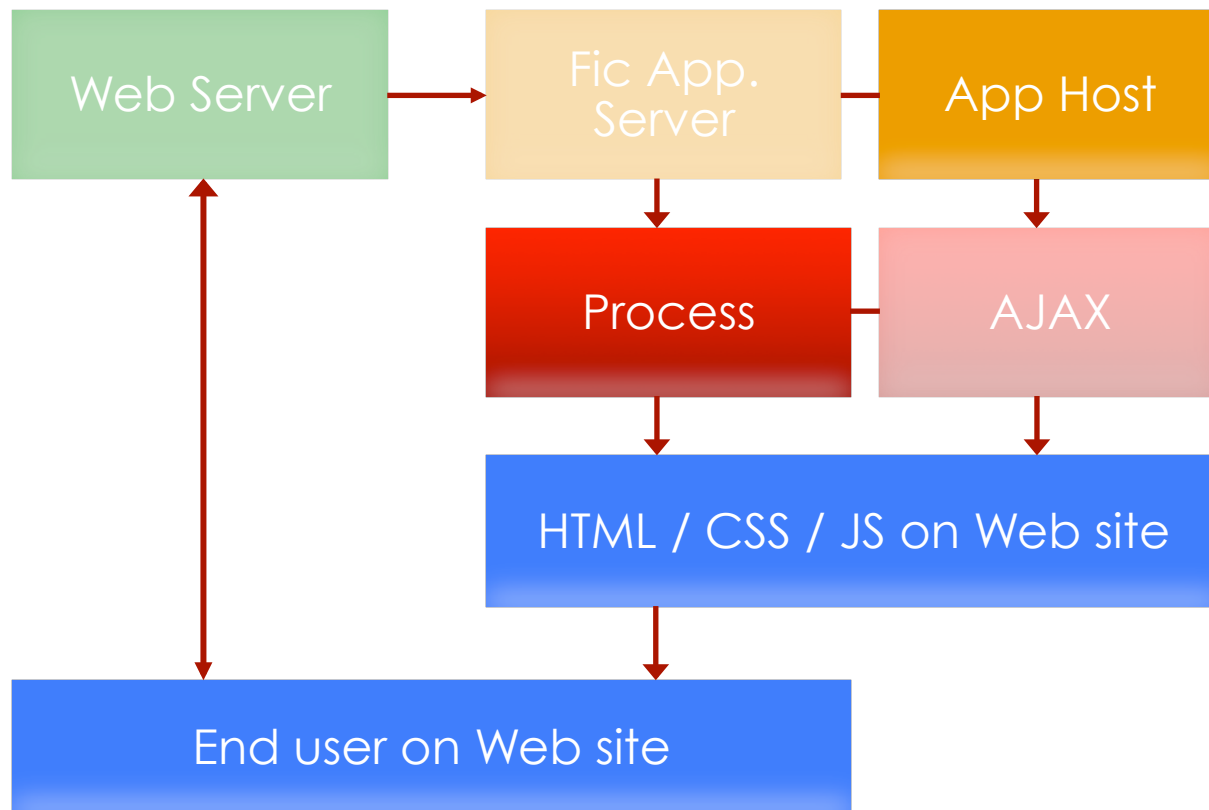
The form we want to display in the browser

`index.scx` is a regular VFP form with several command buttons and a background picture ...
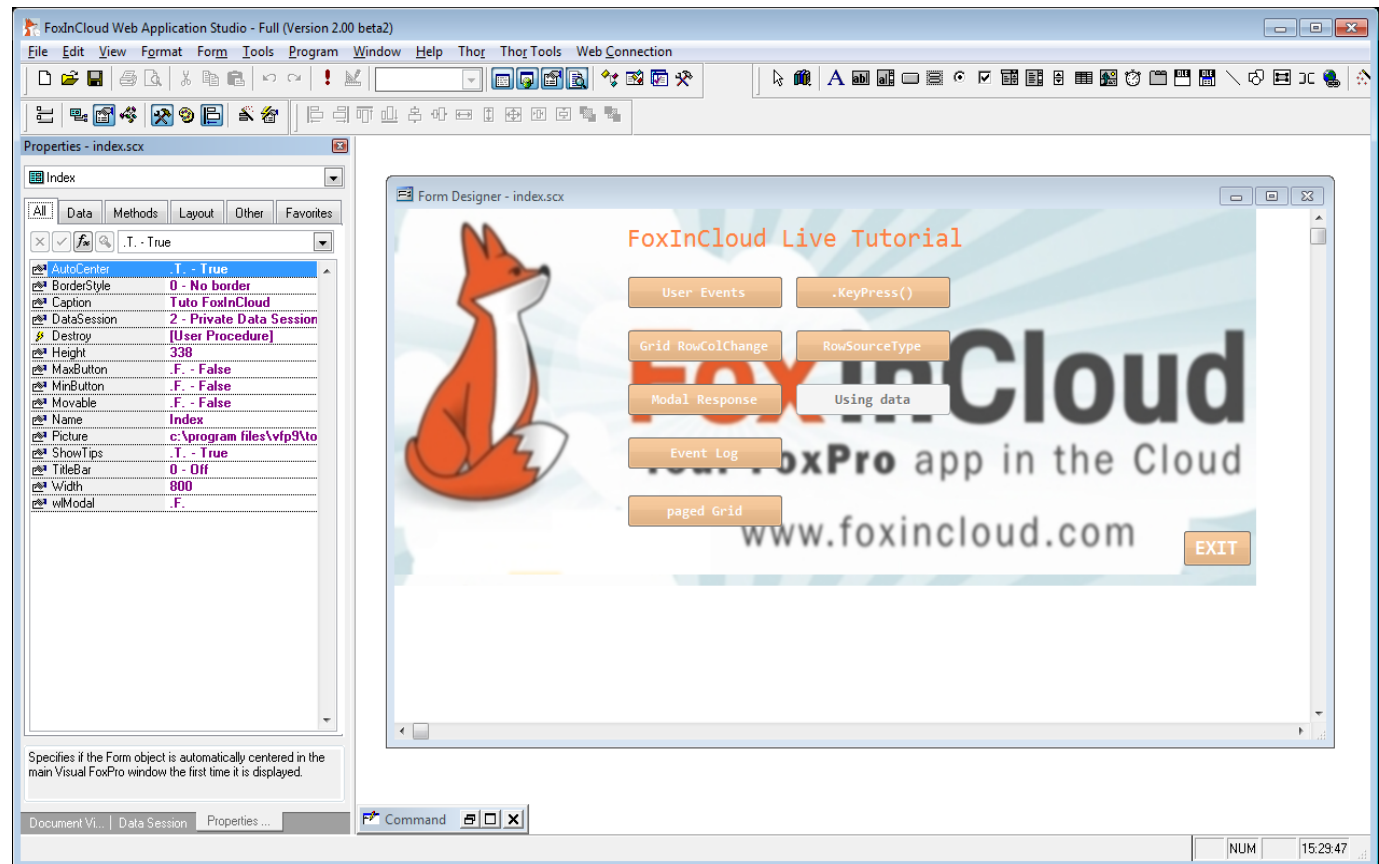
Redirecting IIS default document to FAS home

We redirect IIS's default document `index.htm` to an URL with the extension mapped to our FAS: `index.tuto`
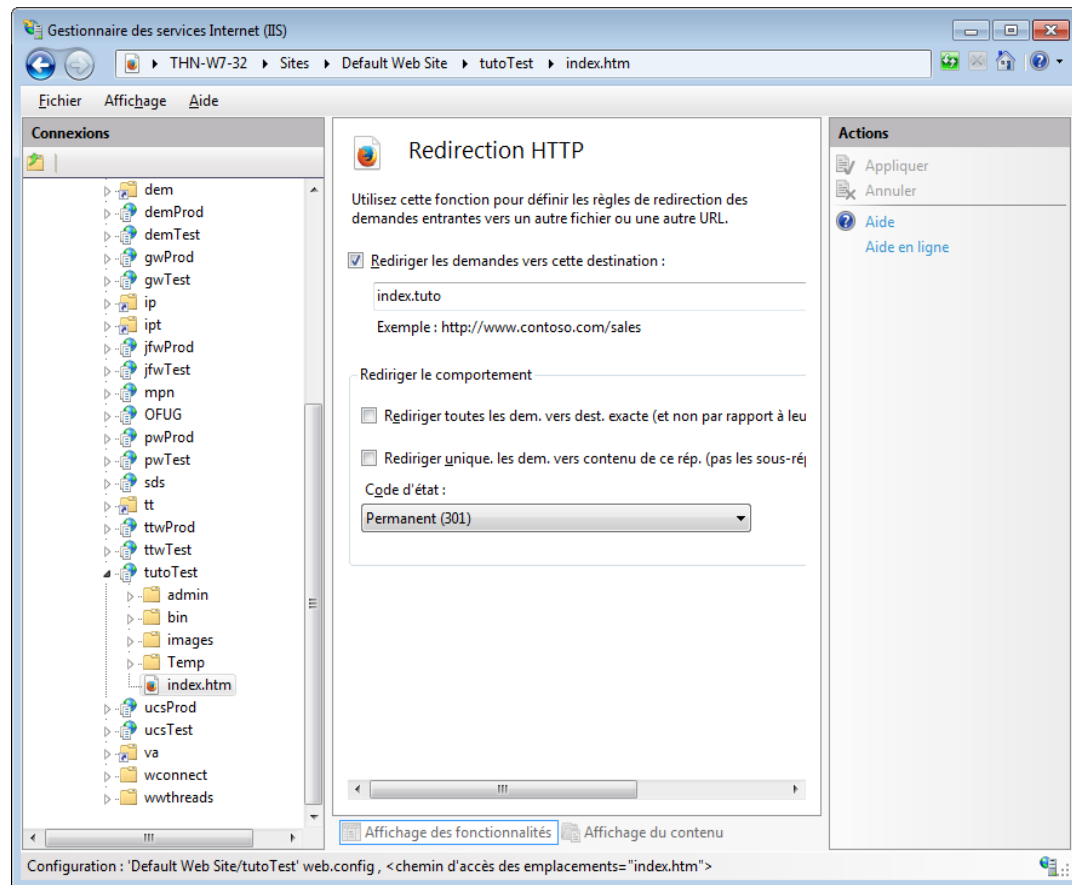
Typing application URL
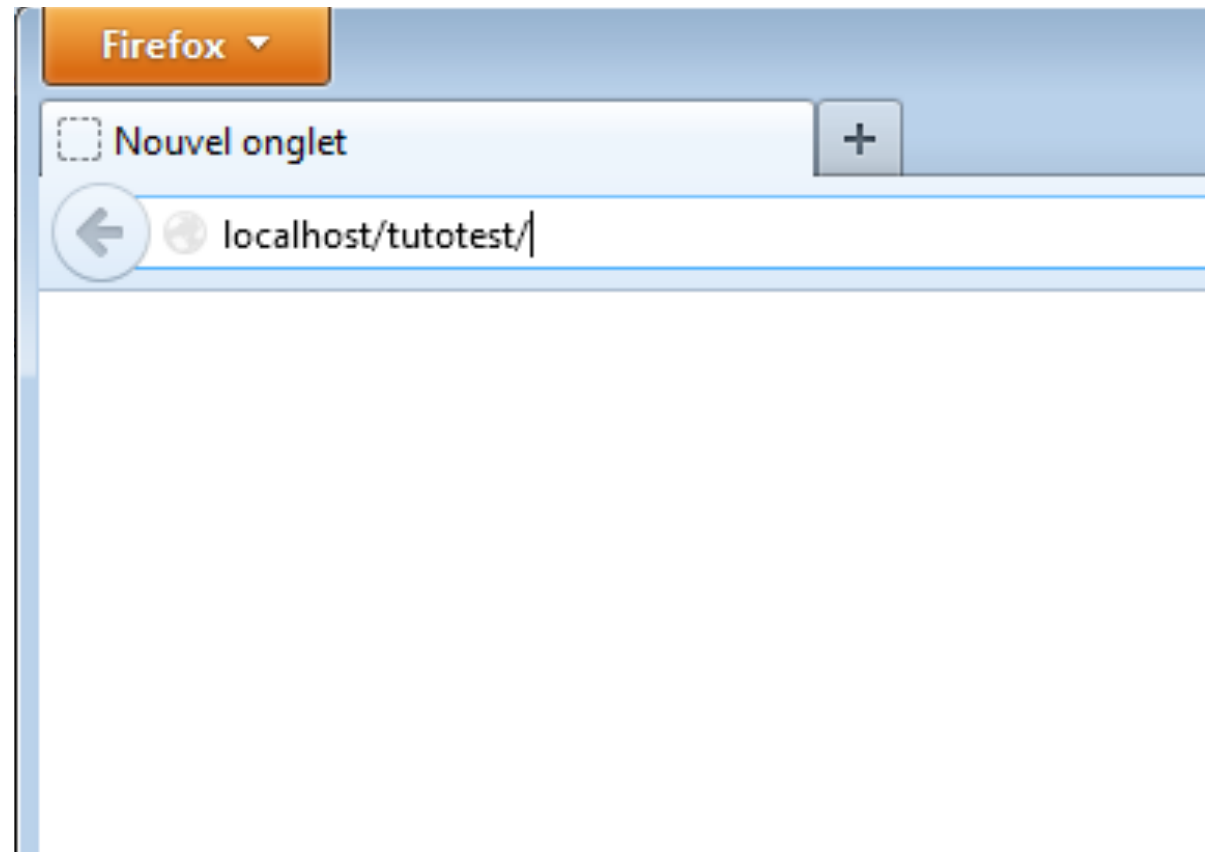
IIS redirects
root URL to our FAS

Due to IIS redirection of `index.htm` to `index.tuto`, typing application root URL `http://localhost/tutotest/` yields application home page
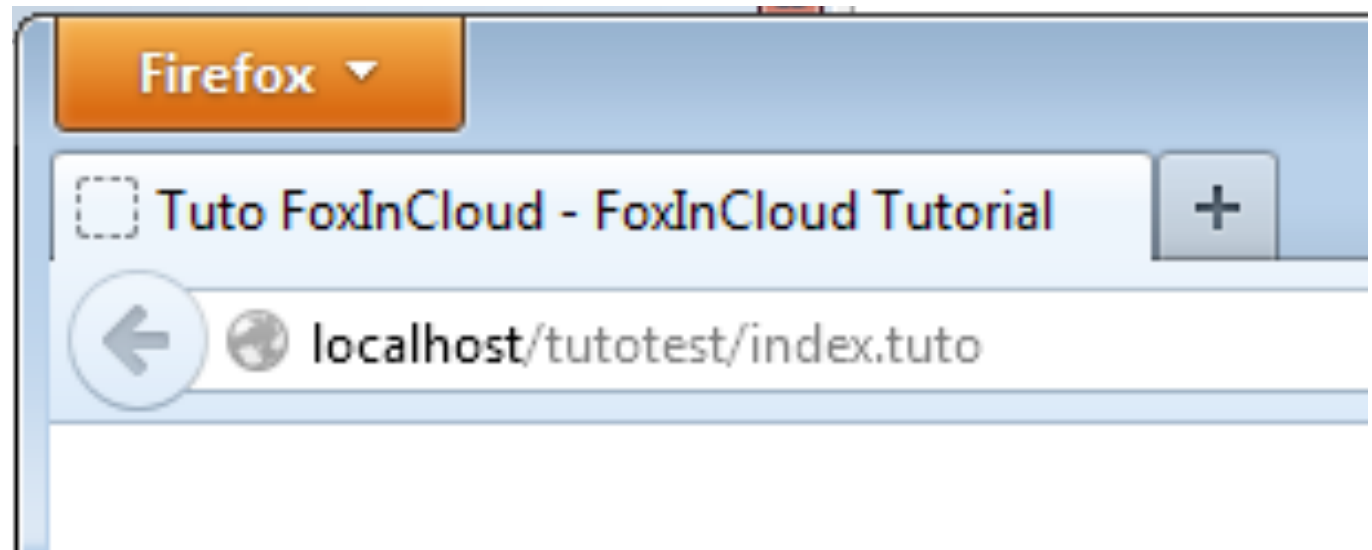
wc.dll gets the request from IIS and dumps it into a file

As we mapped `*.tuto` to `wc.dll`, IIS transmits the `…/index.tuto` request to `wc.dll`

Temp.ini

```
1  &&Output+File=c:\temp\fictuto\WWCDBA3.tmp&DLLVersion=Web+Connection+4%2E67+%2832+servers%29&wcCo
   nfig=C%3A%5CProgram+Files%5CVFP9%5CTools%5CAB%5CAW%5CSamples%5CFIC%5CFicTuto%5Csite%5Cbin%5Cwc%2
   EINI&REQUEST_METHOD=POST&QUERY_STRING=%26MethAddr%3DInitControlsInet%26ObjAddr%3Dindex%5Fscx%26P
   ageInit%3Dtrue%26UserID%3D%26nReq%3D1&Executable+Path=%2Ftutotest%2FMethExec%2Etuto&CONTENT_TYPE
   =application%2Fx%2Dwww%2Dform%2Durlencoded%3B+charset%3DUTF%2D8&CONTENT_LENGTH=19&PATH_INFO=%2Ft
   utotest%2FMethExec%2Etuto&PATH_TRANSLATED=C%3A%5CProgram+Files%5CVFP9%5CTools%5CAB%5CAW%5CSample
   s%5CFIC%5CFicTuto%5Csite%5CMethExec%2Etuto&SCRIPT_NAME=%2Ftutotest%2FMethExec%2Etuto&PHYSICAL_PA
   TH=C%3A%5CProgram+Files%5CVFP9%5CTools%5CAB%5CAW%5CSamples%5CFIC%5CFicTuto%5Csite%5CMethExec%2Et
   uto&SERVER_PROTOCOL=HTTP%2F1%2E1&SERVER_SOFTWARE=Microsoft%2DIIS%2F7%2E5&SERVER_NAME=localhost&S
   ERVER_PORT=80&REMOTE_HOST=%3A%3A1&REMOTE_ADDR=%3A%3A1&HTTP_REFERER=http%3A%2F%2Flocalhost%2Ftuto
   test%2Findex%2Etuto&HTTP_USER_AGENT=Mozilla%2F5%2E0+%28Windows+NT+6%2E1%3B+rv%3A24%2E0%29+Gecko%
   2F20100101+Firefox%2F24%2E0&HTTP_COOKIE=%5F%5Futma%3D111872281%2E777200636%2E1310508010%2E137783
   1031%2E1377947599%2E174%3B+%5F%5Futmz%3D111872281%2E1310508010%2E1%2E1%2Eutmcsr%3D%28direct%29%7
   Cutmccn%3D%28direct%29%7Cutmcmd%3D%28none%29%3B+ip%3D3970QTW9T%3B+pus%2Didv%3DA0039d7dea3096fc47
   1da81b55cbb68e9d69%3B+mpn%3D3NU0TNRAV%3B+tt%3D37F18TQ3Y%3B+ebNewBandWidth%5F%2Elocalhost%3D2247%
   253A1365827628263%3B+tuto%3D3VT1AYS89%3B+%5F%5Futmc%3D111872281%3B+ASPSESSIONIDSSSQRBBQ%3DPFBBJJ
   EBBAFHNGOCGIKAOEOC&APPL_PHYSICAL_PATH=C%3A%5CProgram+Files%5CVFP9%5CTools%5CAB%5CAW%5CSamples%5C
   FIC%5CFicTuto%5Csite%5C&GMT_OFFSET=3600&ALL_HTTP=HTTP_CACHE_CONTROL:no-cache
2  HTTP_CONNECTION:keep-alive
3  HTTP_PRAGMA:no-cache
4  HTTP_CONTENT_LENGTH:19
5  HTTP_CONTENT_TYPE:application/x-www-form-urlencoded; charset=UTF-8
6  HTTP_ACCEPT:text/javascript, text/html, application/xml, text/xml, */*
7  HTTP_ACCEPT_ENCODING:gzip, deflate
8  HTTP_ACCEPT_LANGUAGE:fr,fr-fr;q=0.8,en-us;q=0.5,en;q=0.3
9  HTTP_COOKIE:__utma=111872281.777200636.1310508010.1377831031.1377947599.174; __utmz=111872281.
   1310508010.1.1.utmcsr=(direct)|utmccn=(direct)|utmcmd=(none); ip=3970QTW9T; pus-
   idv=A0039d7dea3096fc471da81b55cbb68e9d69; mpn=3NU0TNRAV; tt=37F18TQ3Y; ebNewBandWidth_.
   localhost=2247%3A1365827628263; tuto=3VT1AYS89; __utmc=111872281;
   ASPSESSIONIDSSSQRBBQ=PFBBJJEBBAFHNGOCGIKAOEOC
10 HTTP_HOST:localhost
11 HTTP_REFERER:http://localhost/tutotest/index.tuto
12 HTTP_USER_AGENT:Mozilla/5.0 (Windows NT 6.1; rv:24.0) Gecko/20100101 Firefox/24.0
```

Timer fires,
FAS reads request from file

As soon as FAS timer fires and finds a request file in the folder shared with wc.dll, FAS timer reads its content.

(Default timer interval is 200 ms.)

FAS routes request to application's Process object

Inside FAS, the `.tuto` extension maps to the `tutoProcess` class.

This class processes the request.

Application executes process class method

The process class executes the method named after justStem(url)

("index.tuto" executes the method 'index' of process class mapped to '.tuto')

All FoxInCloud member names begin with 'w'
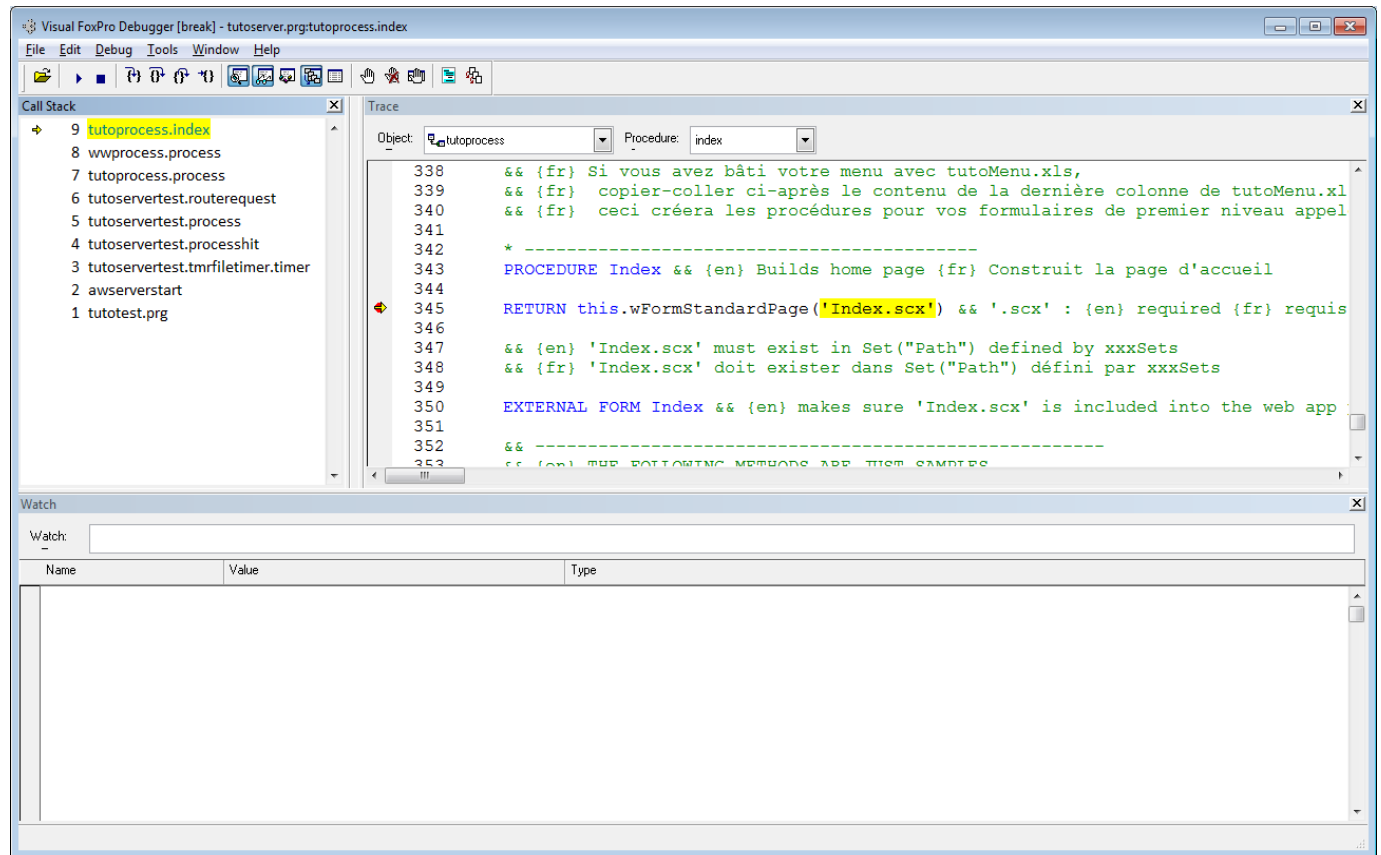
Process asks application host for a reference to the form

oAppHost is FoxInCloud application host object.

Application host object keeps a pool of references to the various forms in the application

First time this form is used: App Host must instantiate

App. Host does not hold any reference to the form requested:

This form needs to be instantiated

App Host stores a reference to the form

App. Host instantiates the form using a regular DO FORM...NAME command.

This form instance lives inside App Host as long as server runs.

All users share this instance.

Form.events() execute

Form instantiates and fires events just like in desktop:

`.Load(),`

`.Init(),`

`.Show(), ...`

FAS saves form's initial state into a VFP table

As any Web application, FoxInCloud needs save application and forms state for each user, including dataSession;

Initial state is restored whenever a user opens a form.

FAS clones form into HTML, CSS and JavaScript

FoxInCloud's **awHTMLgen** class walks through the form's containers hierarchy to generate HTML CSS JS replicating the layout and behavior of original VFP form

FAS send back a full HTML page including the form

FoxInCloud's `.wFormStandardPage()` method builds a full HTML page including `<head>`, FAS standard and application-specific CSS & JS, `<body>`, etc.

Form page finally shows up in the browser …

The whole process, from FAS receiving the request to response delivered **takes less than a second** …

index.tuto – Launching form #1 'index.scx'…
Instantiated *in .124 secs.*
Setup *in .006 secs.*
State *saved in .134 secs.*
HTML/CSS/JS *generated in .334 secs.*
Finished *with success in .914 secs*

Firefox ▾

Tuto FoxInCloud - FoxInCloud Tutorial   +

localhost/tutotest/index.tuto   Google

FoxInCloud Live Tutorial

User Events   .KeyPress()

Grid RowColChange   RowSourceType

Modal Response   Using data

Event Log

paged Grid

FoxInCloud
YourFoxPro app in the Cloud
www.foxincloud.com

Session 1, episode 3
Interacting with a form

```
┌─────────────┐      ┌─────────────┐   ┌─────────────┐
│             │      │  Fic App.   │   │             │
│ Web Server  │─────▶│  Server     │───│  App Host   │
│             │      │             │   │             │
└─────────────┘      └─────────────┘   └─────────────┘
       ▲                    │                 │
       │                    ▼                 ▼
       │             ┌─────────────┐   ┌─────────────┐
       │             │             │   │             │
       │             │  Process    │───│    AJAX     │
       │             │             │   │             │
       │             └─────────────┘   └─────────────┘
       │                    │                 │
       │                    ▼                 ▼
       │             ┌───────────────────────────────┐
       │             │  HTML / CSS / JS on Web site   │
       │             └───────────────────────────────┘
       │                          │
       │                          ▼
┌─────────────────────────────────────────────┐
│          End user on Web site                │
└─────────────────────────────────────────────┘
```
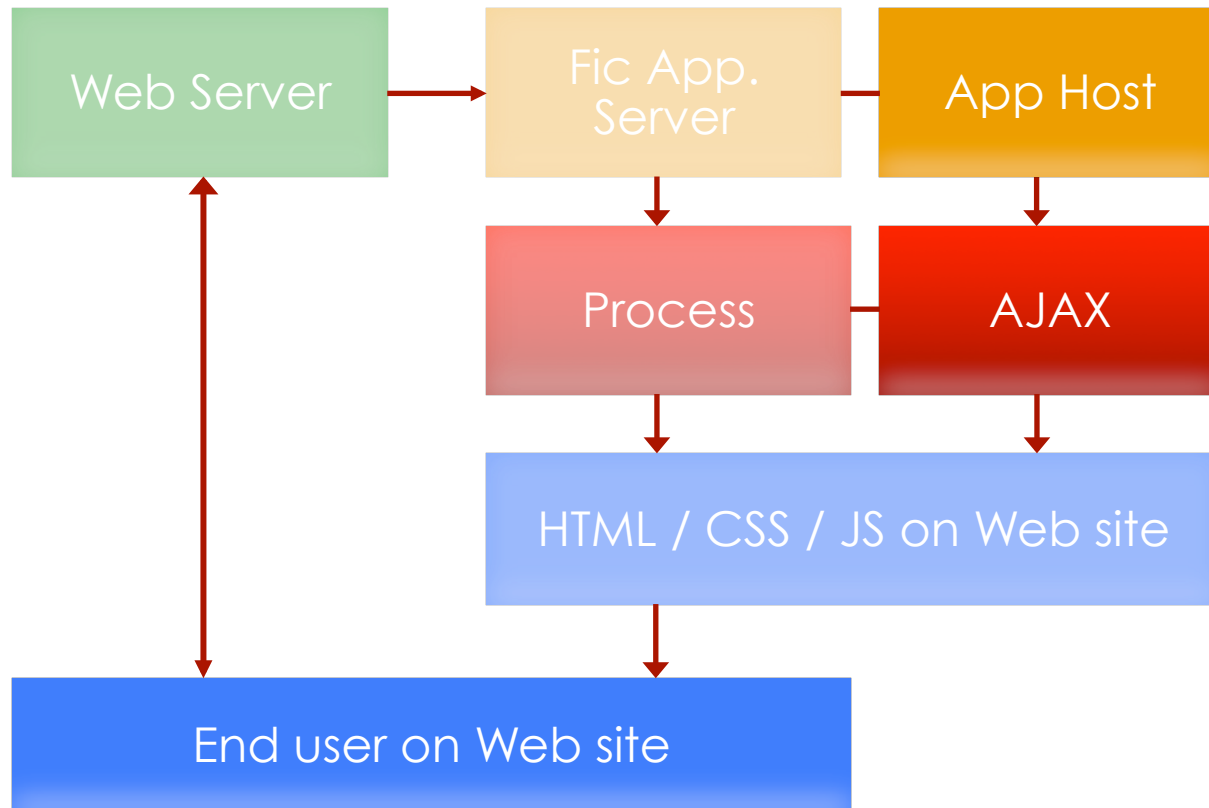
Desktop: clicking a form button opens a child form...

In the desktop application, clicking the `.rowSourceType` command button opens the `rowSourceType.scx` form

Let's click the same button in web browser …

FAS has generated the same button in HTML

FoxInCloud has generated a JS .click() event handler

Because the `.click()` event of this button is implemented in the original VFP form, FAS has generated a similar event handler in form's JavaScript named `index_scx.js`


```
     localhost/tutotest/index.tuto                                    Google

                    FoxInCloud Live Tutorial

                 User Events              .KeyPress()

               Grid RowColChange        RowSourceType

                Modal Response            Using data

                  Event Log

                  paged Grid
                                           www.foxincloud.com
```

```
Console  HTML  CSS  Script ▾  DOM  Réseau  Cookies         Use #<number> to go to lin

Tous ·  index_scx.js ·
   4  $('index_scx-tutocmdmodal').observe('click', FoxInCloud.DOMEvent.bind(FoxInCloud));
   5  $('index_scx-tutocmdrowcolchange').observe('click', FoxInCloud.DOMEvent.bind(FoxInCloud));
   6  $('index_scx-tutolanrelease').observe('click', FoxInCloud.DOMEvent.bind(FoxInCloud));
   7  $('index_scx-tutolanrelease').disabled=true;
   8  $('index_scx-tutocmdlogevent').observe('click', FoxInCloud.DOMEvent.bind(FoxInCloud));
   9  $('index_scx-tutocmddata').observe('click', FoxInCloud.DOMEvent.bind(FoxInCloud));
  10  $('index_scx-tutocmddata').disabled=true;
  11  $('index_scx-tutorowsourcetype').observe('click', FoxInCloud.DOMEvent.bind(FoxInCloud));
  12  $('index_scx-tutocmdpagedgrid').observe('click', FoxInCloud.DOMEvent.bind(FoxInCloud));
```

Clicking this button triggers an AJAX request to FAS

All user interaction events are processed by standard `FoxInCloud.js.DOMEvent` method, resulting in sending an AJAX request to the FAS server; URL details the event occurred

FAS executes button's .click() method

`.DOMEvent()` goes through several FAS methods to end up executing the application button's **`.Click()`** method.

`.wForm()` opens **`RowSourceType.scx`** as a child form

wForm() asks Application Host for a form reference

Similar to what we saw in the previous episode, `.wForm()` asks application host for a reference to the form; if none exist, AppHost launches the form, saves its state and generate HTML/CSS/JS

.wForm() adds child form dialog into AJAX response

Knowing that form's HTML/CSS/JS is available,

`.wForm()` asks the FAS AJAX response object to add the form's HTML/CSS/JS to the AJAX response.

Form HTML/CSS/JS is encoded into the XML response

FAS response to AJAX requests is encode as an XML stream.

FoxInCloud.js parses this XML response into JavaScript instructions to update HTML page.

Browser receives XML response

Web dev. Tools such as Firebug for Firefox provide a very handy debugging environment; here we can see the full XML response received from FAS server, original request, and much more.

FoxInCloud.js executes orders read in XML response

FAS builds an XML response to AJAX requests (mainly user events).

XML response embeds JS instructions for the HTML display to reflect VFP form's state after user action, just like in desktop.

Child form shows up in the browser

By comparing the state of each form before and after user action [`.Click()` in this case], FAS could send all the necessary orders for `FoxInCloud.js` to update the client HTML page accordingly
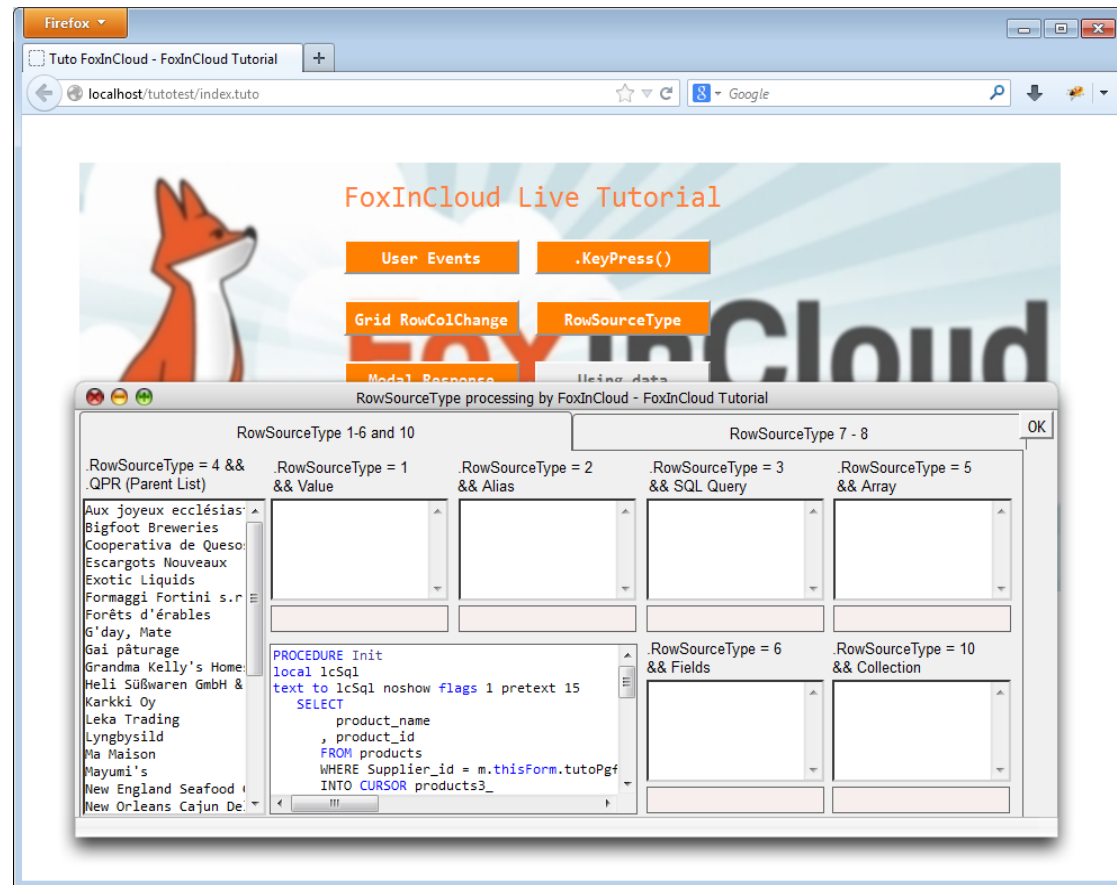
FoxInCloud

A-Z tutorial, Session 2

Requires FAA & FAS V2.0+

Session 2
Adapting your application

To better follow this session, we recommend you download and install both FoxInCloud Adaptation Assistant (FAA) and FoxInCloud Application Server (FAS) `version 2.0+` from http://foxincloud.com/download.php

Both licensed and trial versions of FoxInCloud Application Server install the application presented in this Session in this location:
`<path to VFP9>\Tools\AB\AW\Samples\Tastrade\`

To reproduce the steps presented in this session, please prepare the source files as follows:

- Erase the contents of `…\Tastrade\Sandbox\`
- Copy the contents of `…\Tastrade\_Original\` into `…\Tastrade\Sandbox\`

You may also read adapted source code from `…\Tastrade\Adapted\`

Adapting your application, Agenda

Episode	Subject	Slides
0	A simple business Application	5
1	Analyzing Application (FAA step 1)	8
2.1	Automated Adaptations (FAA step 2)	9
2.2	Manual Adaptations **after** FAA step 2	20
		42

Session 2, prologue
A simple business application
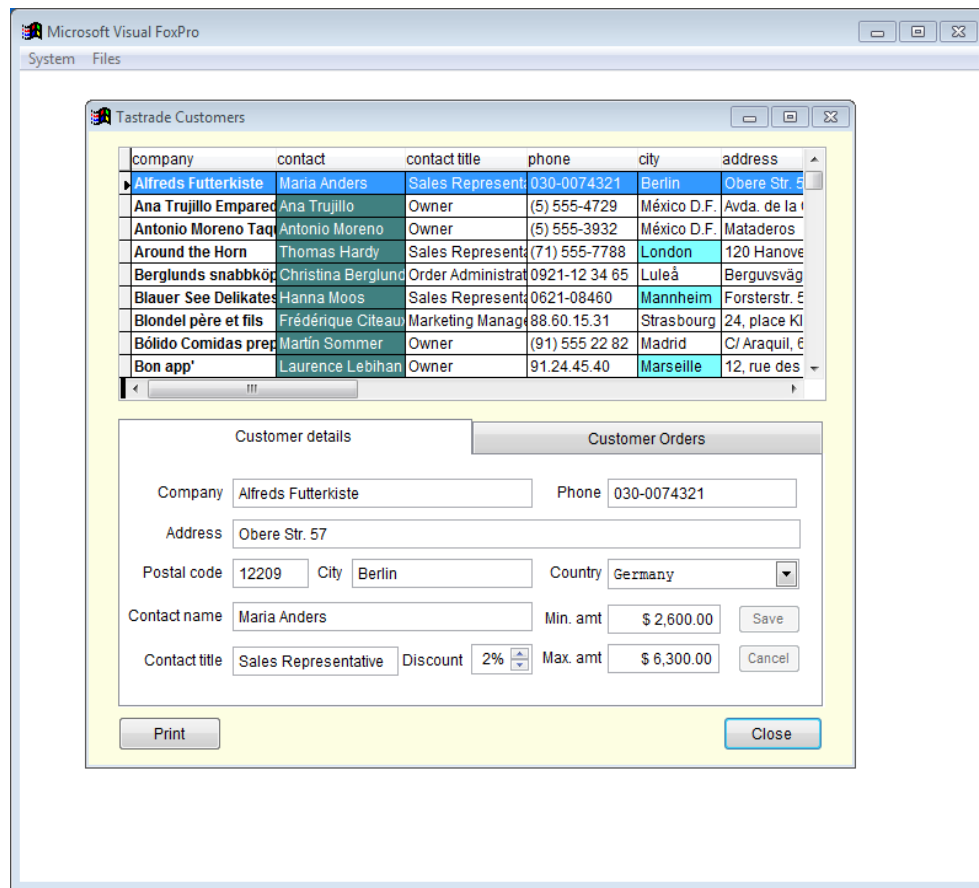
Sample Tastrade Application

Though simple, this application demoes the most common VFP business app. features.
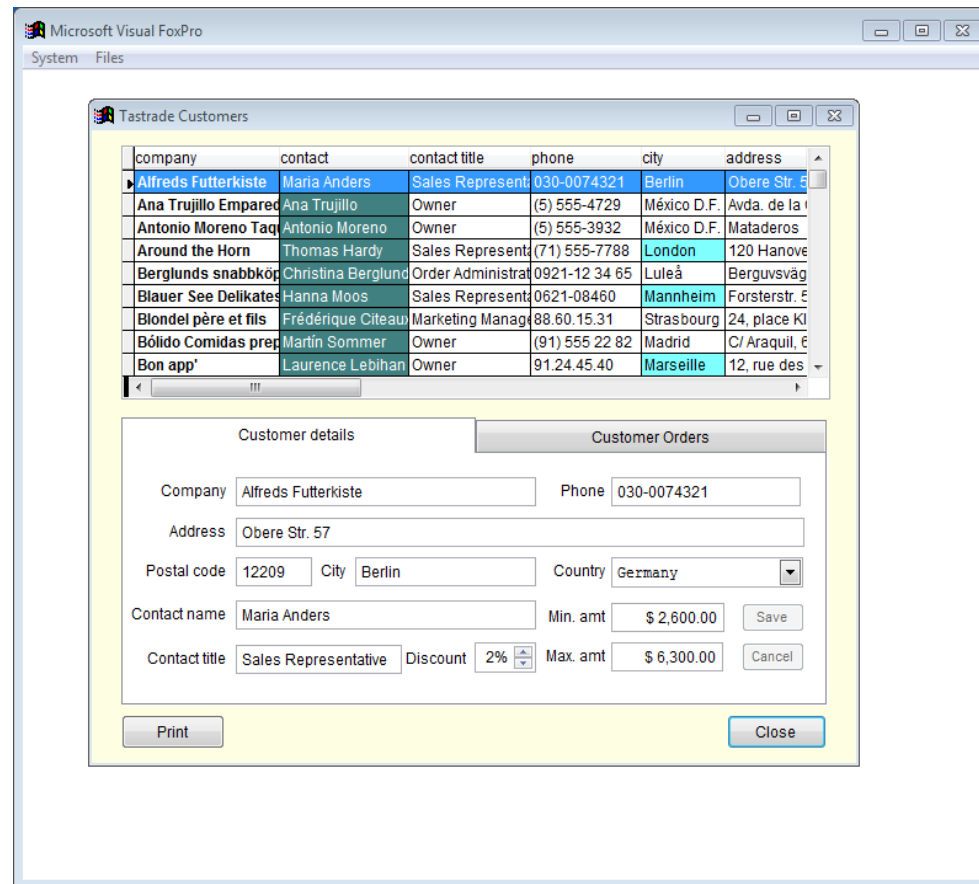
Double clicking orders grid opens order form

Orders grid is located on 'orders' page.

Customer report

Viewing Source Code

You may
open
either
`*.?cx`
or
`*.?cx.src`

Session 2, episode 1
Analyzing application (FAA step 1)

Open FoxInCloud Adaptation Assistant (FAA)

FAA opens in Copy mode by default
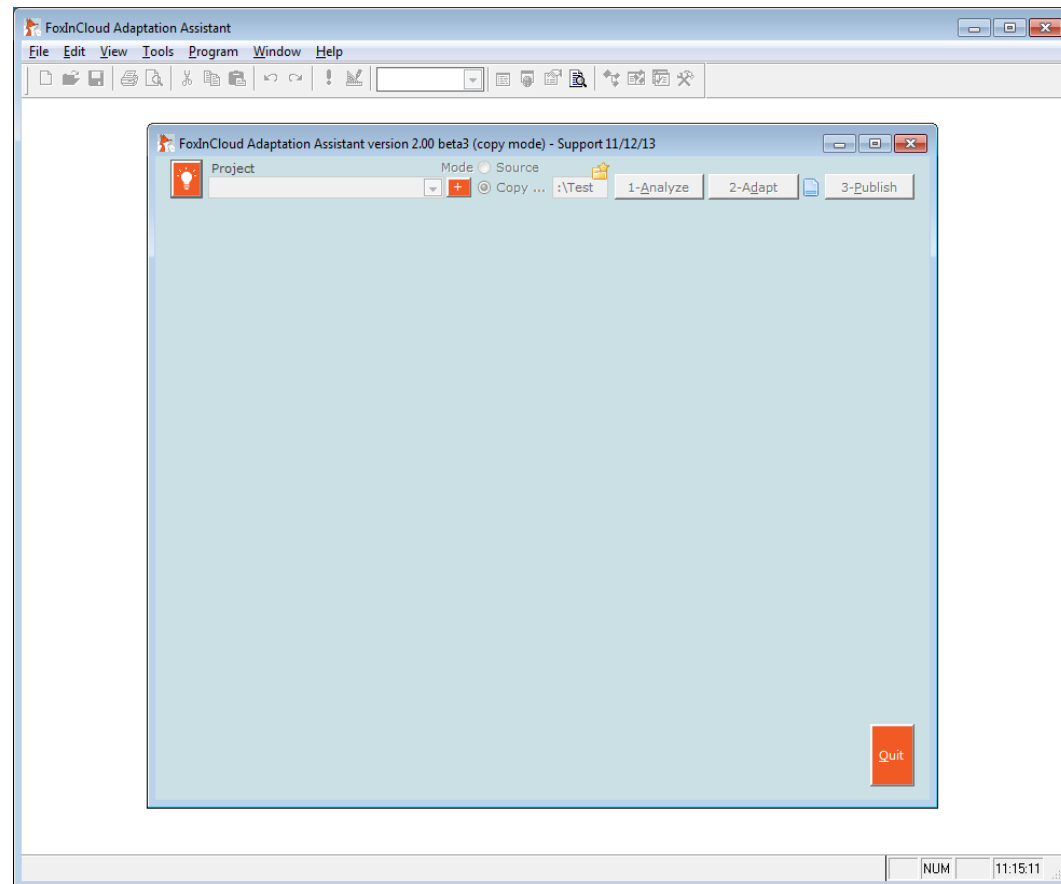
Pick the sample project in Sandbox

`Samples` `\tastrade` `\Sandbox\` is where you can do your tests.

Please keep `Samples` `\tastrade` `\_Original\` unchanged
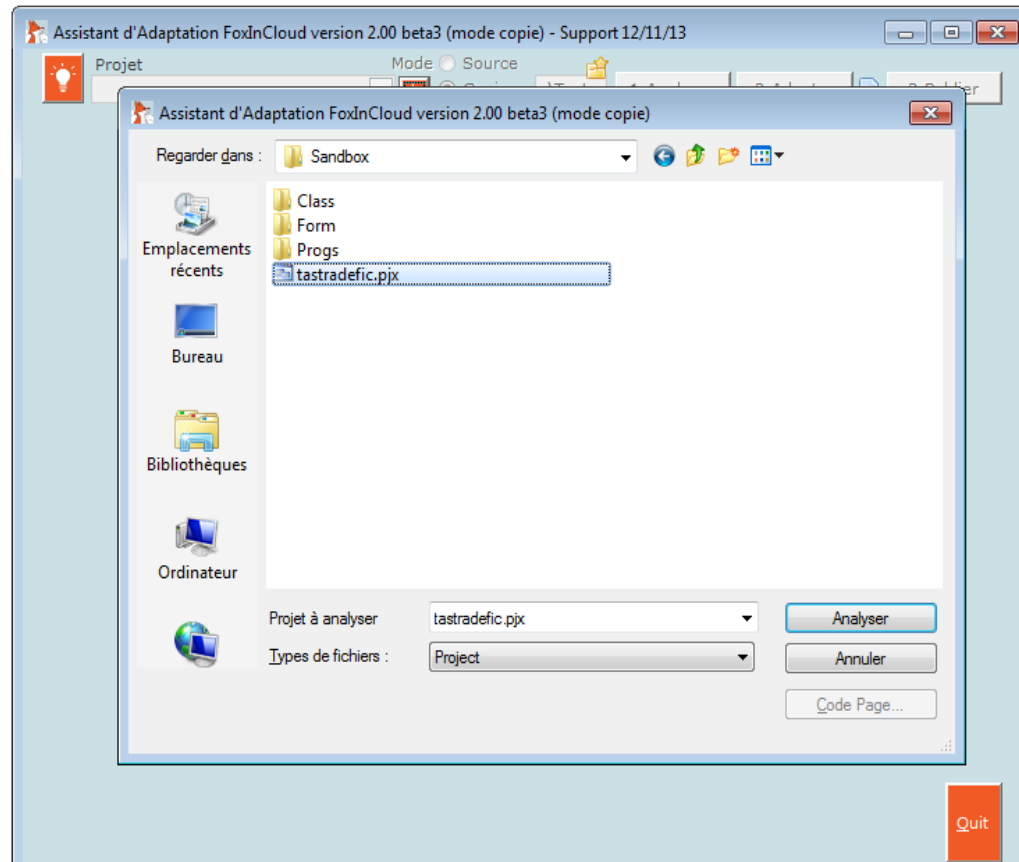
Switch to source mode

As we are working in a sandbox, we can go straight to source mode.

For a real application, running copy mode first is highly recommended
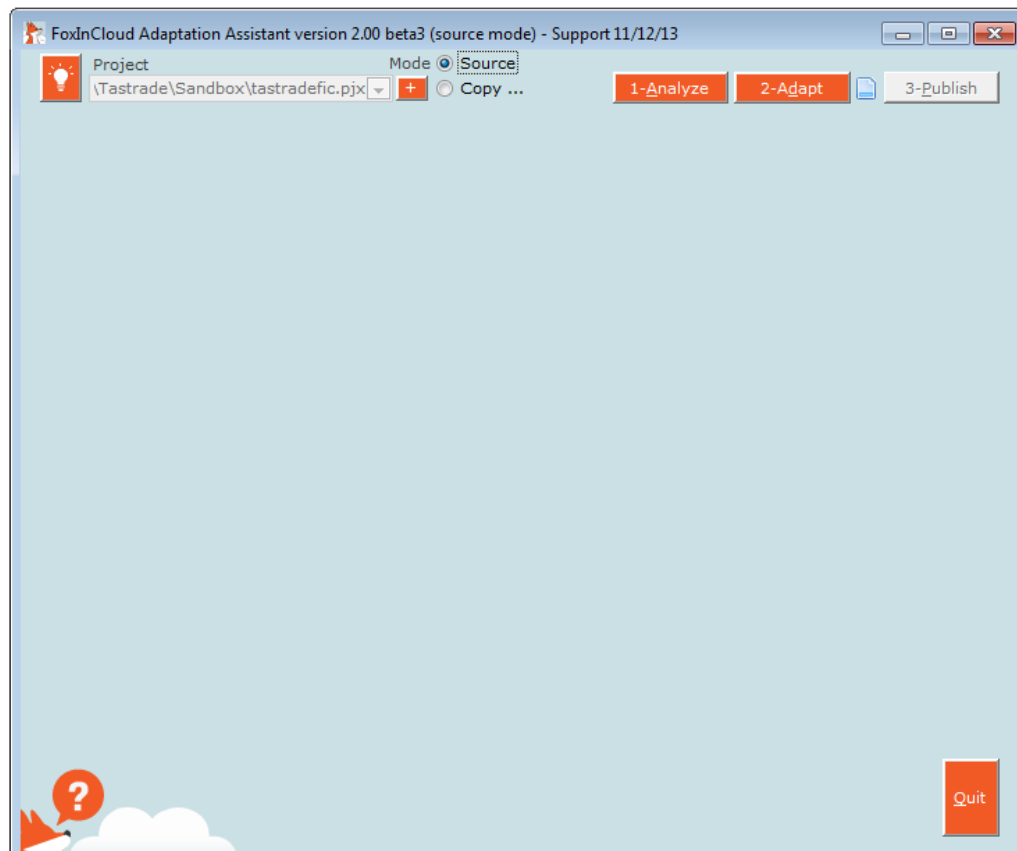
Analysing …
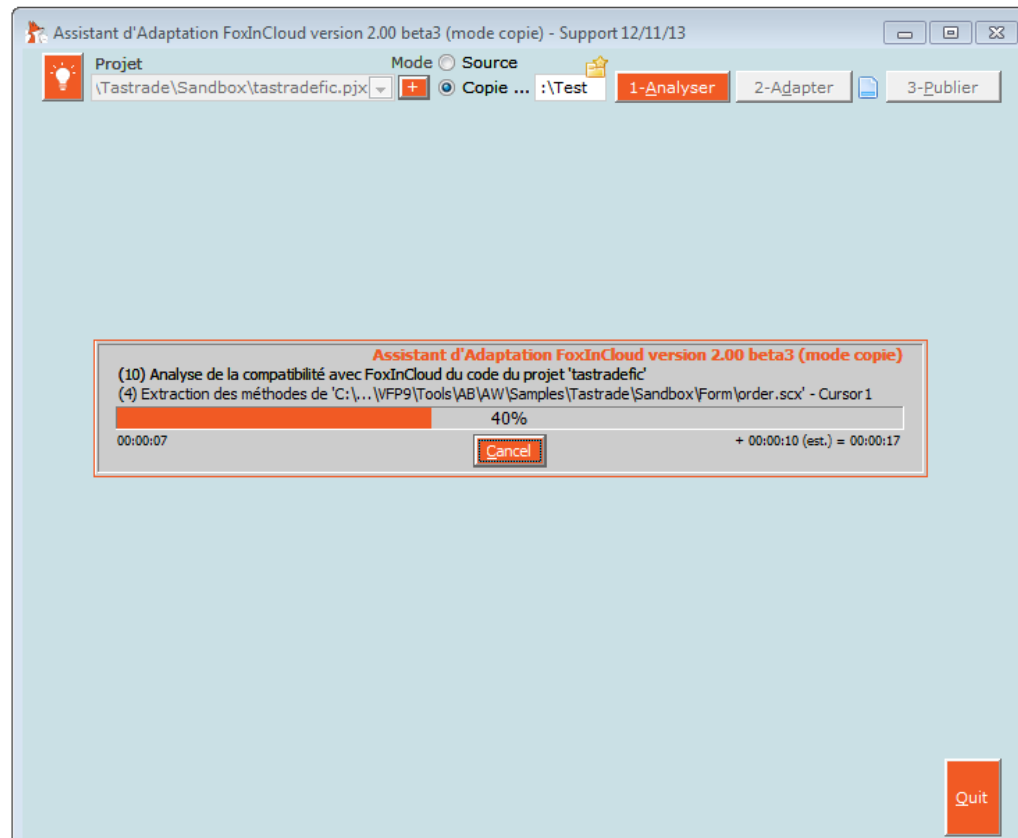
Sharing analysis results

Sharing your results not only unlocks the automated adaptation features, it helps FoxInCloud know the current adaptation issues

Any question? Post with screenshot in the 'FoxInCloud' section of http://west-wind.com/wwThreads/ or http://www.universalthread.com/
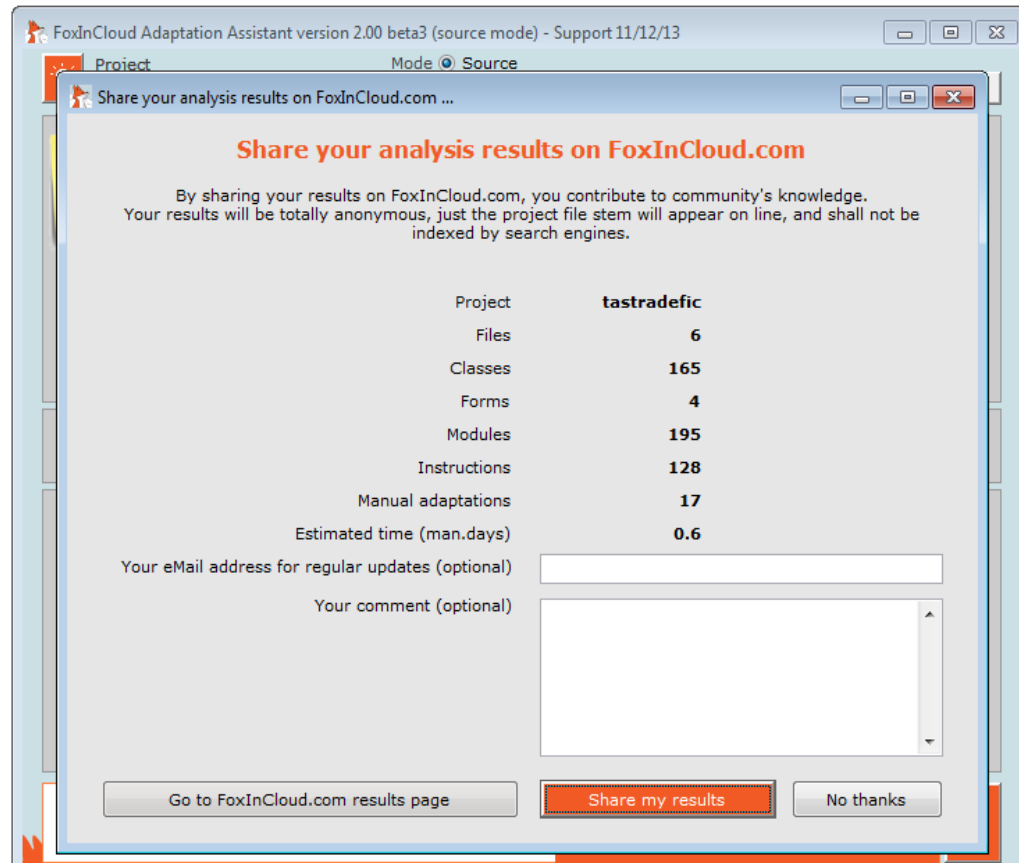
Browsing results

For details on how to use the FAA interface, please browse foxincloud.com/how-to.php

Double clicking a grid line opens source code

Session 2, episode 2.1
Automated adaptation (FAA step 2)

Once adapted with FoxInCloud, your application can run on the desktop as usual, and on the Web within FoxInCloud Application Server
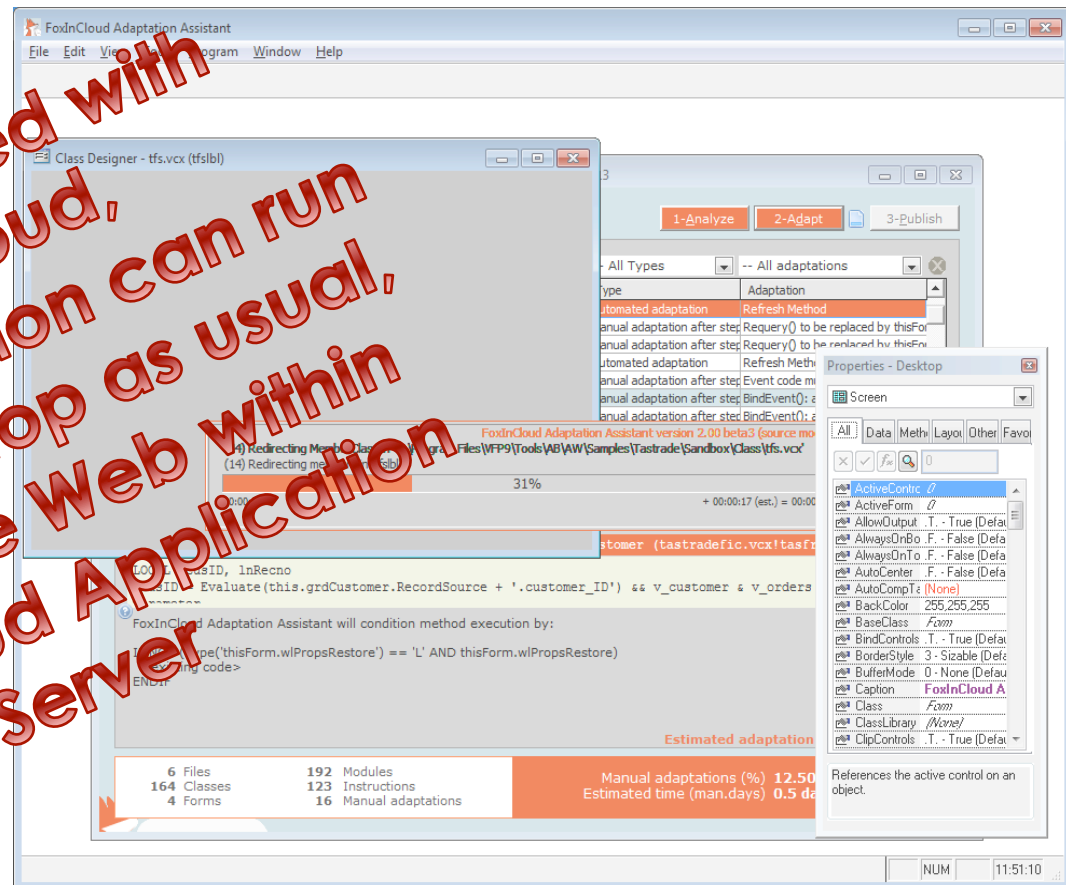
Let's start adaptation

As FAA needs write files into the `program files` directory, when using Windows >= Vista, you need to run FAA as an admin.
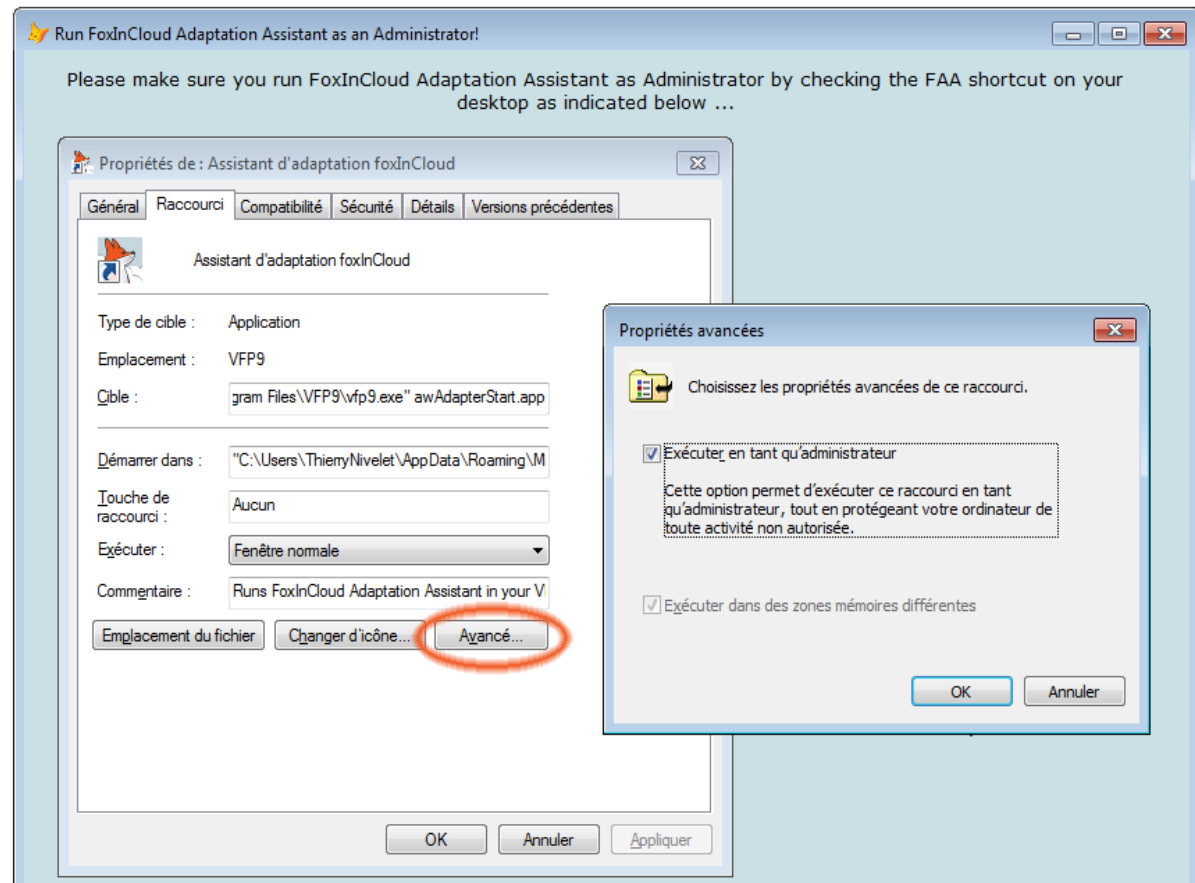
1- Folder where you want to store your sub-classes of FiC

Folder where FAA stores sub-classes of FiC 'aw' *base* classes.

It can be either within your project dir. or shared across projects.
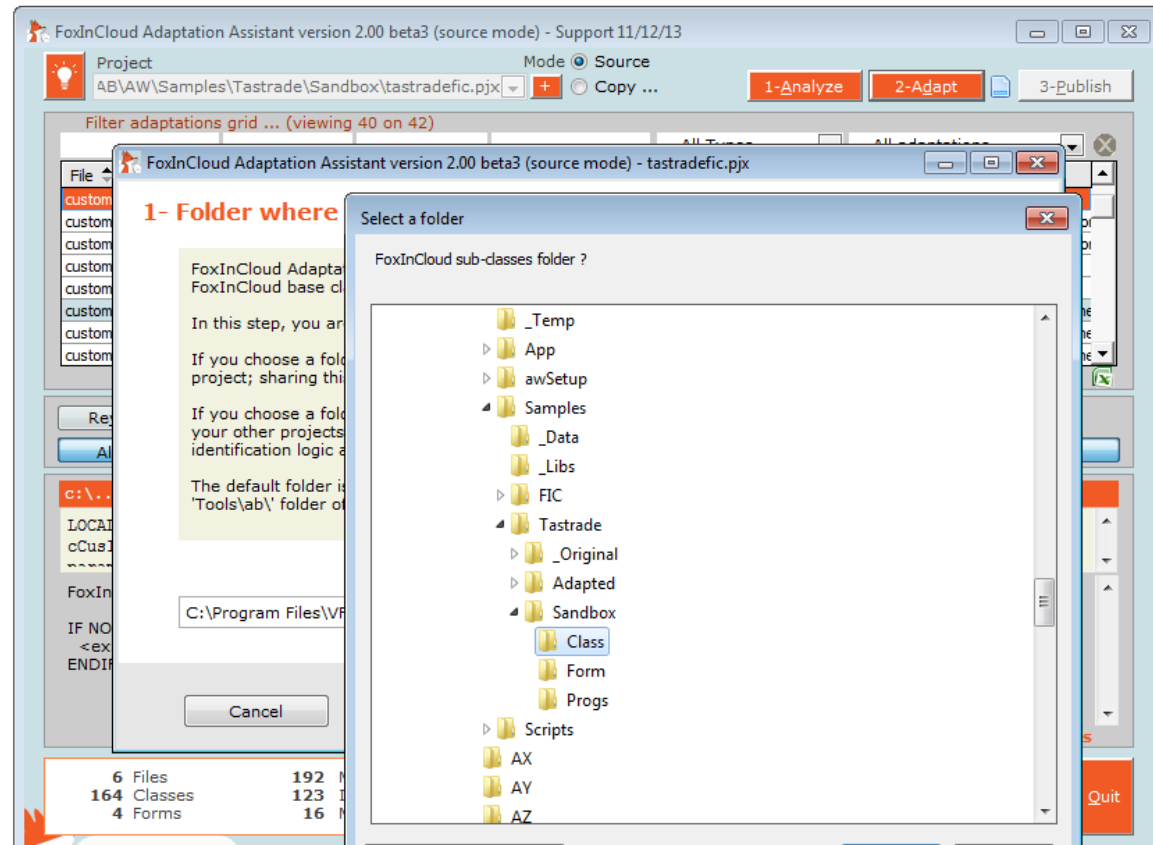
2- FiC sub-classes prefix

Your vcx and contained classes will share a common prefix.
This prefix is used throughout your web app.

('tfs' stands for Tastrade FiC Sandbox)
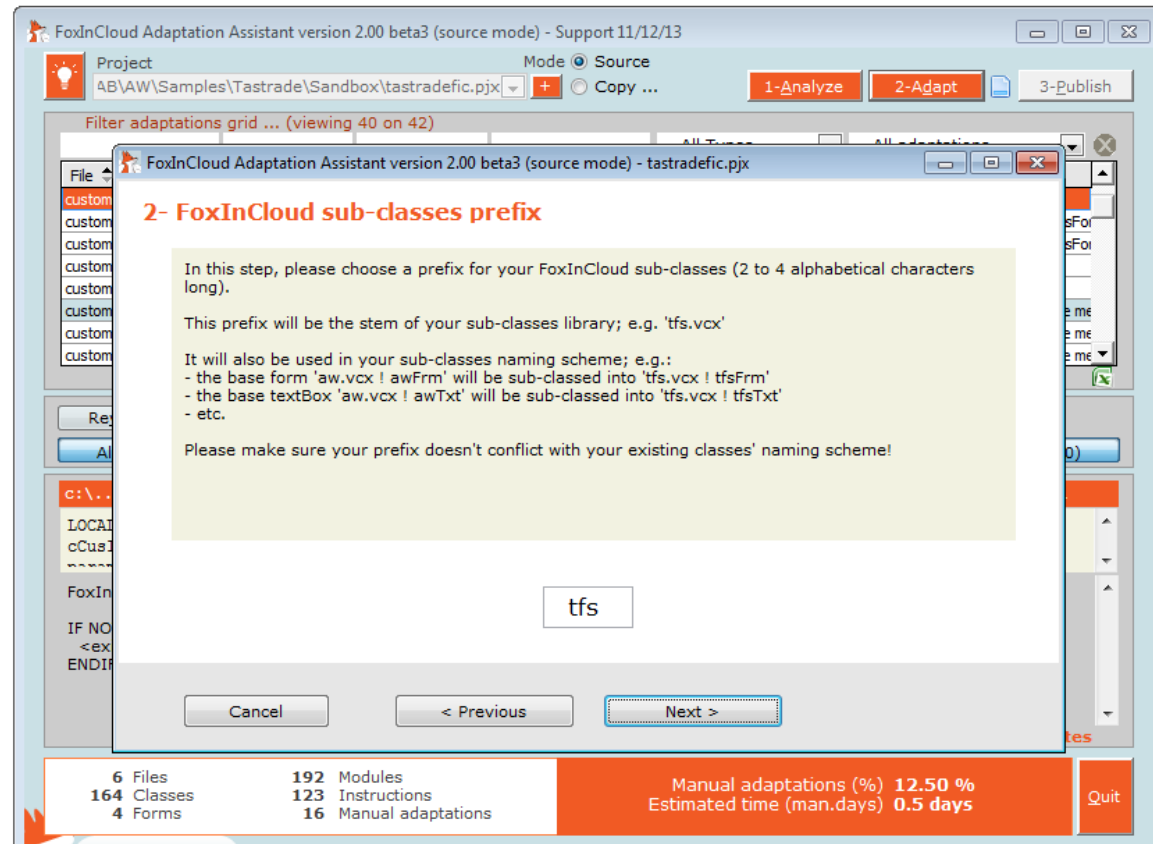
Adapting your app …

Automated Adaptations:

1- replace any native VFP class by your set of sub-classes

2- make your code web-compatible

Testing the adapted app.

After automated adaptations, FAA invites you to check that your app. works exactly the same as before.

Testing the adapted app…

All features of your original app. should work exactly the same with the adapted source code
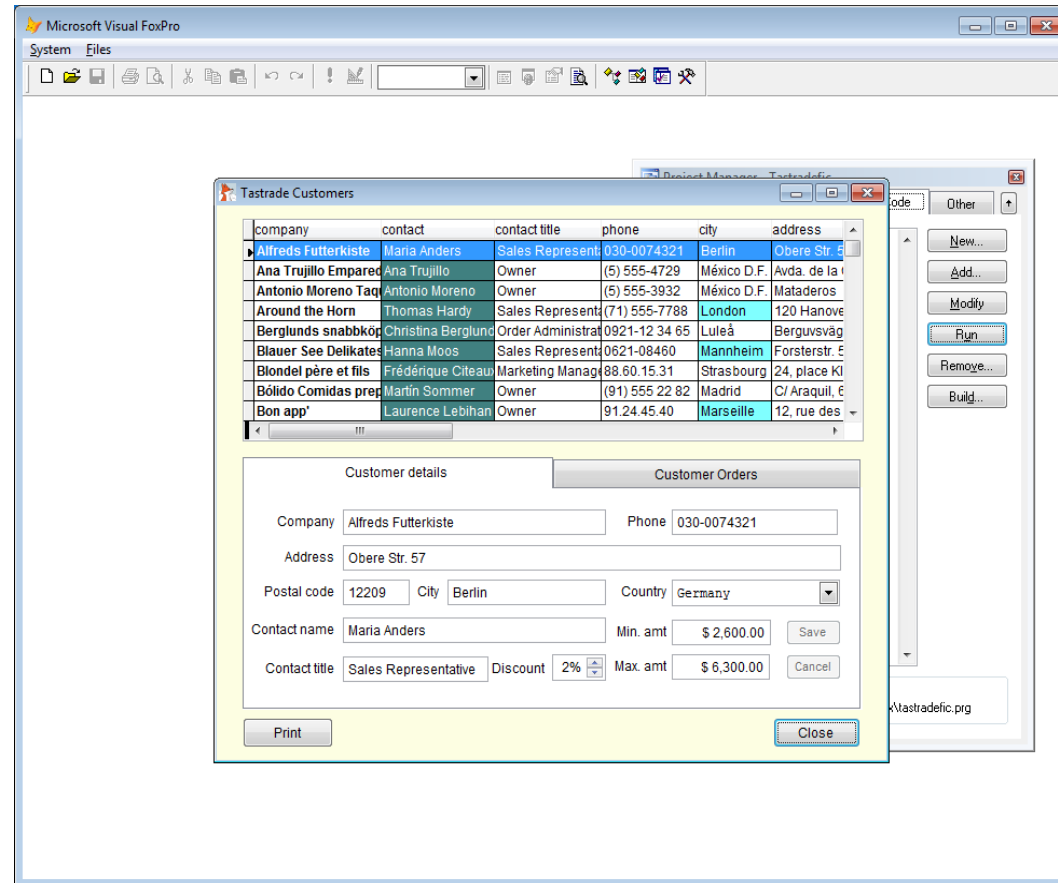
Testing the adapted app ...

You check that modal child forms and value returned to parent still work OK
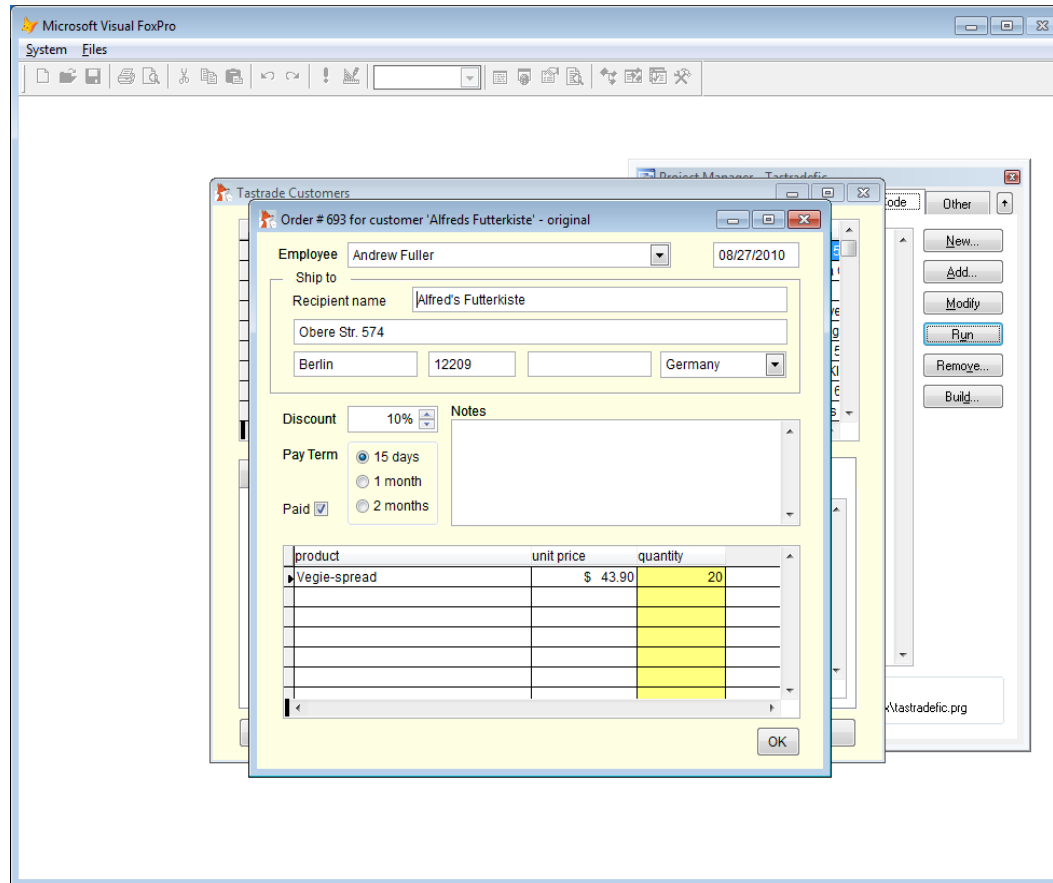
Comparing source code before / after automated adaptations

Using a source code diff utility such as Win Merge helps compare code before/ after adaptation (see *.src.?cx)

Session 2, episode 2.2
Manual adaptations after FAA step 2
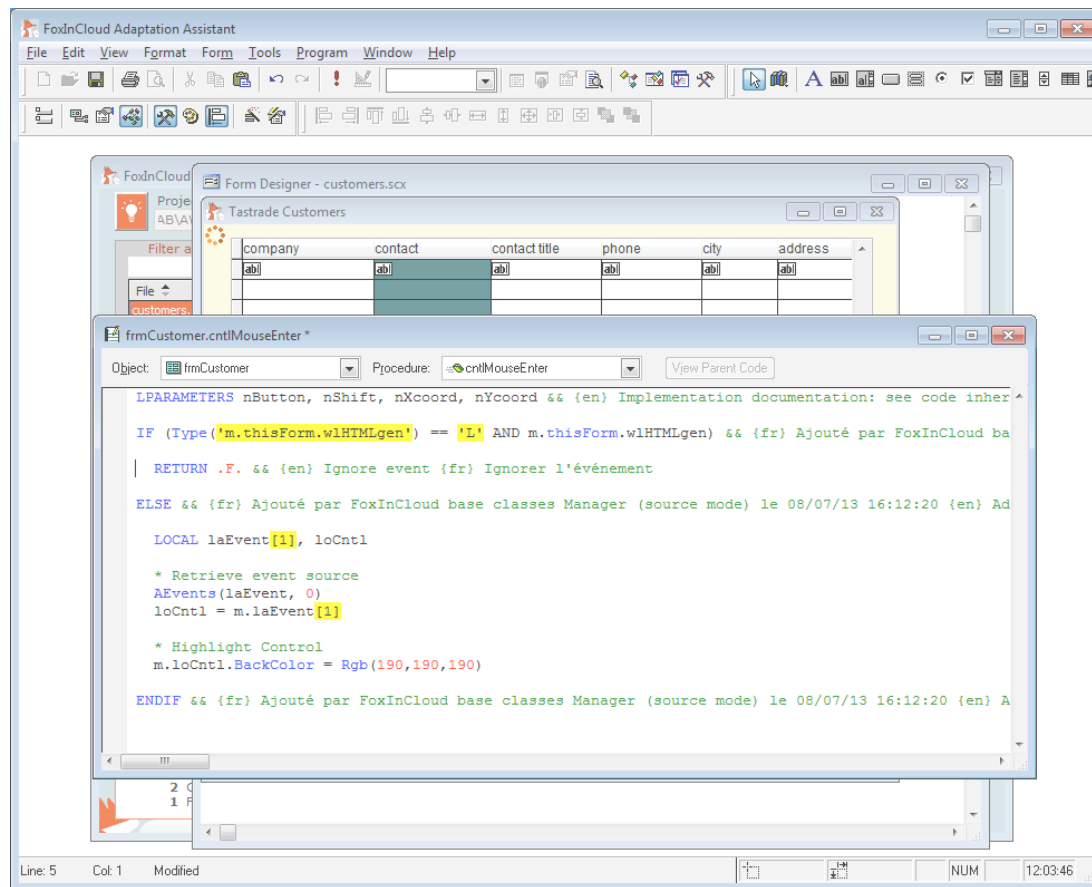
(A) Moving modal response code to a call-back method

Double-clicking an orders grid row opens the 'order' child form.

Parent form expects a Boolean response from 'order' child form.

(A) Moving modal response code to a call-back method…

… if child form returns .T. user has modified current order; update base table and feedback success to user.

(A) Moving modal response code to a call-back method...

1/ Set `.wForm()` second parameter as the name of the `callback` method where response code will execute. (relative to this object or absolute to the form)

(A) Moving modal response code to a call-back method…

2/ cut the response code and paste into the target call-back method.

3/ Give the parameter the same name as the variable in source method.

Any question? Post with screenshot in the 'FoxInCloud' section of http://west-wind.com/wwThreads/ or http://www.universalthread.com/

(B) Adapting delegate event methods

While FAA auto adapts native events, you need to adapt manually methods that are event delegates.

(B) Adapting delegate event methods…

During HTML generation, FAS implements JS events based on on the value returned by VFP event methods.

Adapt delegate method as if it was the source event.

(C) Checking .RowSource assignments in code

For any `combobox` or `listbox` object or class, make sure your code (re)assigns `.RowSourceType` just before assigning a non-empty value to `.Rowsource`

(D) Replacing Requery() by thisform.wViewParmSet()

In order to restore form datasession state for each user, FAS needs to know the name and values of view parameters

(D) Replacing Requery() by thisform.wViewParmSet()…

Comment `Requery()` and add one call to `thisform.wViewParmSet()` per view parameter:

- View name
- Parm name
- Parm value
- .T. triggers requery().

(E) Splitting .*RowColChange() code into .*RowChange() / .*ColChange()

Whereas VFP fires a single event for grid row and col change, JS grids fire 2 different events.

FAS should run the right code.

(E) Splitting .*RowColChange() code to .*RowChange() / .*ColChange() ...

1/ Isolate portions of code that should run on either row or column change.

e.g., If row change, open `.wAfterRowChange()` code
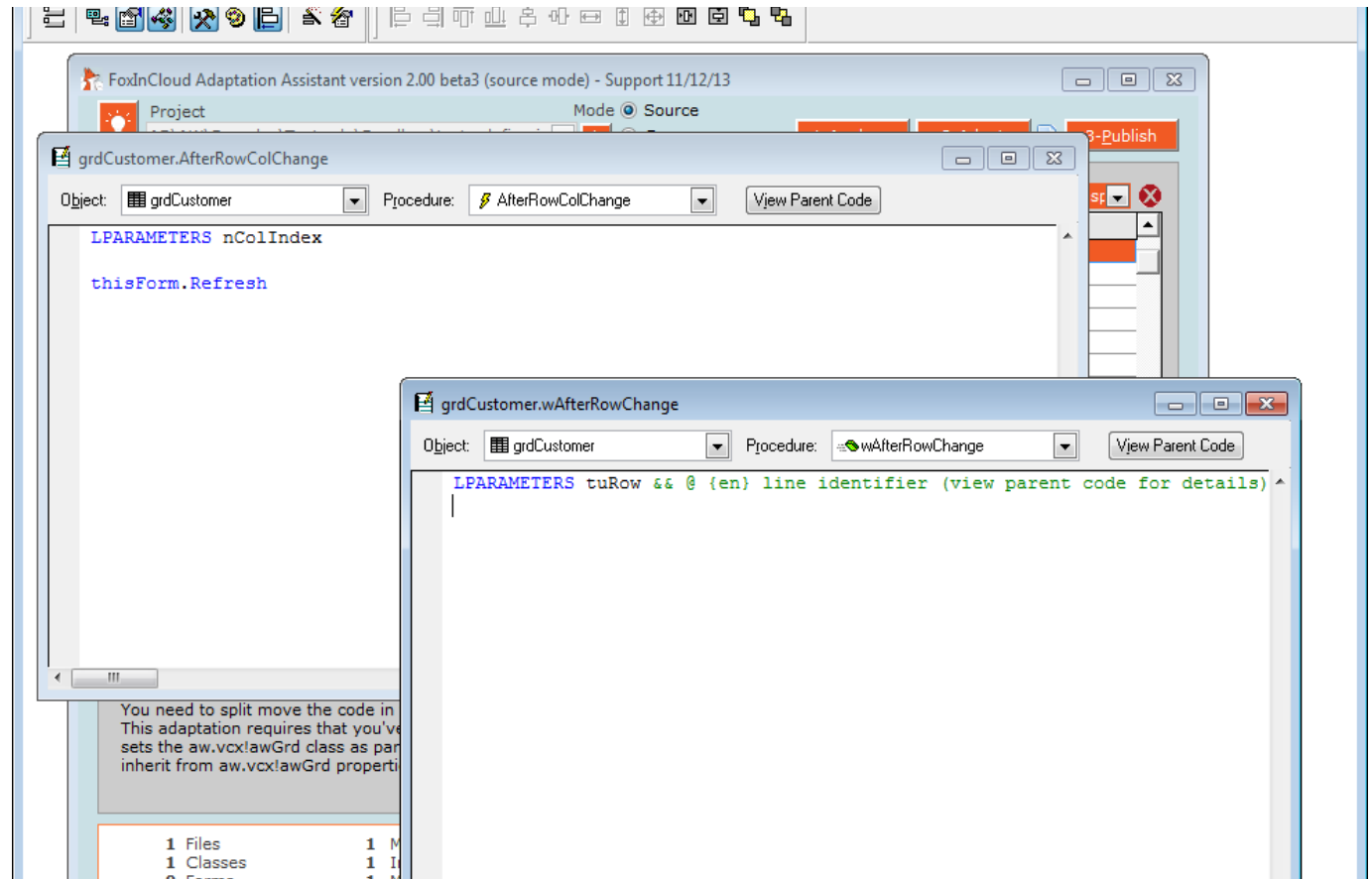
(E) Splitting .*RowColChange() code to .*RowChange() / .*ColChange() …

2/ Click 'view parent code', select `awGrd`, copy and paste inherited implementation code sample into your empty code snippet

(E) Splitting .*RowColChange() code to .*RowChange() / .*ColChange() …

3/ Move your source code to the location indicated

4/ Empty your original .*RowColChange() method.

(F) Reporting

Because VFP reporting involves Windows resources, you need a different REPORT code in desktop mode and Web mode.

(F) Reporting…

1/ Create 2 blocks of code, one for desktop mode, one for web mode.

To test desktop or web mode, you can use `wlWeb()` or `wlLAN()` (in `awPublic.prg`), or `thisForm.wlWeb`

FoxInCloud Adaptation Assistant version 2.00 beta3 (source mode) - Support 11/12/13

Project: AB\AW\Samples\Tastrade\Sandbox\tastradefic.pjx   Mode ● Source ○ Copy …   1-Analyze   2-Adapt   3-Publish

Filter adaptations grid … (viewing 1 on 25)

(8) Manual adaptati ▾ (1) REPORT FORM in Web ▾ ⊗

File ▲	Object/Class	Base Class	Module	Type	Adaptation
customers.scx	frmcustomer.cmdpr	Commandbutton	Click	Manual adaptation after ste	REPORT FORM in Web mode

cmdPrint.Click *

Object: ▣ cmdPrint ▾   Procedure: ⚡ Click ▾   View Parent Code

```
      For more details and options, see code inherited from aw*

   #endif && {en} ======= / Implementation documentation ========== {en}


   RETURN .T.&& Process event on server (ELSE block code)

 ELSE && Added by FoxInCloud Adaptation Assistant version 2.00 beta3 (source mode) on 11/12/13

   local customerID
   customerID = Trim(customer.Customer_ID)

   if wlLAN()
     report form customer for Customer_ID = m.customerID preview && runs in a private datasessi
   else

   endif

 ENDIF && Added by FoxInCloud Adaptation Assistant version 2.00 beta3 (source mode) on 11/12/13
```

(F) Reporting...

2/ In Web mode, `REPORT FORM` … to a PDF file using the PDF generator of your choice.

3/ Send back the PDF to the browser using `thisform.wFileSaveAs()`

Testing the adapted application

Just like you did after automated adaptations, run your application and make sure every feature works exactly the same as before …

Manual adaptations after step 2 completed!

If you ignore each adaptation once done, or re-analyse your project, your adaptation count and time comes down to 0!

Remind of Manual Adaptations after step 3…

Keep in mind that you'll have a few more adaptations to review after next step (3-Publish) is complete!

FoxInCloud

A-Z tutorial, Session 3

Requires FAA & FAS V2.0+

Session 3
Publishing your application on localhost

To better follow this session, we recommend you download and install both FoxInCloud Adaptation Assistant `(FAA)` and FoxInCloud Application Server `(FAS) version 2.0+` from
http://foxincloud.com/download.php

Both licensed and trial versions of FoxInCloud Application Server install the application presented in this Session in this location:
`<path to VFP9>\Tools\AB\AW\Samples\Tastrade\`

To reproduce the steps presented in this session, please follow session 2 beforehand.

You may also read adapted source code from `…\Tastrade\Adapted\`

Publishing your application on localhost, Agenda

Episode	Subject	Slides
1	Automated publication (FAA step 3)	16
2	Manual adaptations after FAA step 3	17
3	Getting prepared for production	8
		41

Session 3, episode 1
Automated publication (FAA step 3)

Click FAA's Publish button

Clicking the publish button starts the automated installation of your web application in http:// localhost/ using IIS

1/ Checking your IIS installation

FAA makes sure your dev. machine has a proper version of IIS installed.

1/ Checking your IIS installation…

If your machine runs IIS>=7, you're invited to manually check that all the necessary IIS options are installed.
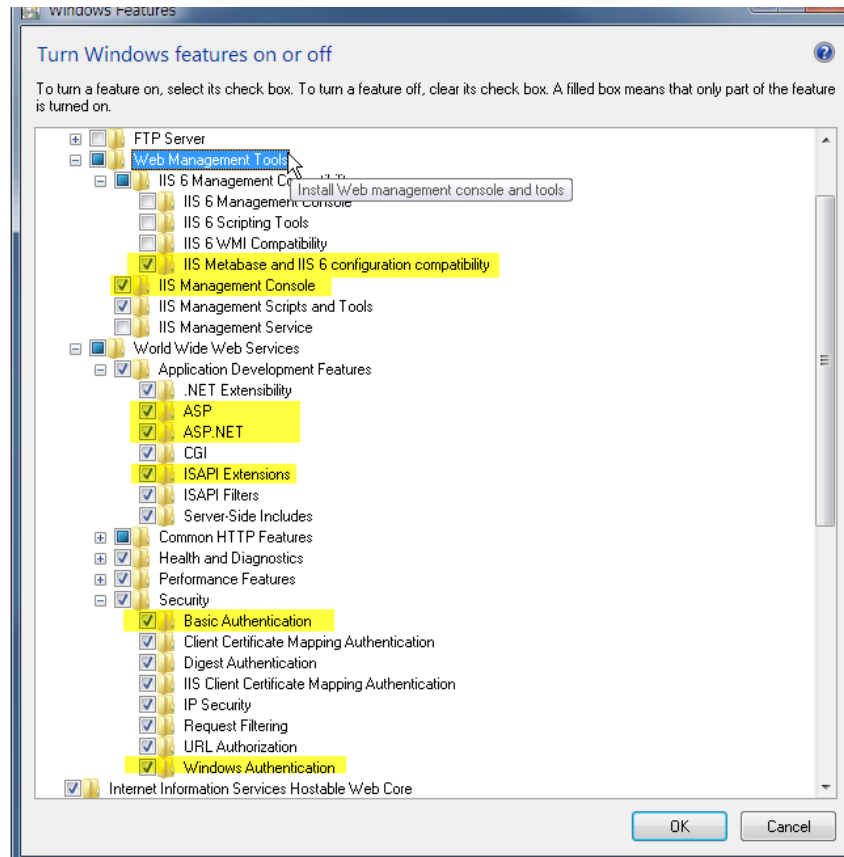
2/ Checking your FAS installation

FAA checks FoxInCloud Application Server is installed on your machine; if not, get a free download using the form displayed .
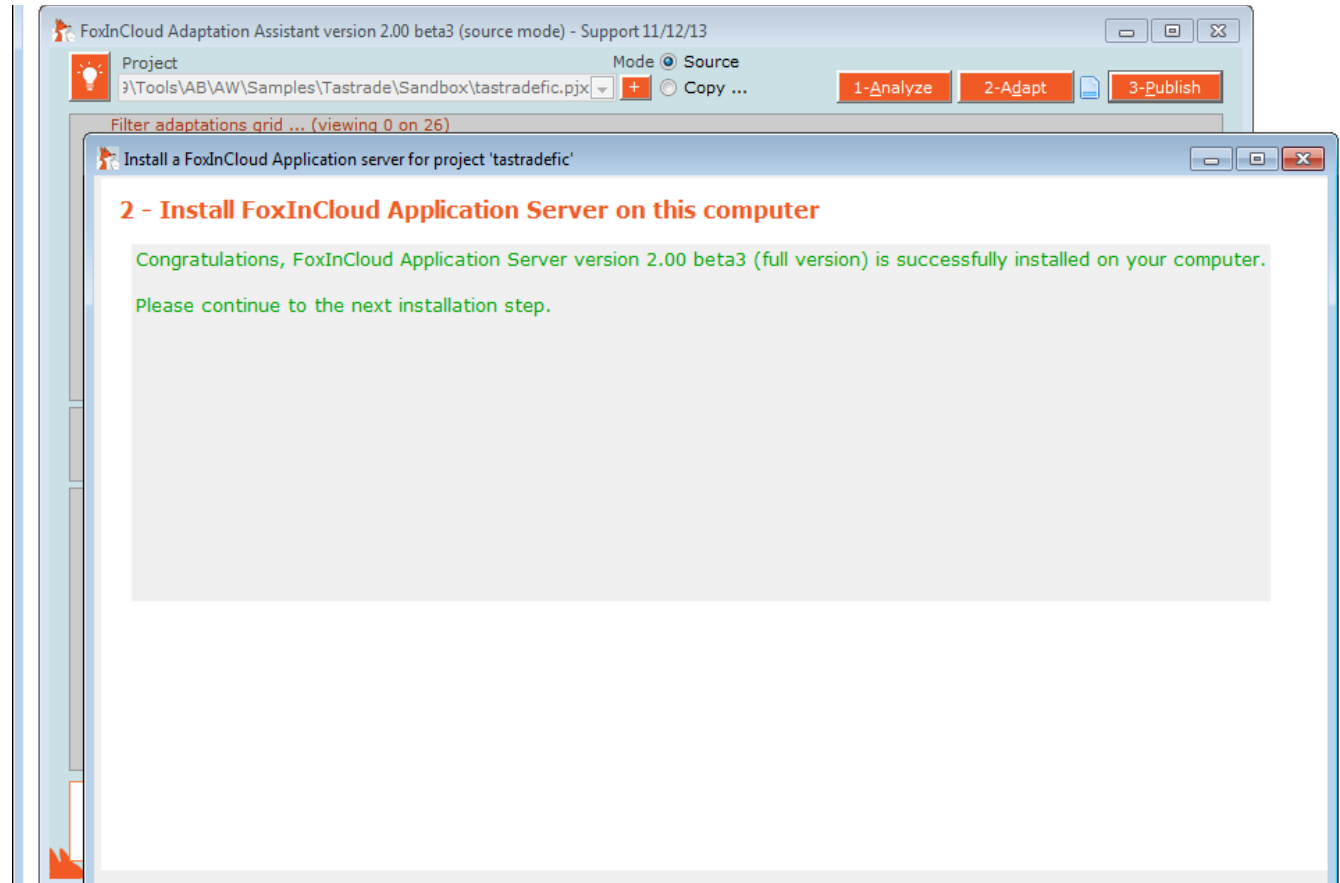
3/ Installing FoxInCloud's *.js and *.css

FAA creates the `awScripts /` virtual directory in localhost where all your Web apps can access FoxInCloud standard `*.js` and `*.css`

FoxInCloud Adaptation Assistant version 2.00 beta3 (source mode) - Support 11/12/13

Project
9\Tools\AB\AW\Samples\Tastrade\Sandbox\tastradefic.pjx Mode ● Source ○ Copy ...   1-Analyze   2-Adapt   3-Publish

Filter adaptations grid ... (viewing 0 on 26)

Install a FoxInCloud Application server for project 'tastradefic'

3 - Installing virtual directory /awScripts/

Congratulations, virtual directory /awScripts/ was successfully installed.

Before proceeding to the next installation step, please check that /awScripts/ functions correctly by clicking the image below; you should see the same image in your default browser.
Thank you.

FoxInCloud
CSS/Javascript powered

3/ Installing FoxInCloud's *.js and *.css …

When this step is completed successfully, clicking the image in FAA opens the same image in your default Web browser

4/ Installing Web Connect Temp folder

wConnect stores requests, responses and text logs into a Temp folder.

By default, FAA installs this folder in a public area where IIS anonymous user has full rights.
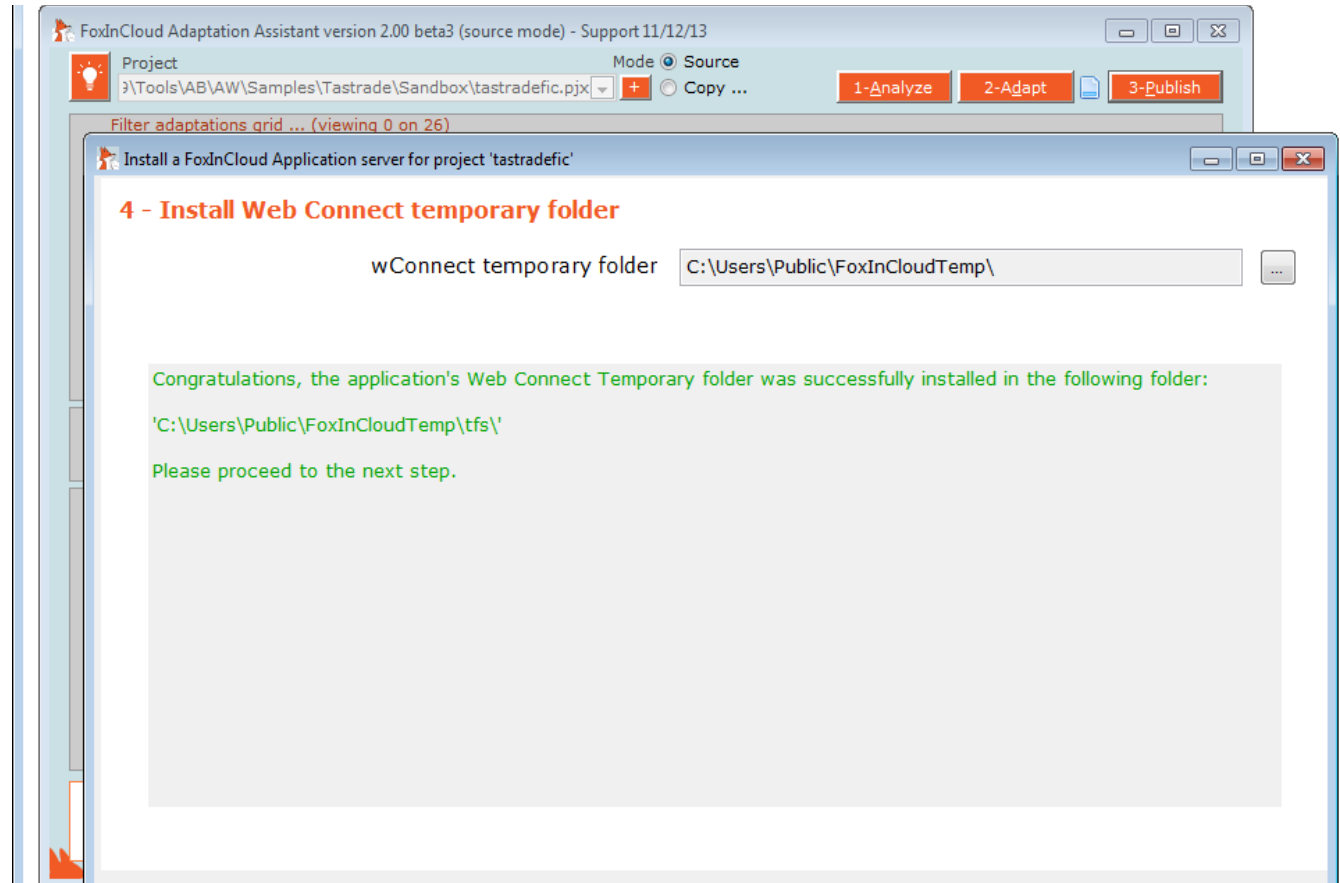
5/ Installing your FoxInCloud Application Server files

Application code prefixes your FAS sub-class names, and serve as default URL extension;

using the suggested app code is recommended
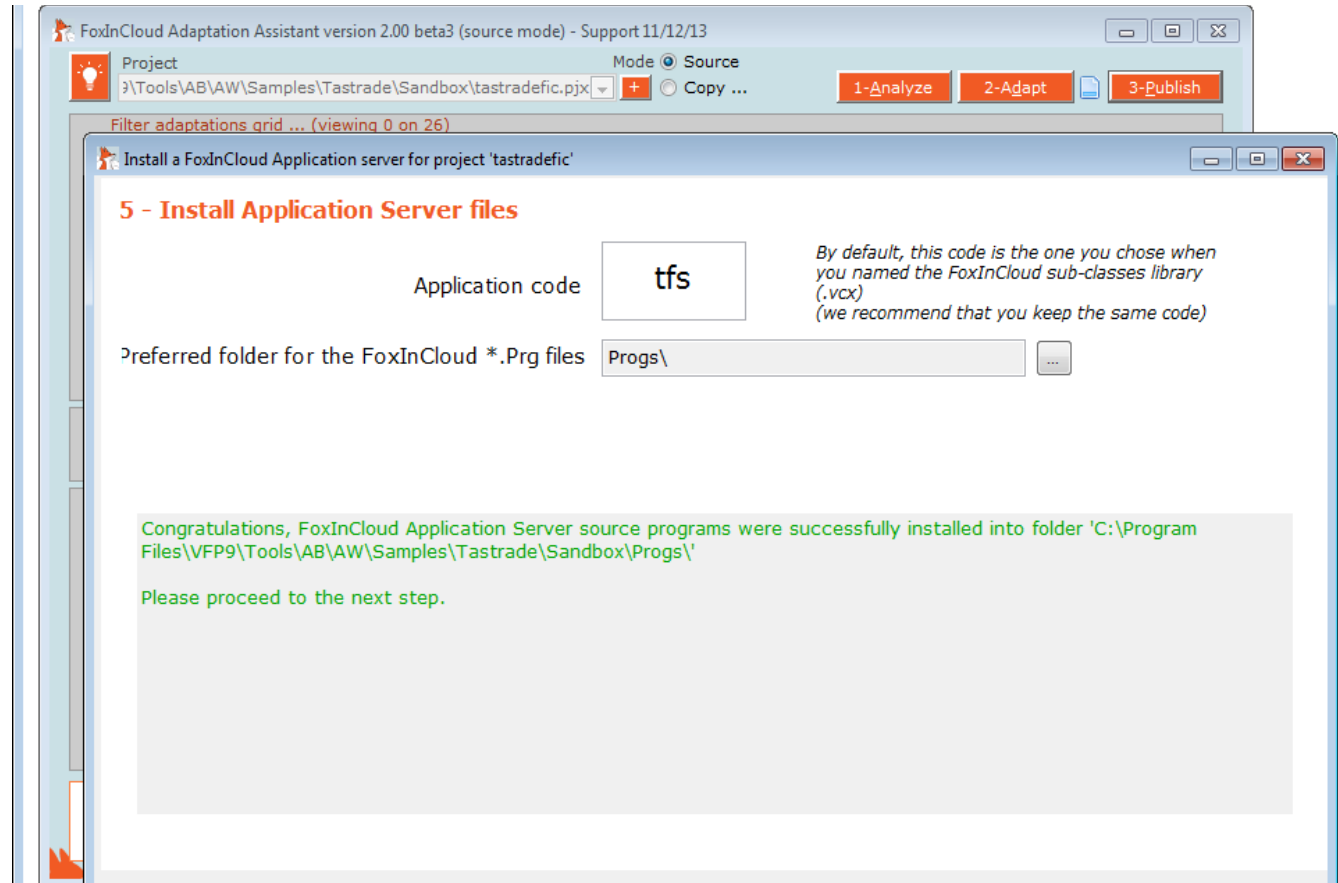
6/ Installing app virtual directories

FAA installs 2 virtual dirs., one for test and one for production.

Both your site and projects are split into test and prod.
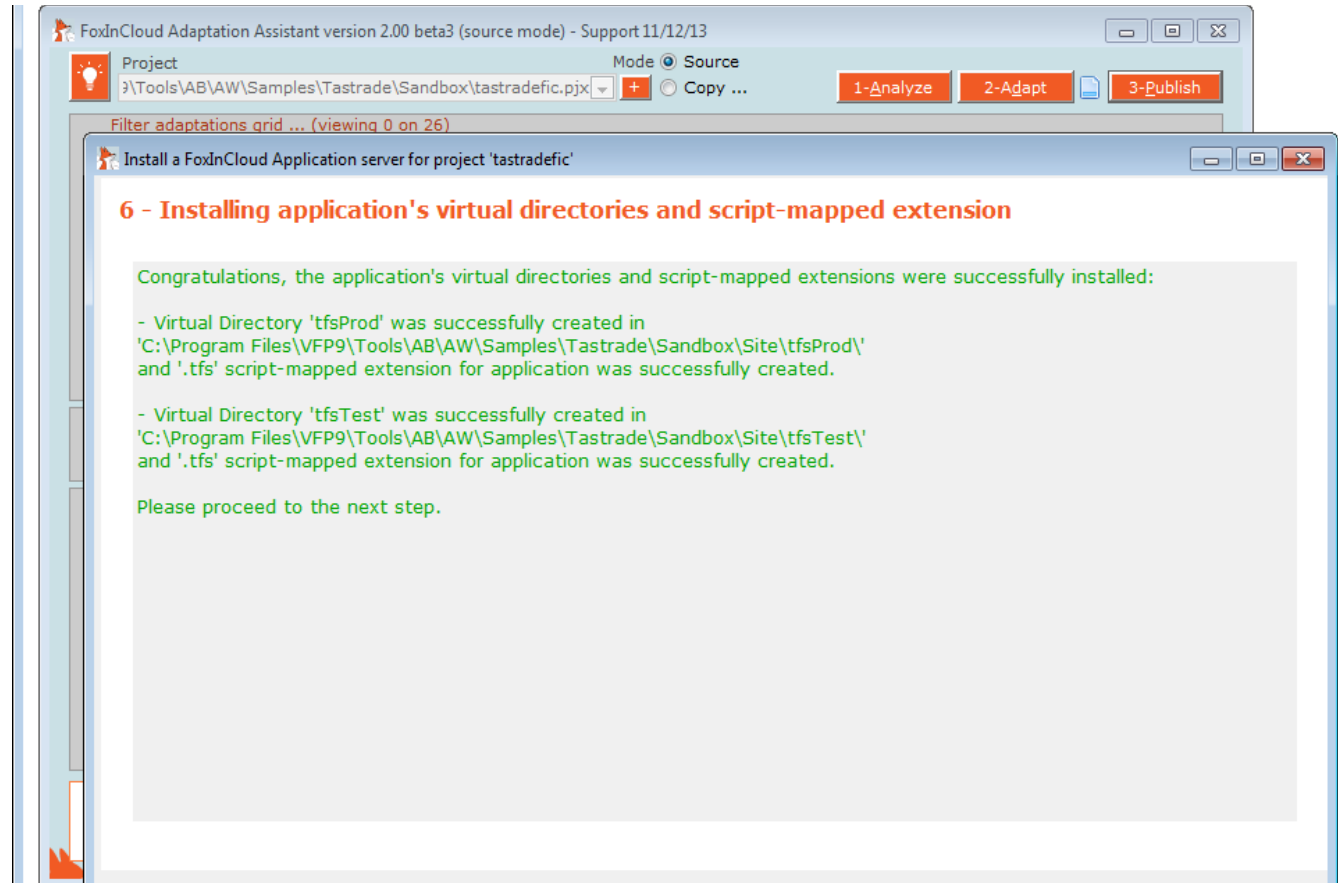
7/ Building application server projects

FAA builds 2 projects so that you can later install 2 distinct apps and COM objects on your production server.
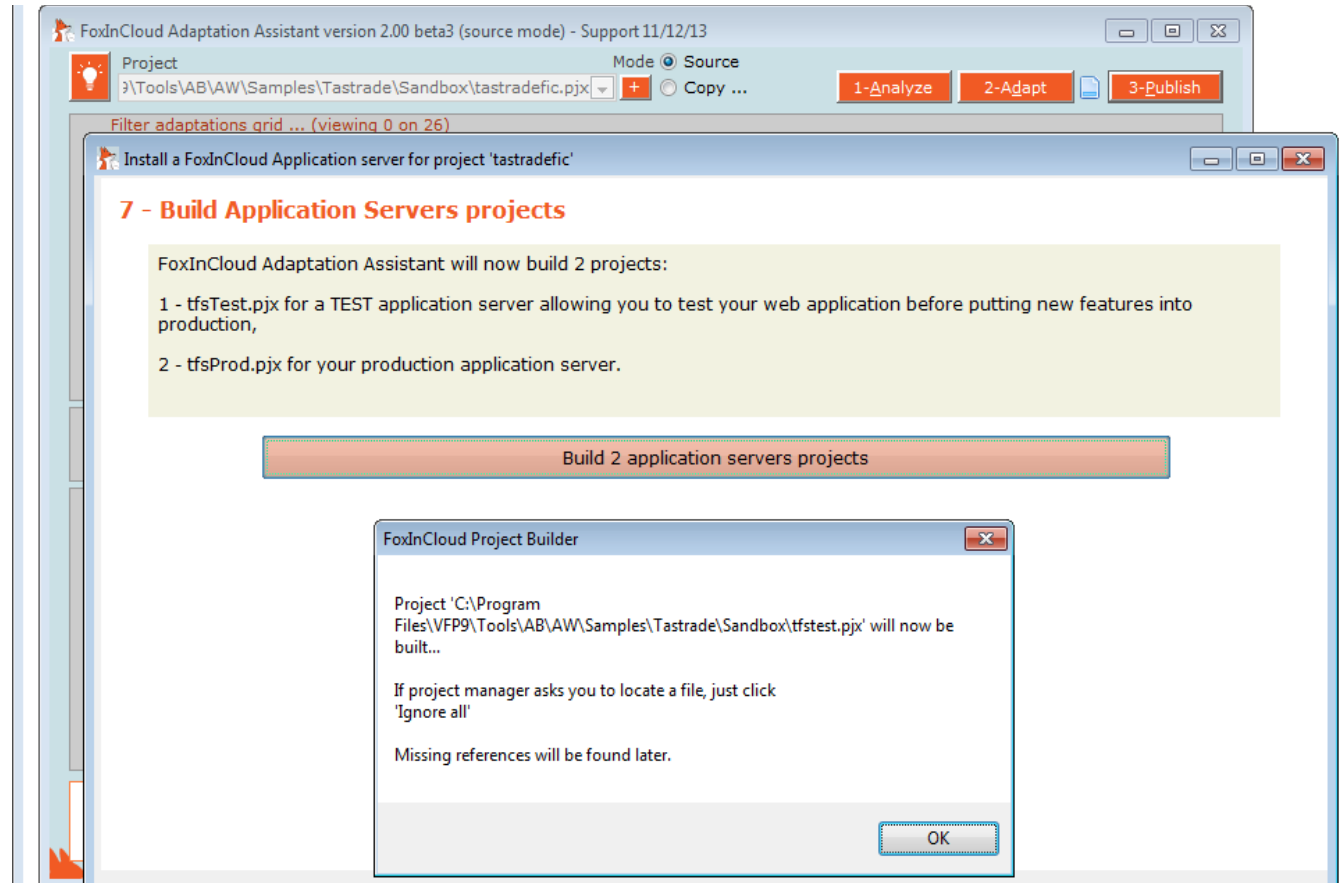
7/ Building application server projects ...

While attempting to build your projects, if FAA asks for missing references, just `ignore all`, references will be found later.

FoxInCloud Adaptation Assistant version 2.00 beta3 (source mode) - Support 11/12/13

Project
9\Tools\AB\AW\Samples\Tastrade\Sandbox\tastradefic.pjx   Mode ● Source   ○ Copy ...   + 1-Analyze   2-Adapt   3-Publish

Filter adaptations grid ... (viewing 0 on 26)

Install a FoxInCloud Application server for project 'tastradefic'

7 - Build Application Servers projects

FoxInCloud Adaptation Assistant will now build 2 projects:

1 - tfsTest.pjx for a TEST application server allowing you to test your web application before putting new features into production,

2 - tfsProd.pjx for your production application server.

Locate File

Unable to find Program WWSERVER

Locate   Ignore   Ignore all   Cancel

7/ Building application server projects …

FAA warns you if some reference are still missing due to an incomplete `Set('PATH')`.

This is a **simple WARNING**, you can carry on.

FoxInCloud Adaptation Assistant version 2.00 beta3 (source mode) - Support 11/12/13

Project: 9\Tools\AB\AW\Samples\Tastrade\Sandbox\tastradefic.pjx
Mode: ● Source ○ Copy …
1-Analyze   2-Adapt   3-Publish

Filter adaptations grid … (viewing 0 on 26)

Install a FoxInCloud Application server for project 'tastradefic'

7 - Build Application Servers projects

FoxInCloud Adaptation Assistant will now build 2 projects:

1 - tfsTest.pjx for a TEST application server allowing you to test your web application before putting new features into production,

2 - tfsProd.pjx for your production application server.

Build 2 application servers projects

This is a simple warning for your information; you can go ahead installing.

Building your Application Server projects encountered some trouble:
"Filecompile() - File 'tfsTest.pjx.prg' compiled successfully
Project 'C:\Program Files\VFP9\Tools\AB\AW\Samples\Tastrade\Sandbox\tfstest.pjx' could not be (re)built properly
Filecompile() - File 'tfsProd.pjx.prg' compiled successfully
Project 'C:\Program Files\VFP9\Tools\AB\AW\Samples\Tastrade\Sandbox\tfsprod.pjx' could not be (re)built properly"

You may (re)build your projects by running the following programs:
C:\Program Files\VFP9\Tools\AB\AW\Samples\Tastrade\Sandbox\tfsTest.pjx.prg
C:\Program Files\VFP9\Tools\AB\AW\Samples\Tastrade\Sandbox\tfsProd.pjx.prg

8/ Testing your Web application in http://localhost/…

Clicking 'Yes' will:

1- start your test FoxInCloud App Server in VFP IDE

2- Open your test site default home page.

8/ Testing your Web application in http://localhost/…

Note to trial version users:

Check VFP IDE for any wConnect splash form, click 'continue' if any.
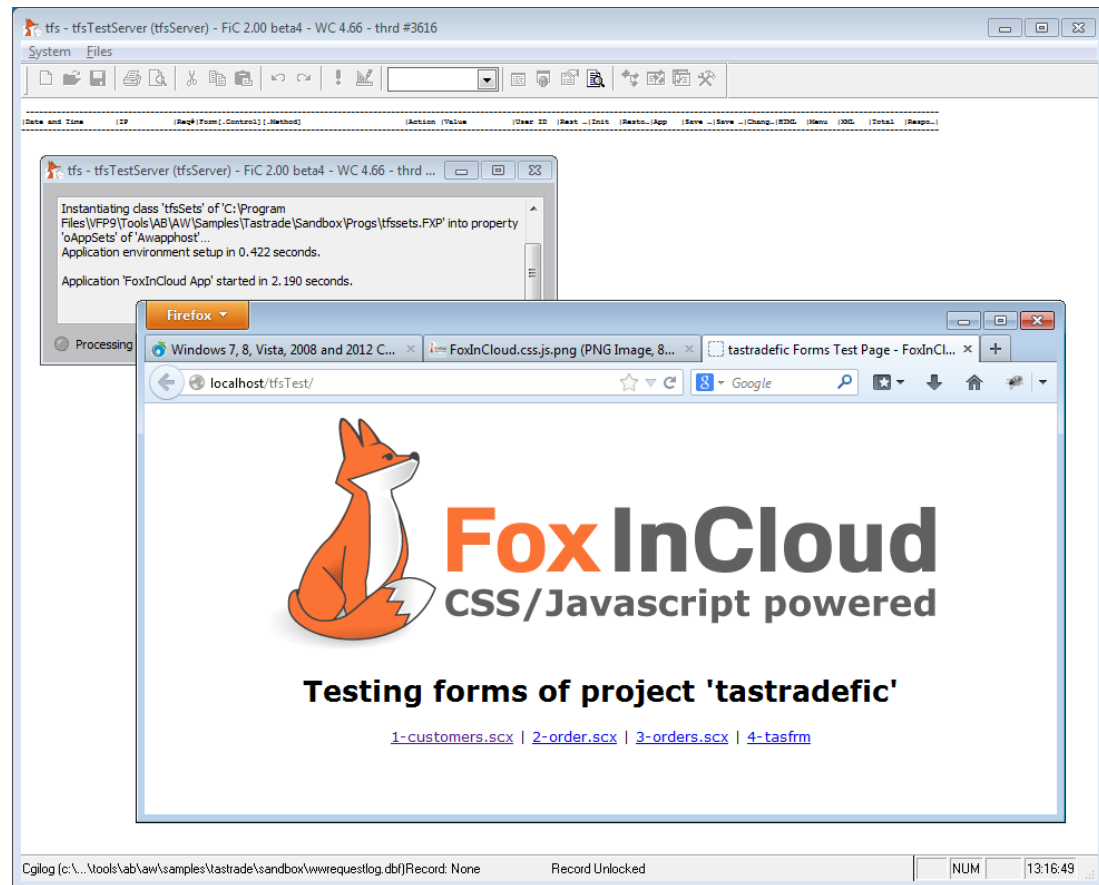
8/ Testing your Web application in http://localhost/…

Clicking the `customers.scx` link displays the customers form inside the browser

Note how the menu and form are similar to VFP originals …

Session 3, episode 2
Manual adaptations after step 3

A very last round of adaptations …

FAA has built `xxxSets.prg`, your app init. class, with whatever code found in your desktop app main.prg;

You now need to `review this generated code.`

Reviewing tfsSets.prg code

`tfsSets.Init()` fires when application starts, `tfsSets.destroy()` fires when it stops.

`tfsSets.wlLan` & `tfsSets.wlWeb` tell whether app runs in desktop or Web mode.

`tfsSets.Set()` performs SETtings and restores initial `Set()` when application unloads.

```
ABC tfssets.prg

* ----------------------------------------------

* Before this code executes, FoxInCloud Application Server has set CurDir() to
* the application's main folder, where project resides
* (C:\Program Files\VFP9\Tools\AB\AW\Samples\Tastrade\Sandbox\)
* All the following pathes are relative to this folder …

SET PATH TO "..\..\_DATA\;..\..\_LIBS\;.\CLASS\;.\FORM\;.\PROGS\" ADDITIVE

IF NOT InList(_VFP.StartMode, 2, 3, 5) && In COM automation mode, Visual FoxPro implicitly p

  SET PROCEDURE TO tastradefic ADDITIVE

  ClassLibAdd("tastradefic.vcx,tfs.vcx") && modify command abDev
  EXTERNAL PROCEDURE ClassLibAdd

ENDIF

this.Set("Status Bar", 'off')

if this.wlLAN
  set sysmenu save
endif

do foxyPreviewer.app
do tastradeFIC.mpr

return m.success

* ----------------------
PROCEDURE Destroy

= DoDefault()

if this.wlLAN
  set sysmenu to default
endif

removeProperty(_Screen, 'oFoxyPreviewer')
```

Choose your app home page

Go to your test app virtual, switch to contents view, select `index.htm`, switch back to features view, open 'HTTP redirect', click redirect to `index.tfs`.
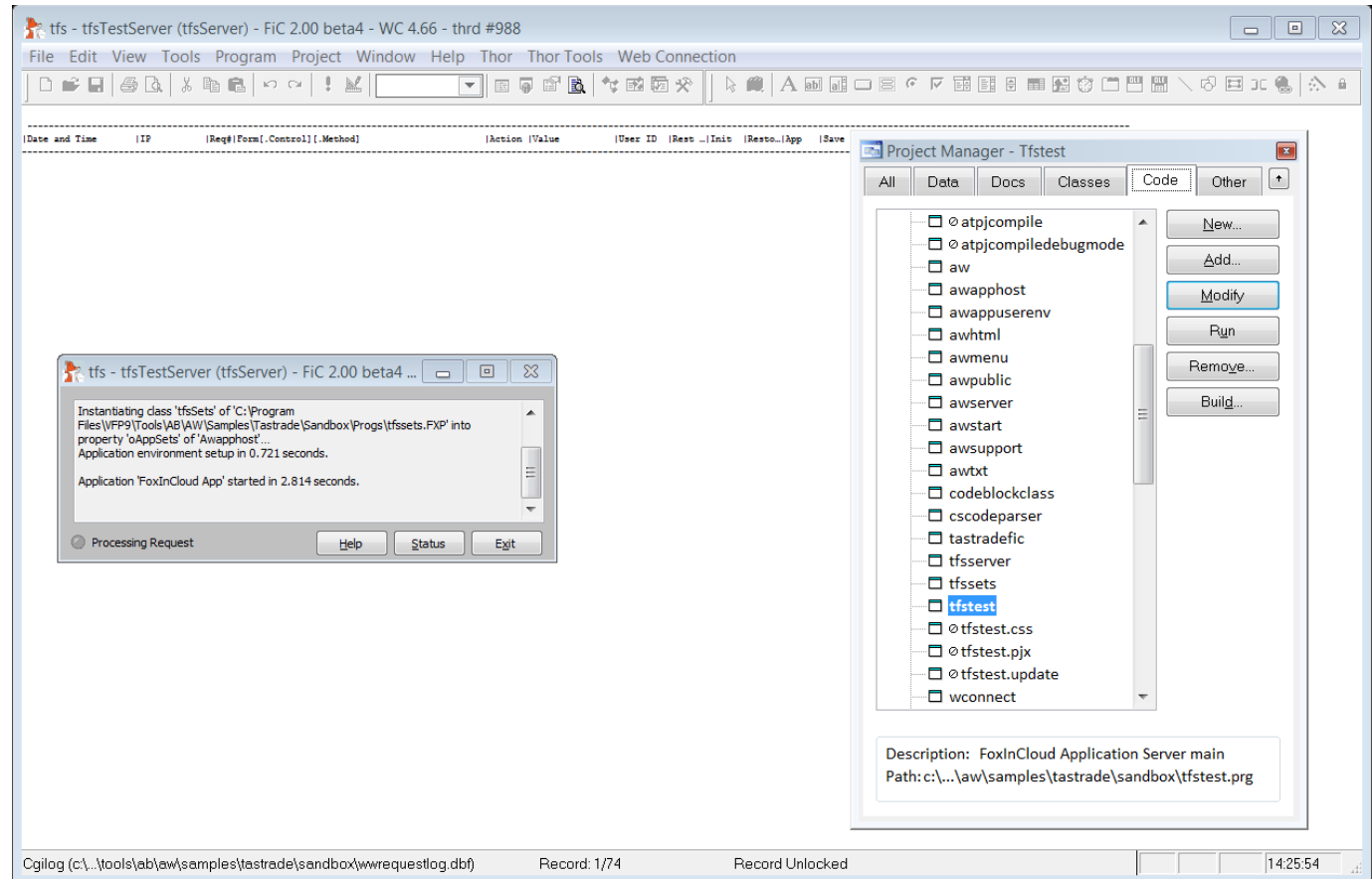
Starting your FAS in development mode …

In your app. folder, open your web project named tfsTest.pjx, and run the eponym tfsTest.prg

Displaying standard FoxInCloud home page

Your web app has now built the home page using FAS's standard `awProcess. index()` method.

You can override this code in your subclass `tfsServer.prg !tfsProcess`

Allowing anonymous access to app's forms

As this sample app is public, you need to set `.wlAnonymousAllowed` to .T. in you form class.

In a real-life application, only the login form should have this ability.

Displaying a form from the 'Files' menu

By default, FAA adapts `DO FORM` into `wForm();` this makes form to display as a child window of current page…

Displaying a form from the 'Files' menu in its own page...

Would you prefer your form to display in its own HTML page, just open your menu snippet and change `wForm()` into `wFormMaster()`, and discard parameter #2 (.F.)


```
      *-FIC- Replaced by FoxInCloud Adaptation Assistant versi
      *-FIC- do form customers


external form customers
customers = wFormMaster('customers.scx') && second parameter
```

Displaying a form from the 'Files' menu in its own page…

When your menu is open, click `menu > generate`, then `modify command tastrade.mpr` … you'll see the FIC menu generator in action …

Display the 'order' child form by double clicking an orders grid row

`gridOrders.dblClick()` calls `thisForm.wForm('order.scx', …):` order form displays in a child window within current page.

Setting the 'modal return' property in order form

For the order form `callback` to work, you need to set `.wcModalChoiceProp` to the name of the property holding the returned value, `'lEdit'` here.

Modal form call-back in action

Once `.wcModalChoiceProp` is set to `'lEdit'`, a message box appears whenever user modifies the order before closing the child form.
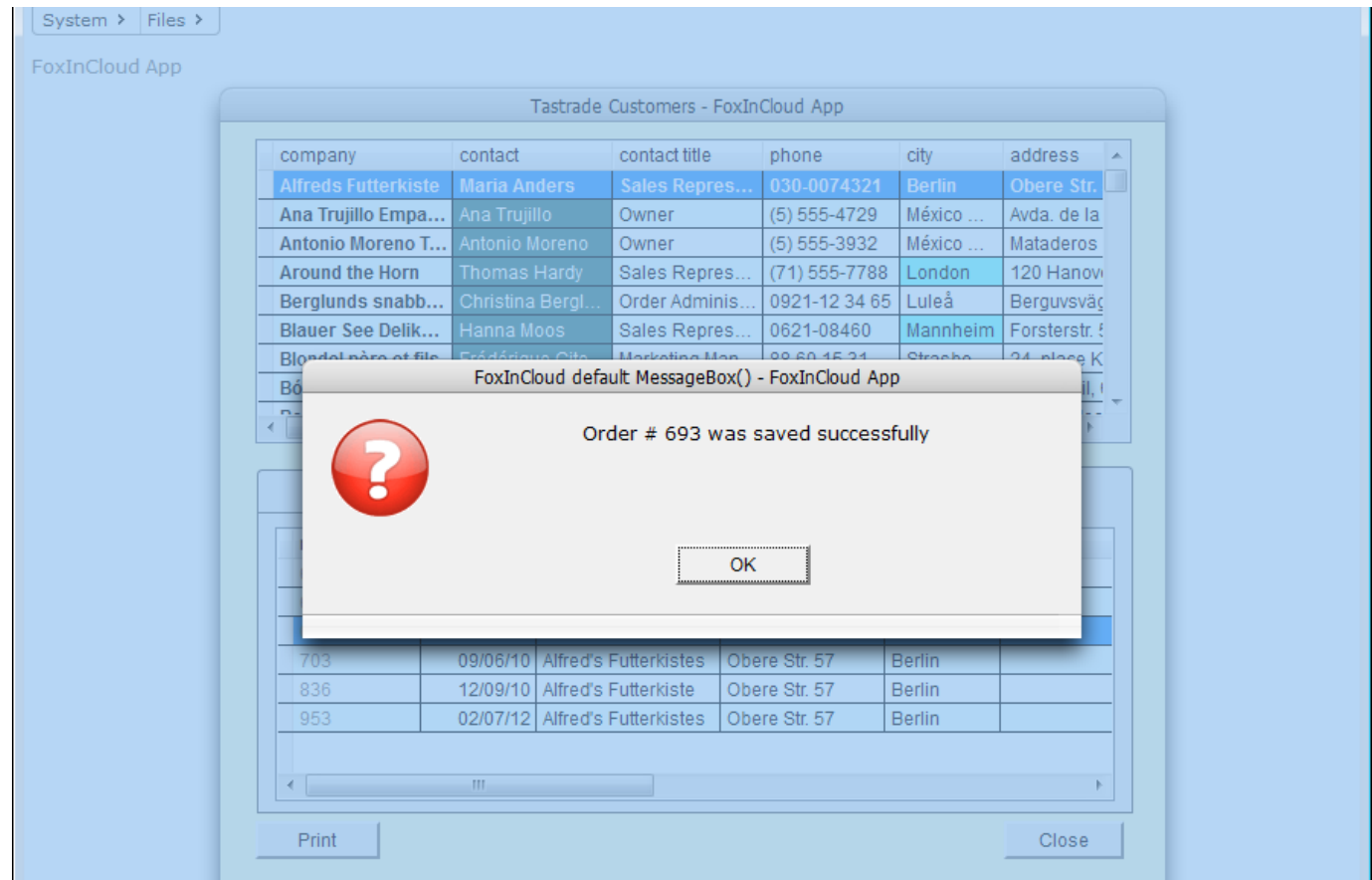
Reporting as PDF

Using `.wFileSaveAs()` opens a new tab in the browser, facing the standard popup blocker; this requires user's consent.

Reporting as PDF…

Once user allows popups on your web app, the PDF report displays in a new window.

Alternatively, you can display the pdf in a child form using your subclass of `aw.vcx!awolepdf*`

.MouseEnter() / .MouseLeave() effects using CSS

If you use VFP's `.MouseEnter()` `.MouseLeave()` for visual effects on hovered objects, you can easily achieve the same result using CSS

```
 1   /*
 2   This your application's style sheet.
 3   By default, [see awServer.prg!awProcess::cawCSSinc()], FoxInCloud loads this styleSheet after
 4   javascript components' CSS, and default CSS generated from your application's classes
 5   and forms: [awDefault.css]
 6   You may add any CSS directives to override styles inherited from you VFP application's
 7   visual objects.
 8   */
 9
10   /* this CSS replaces the desktop .MouseEnter() and .MouseLeave() */
11
12   #customers_scx-pgf-pagdetails input:hover, #customers_scx-pgf-pagdetails select:hover {
13       background-color: rgb(190,190,190);
14       border-color: transparent;
15       box-shadow: 2px 2px 2px #535353;
16   }
17
```

.MouseEnter() / .MouseLeave() effects using CSS…

Adding the proper CSS directives to your app's css (`tfs.css` here) produces a cool backcolor change when hovering controls.

Session 3, episode 3
Getting prepared for production

Completing this section requires a full <u>FoxInCloud</u> license.

For more information about this license, please browse
http://foxincloud.com/pricing.php,
and/or contact sales@foxincloud.com

Compiling your app in production mode

Running `atPJcompileDebugMode.prg` toggles application between dev and production modes

Running your Application in production mode

In production mode, request log and forms won't appear in `_screen`, and errors are captured by a global handler

Building your application.exe

Like for your desktop app., you distribute your web app. as an executable

Build a COM server (exe)
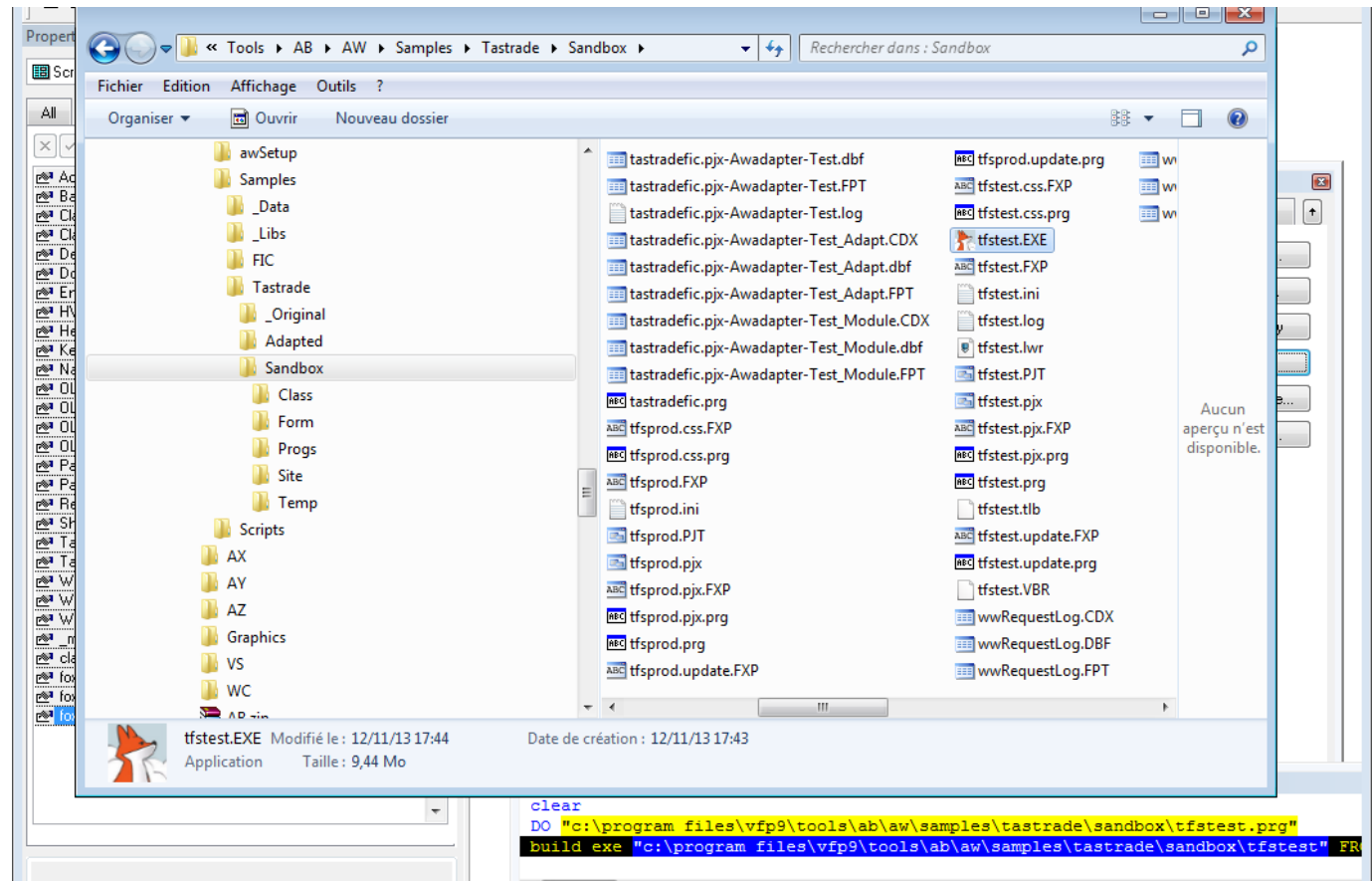
Build your global default CSS (awDefaultAll.css)

Running `xxxTest.css.prg` builds your app's global CSS.

`xxxTest.css.prg` sits in your project's home folder, next to `xxxTest.*`

```
2   This is the standard style sheet generated as of your VFP application's classes and objects
3   hierarchy.
4   To get the CSS definitions of all class and objects existing in your application, run
5   xxx[Test|Prod].css.prg, or start your FoxInCloud server in file mode with the first parameter set
6   to .T.
7   In this case, the stylesheet generated is named 'awDefaultAll.css', otherwise 'awDefault<Sys(2015)>.
8   css' dynamically refreshed.
9   To make your Web application look and feel different from its VFP 'ancestor', you can simply
10  override any of these CSS definitions in your application's Style Sheet 'xxx.css' (always loaded
11  last).
12  -- */
13
14
15  /* -- General (W3C CSS 2.1 default) -- */
16  * {padding:0; margin:0;}
17  img {border:none;}
18  table {border-spacing:0;}
19
20  /* -- Classes as 'Checkbox' -- */
21  .checkbox {color:#000000;height:17px;margin:0;font:normal normal 9pt Arial; text-decoration:none;pos
22  .checkbox.standard {background:#FFFFFF;}
23  .awchk {}
24  .awchk.standard {}
25  .tfschk {}
26  .tfschk.standard {}
27  /* -- / Classes as 'Checkbox' -- */
28
29  /* -- Classes as 'Combobox' -- */
30  .combobox {background:#FFFFFF;border:1px solid #646464;color:#000000;padding:2px 2px;width:94px;heig
31  .awcbo {}
32  .tfscbo {}
33  /* -- / Classes as 'Combobox' -- */
34
35  /* -- Classes as 'Commandbutton' -- */
36  .commandbutton {background:#F0F0F0 no-repeat center;padding-top:0px;color:#000000;width:98px;height:
```

Any question? Post with screenshot in the 'FoxInCloud' section of http://west-wind.com/wwThreads/ or http://www.universalthread.com/
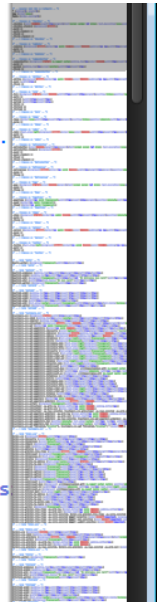
Running your Web app in COM mode

Open your wConnect admin page, switch to COM mode and load server.

(browse [wConnect on line documentation](#) for details)

Running your Web app in COM mode …

As server auto-starts In COM mode, and response times are twice faster than in file mode,
COM is the recommended mode for production.

Browsing your server log

From your wConnect server admin page, you can browse the requests against your web app.

Web Connection Log

File: wwRequestLog.DBF
Total Log: 110
Today: 110
Displaying last 110 Entries

Time	Script	Ip	Duration	Memory
12/11/2013 18:00:32	propch.tfs?&Prop=activepage&ObjAddr=customers_scx-pgf&PageInit=&UserID=3X7128VYQ&Value=2&ValueType=number&nReq=	::1	0,78	3657664
12/11/2013 18:00:31	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=ALFKI&V	::1	0,88	3660076
12/11/2013 18:00:30	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=ANTON&V	::1	0,98	3660284
12/11/2013 18:00:29	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=BERGS&V	::1	1,27	3660508
12/11/2013 18:00:27	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=BOLID&V	::1	0,85	3659628
12/11/2013 18:00:26	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=BLAUS&V	::1	0,93	3659488
12/11/2013 18:00:25	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=AROUT&V	::1	1,56	3659688
12/11/2013 18:00:23	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=BLAUS&V	::1	1,01	3659616
12/11/2013 18:00:22	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=ALFKI&V	::1	1,22	3659756
12/11/2013 18:00:21	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=ANATR&V	::1	1,24	3659556
12/11/2013 18:00:19	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=AROUT&V	::1	1,23	3659680
12/11/2013 18:00:18	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=BOLID&V	::1	0,71	3659456
12/11/2013 18:00:17	gridmethod.tfs?&MethAddr=wAfterRowChange&ObjAddr=customers_scx-grdcustomer&PageInit=&UserID=3X7128VYQ&Value=BERGS&V	::1	0,86	3659148

FoxInCloud

A-Z tutorial, Session 4

Requires FAA & FAS V2.0+

Session 4
Deploying for production

Completing this section requires a full <u>FoxInCloud</u>, West-Wind Web Connect, and ActiveWidgets license.

For more information about these licenses, please browse http://foxincloud.com/pricing.php, and/or contact sales@foxincloud.com
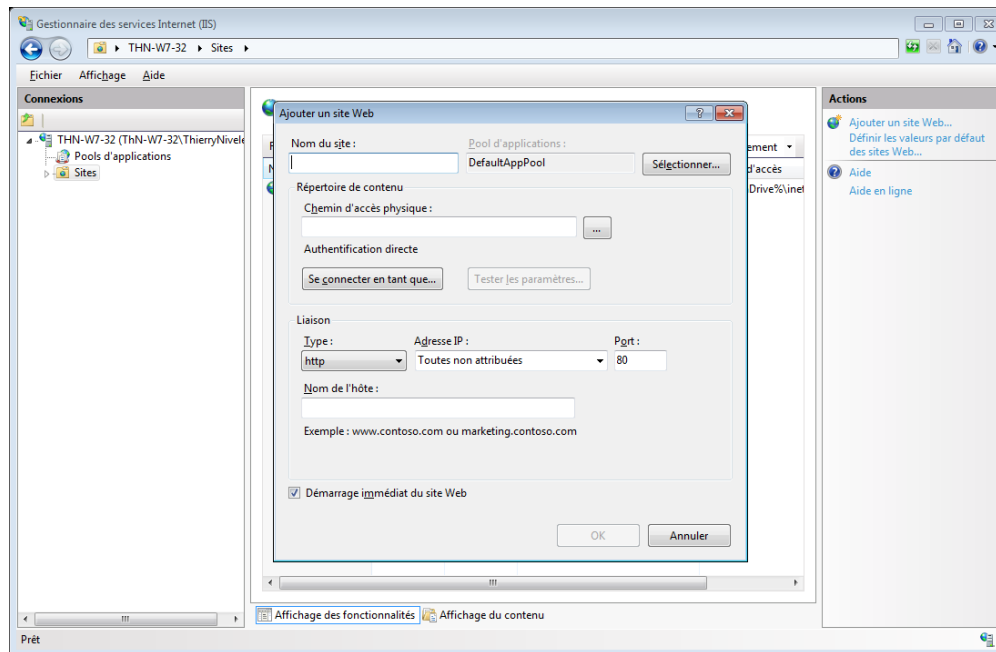
Deploying for production, Agenda

Episode	Subject	Slides
1	Creating your Web App Site	14
2	Creating your Web Application	9
3	Linking Web App Site and Web Application	12
		35

Session 4, episode 1
Creating your Web App. Site

If your server runs IIS >=7, check your IIS installation

If your production server runs IIS 7+ (Windows server 2008+), check your IIS installation as you previously did in FAA step 3-publish.
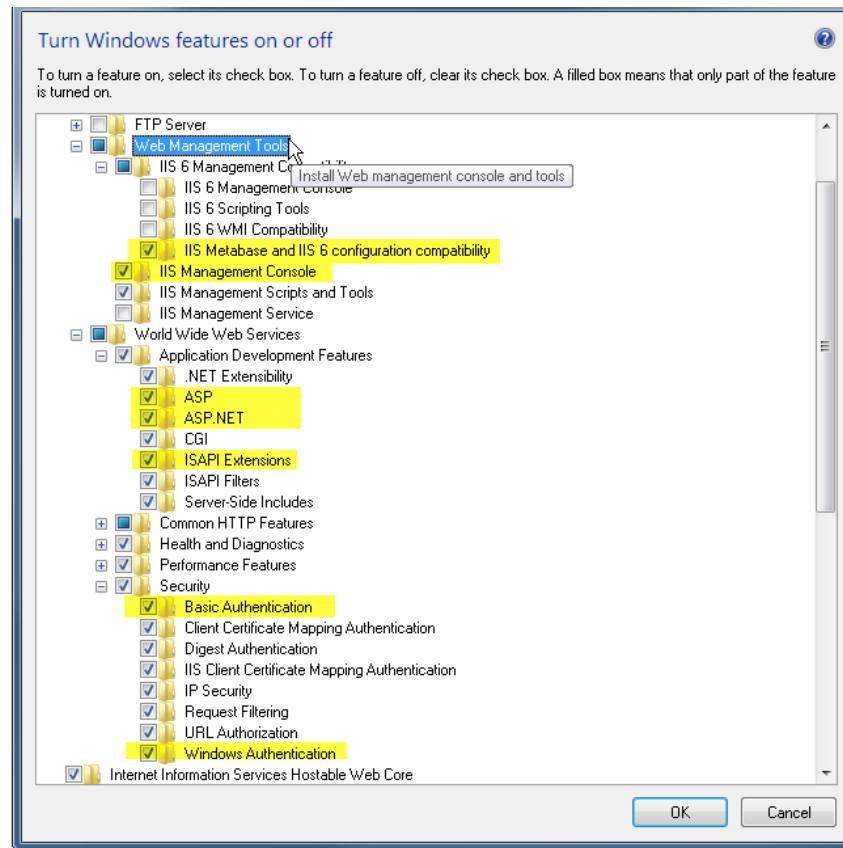
Copy site files from dev machine to prod. Server

Source files are in your project's `Site\` folder.

Create a new sub-folder in `inetpub\` and copy your files there.
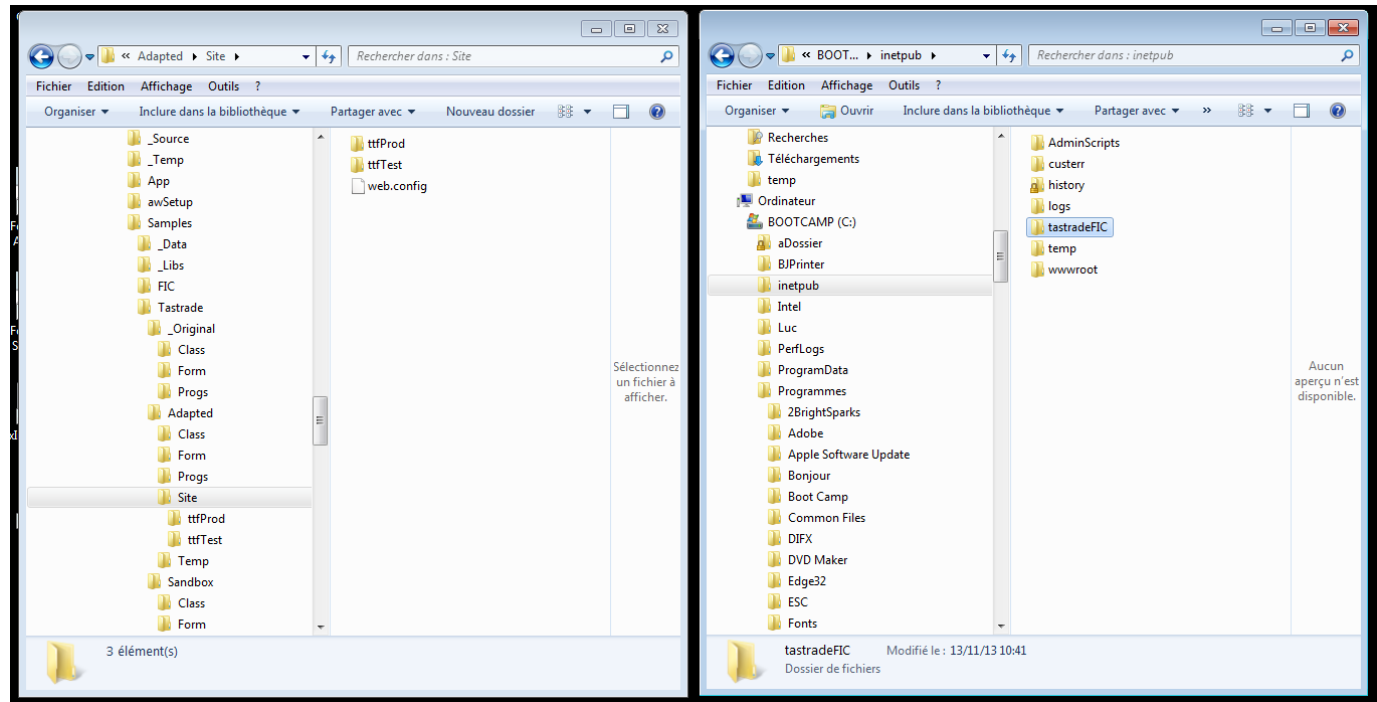
1 site for tests
1 site for production

As in your dev. environment, you will create 2 sites on target production server, one for test and one for production.
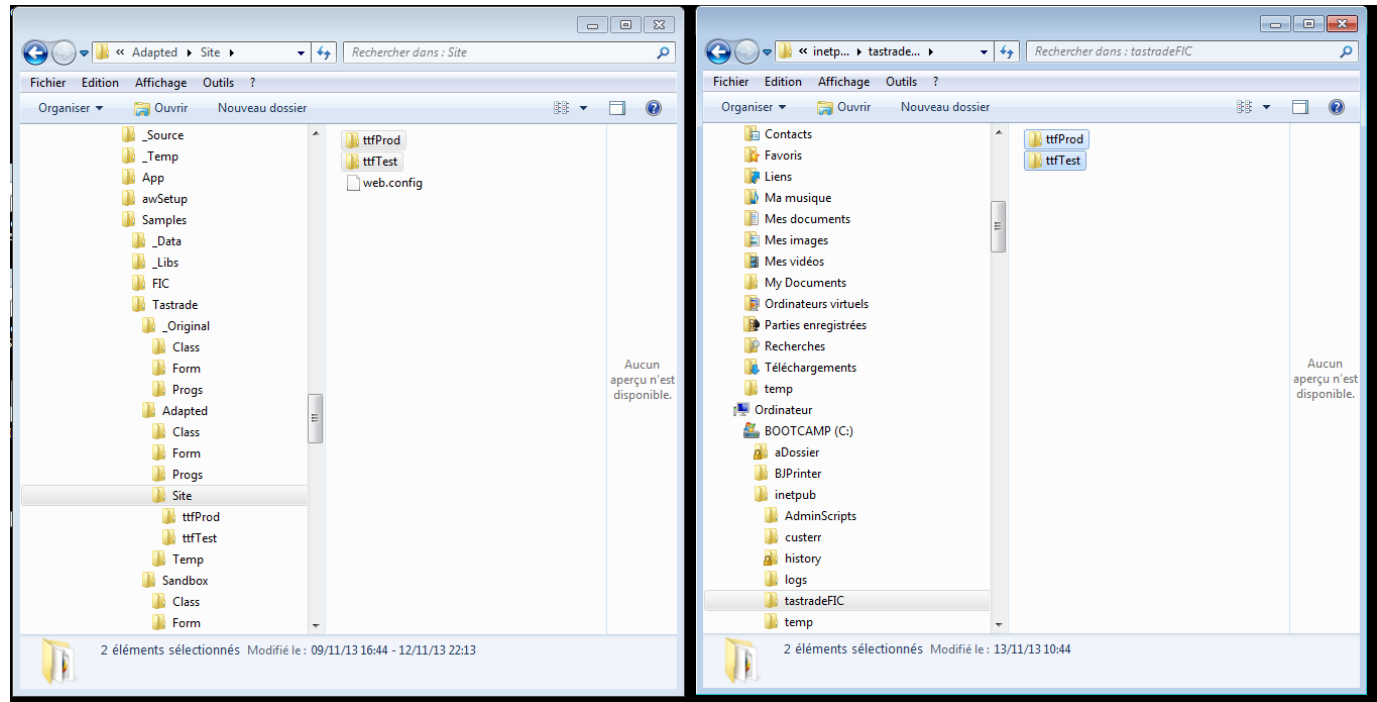
Delete copied web.config files

Make sure to delete the `web.config` files as they contain addresses from your dev. machine that are irrelevant on your prod. server.
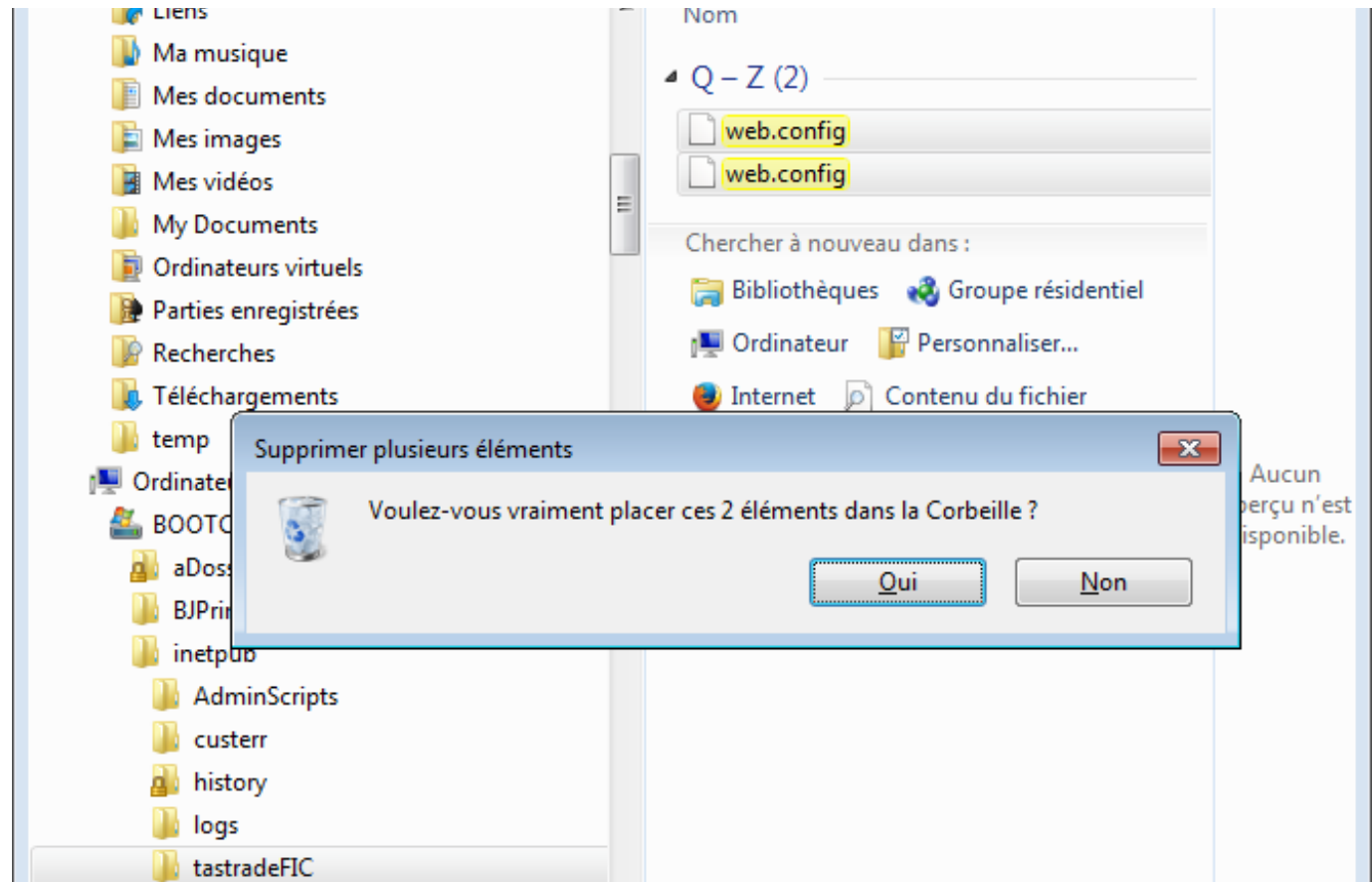
Create a pool for FoxInCloud applications

(IIS >=6)
At your server root, add an application pool with the properties pictured here.

Create your IIS Web Site

From the server root > sites, add a site using the FoxInCloud app. Pool.

For simplicity, name it according to your source project (`xxxTest`|`xxxProd`)
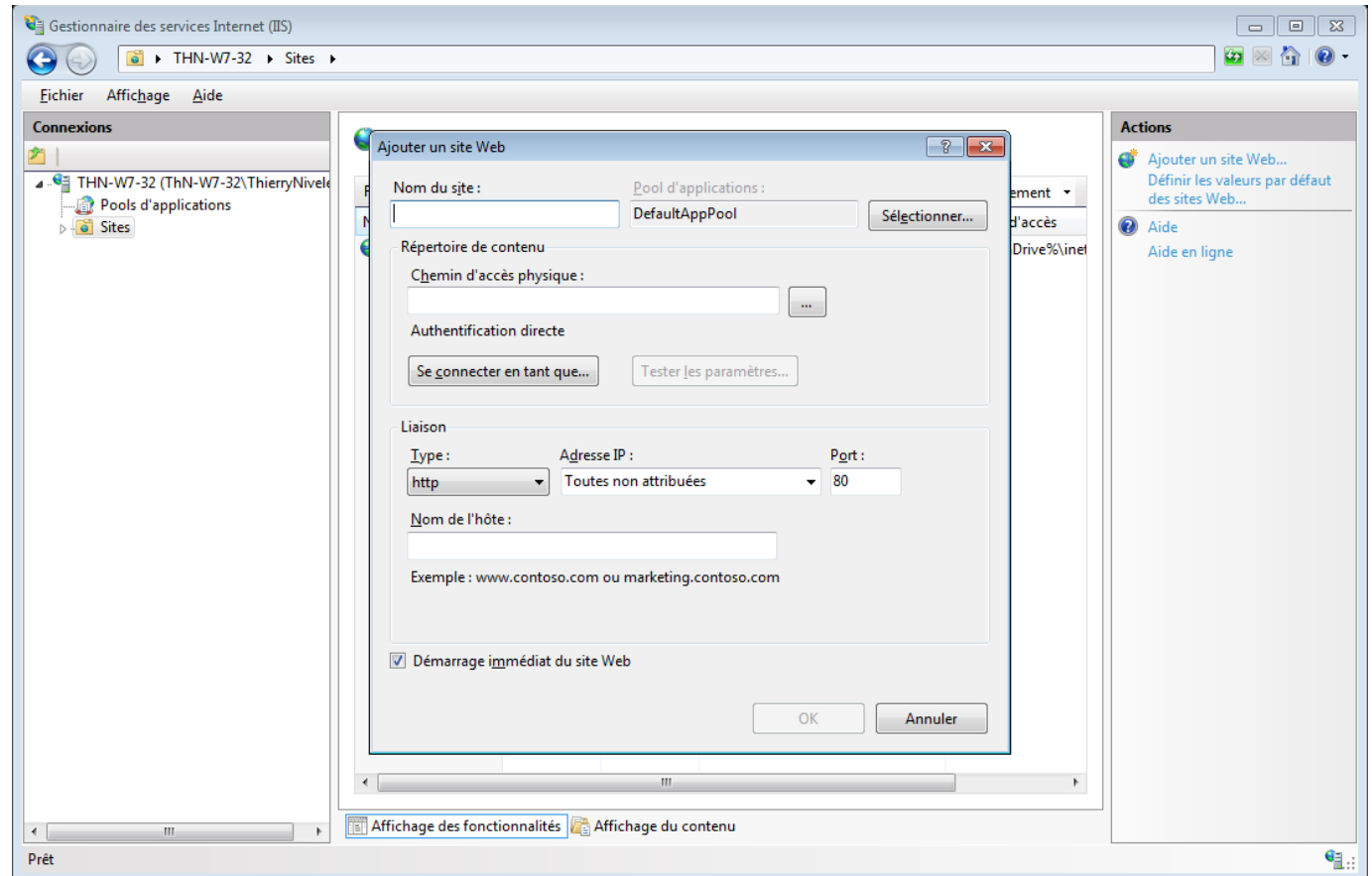
Point site to the folder you just copied

Site root folder is one of the 2 sub-folders you've just created in `inetpub\`
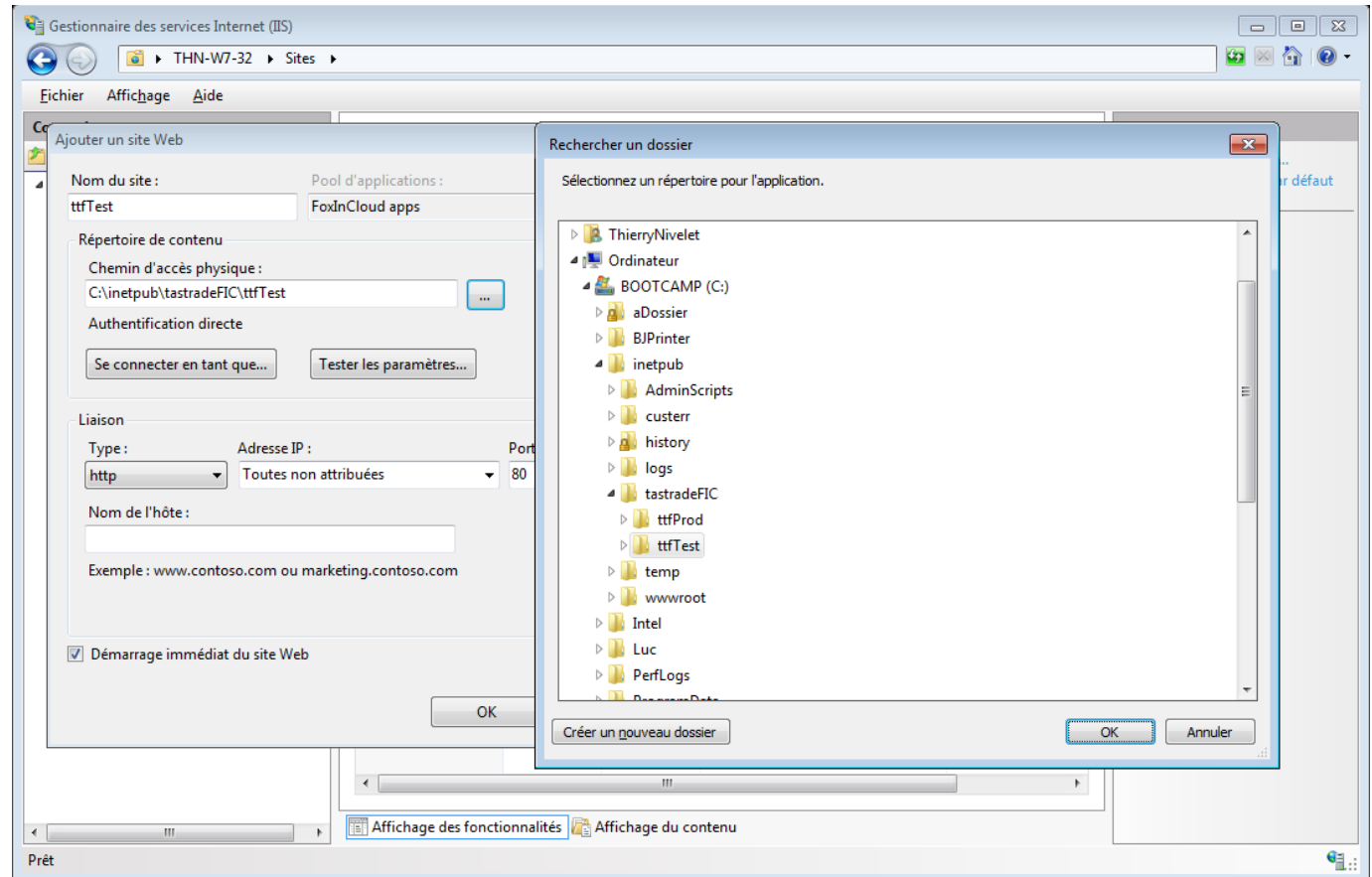
Site is created

New site shows the folder contained in inetpub sub-folder.
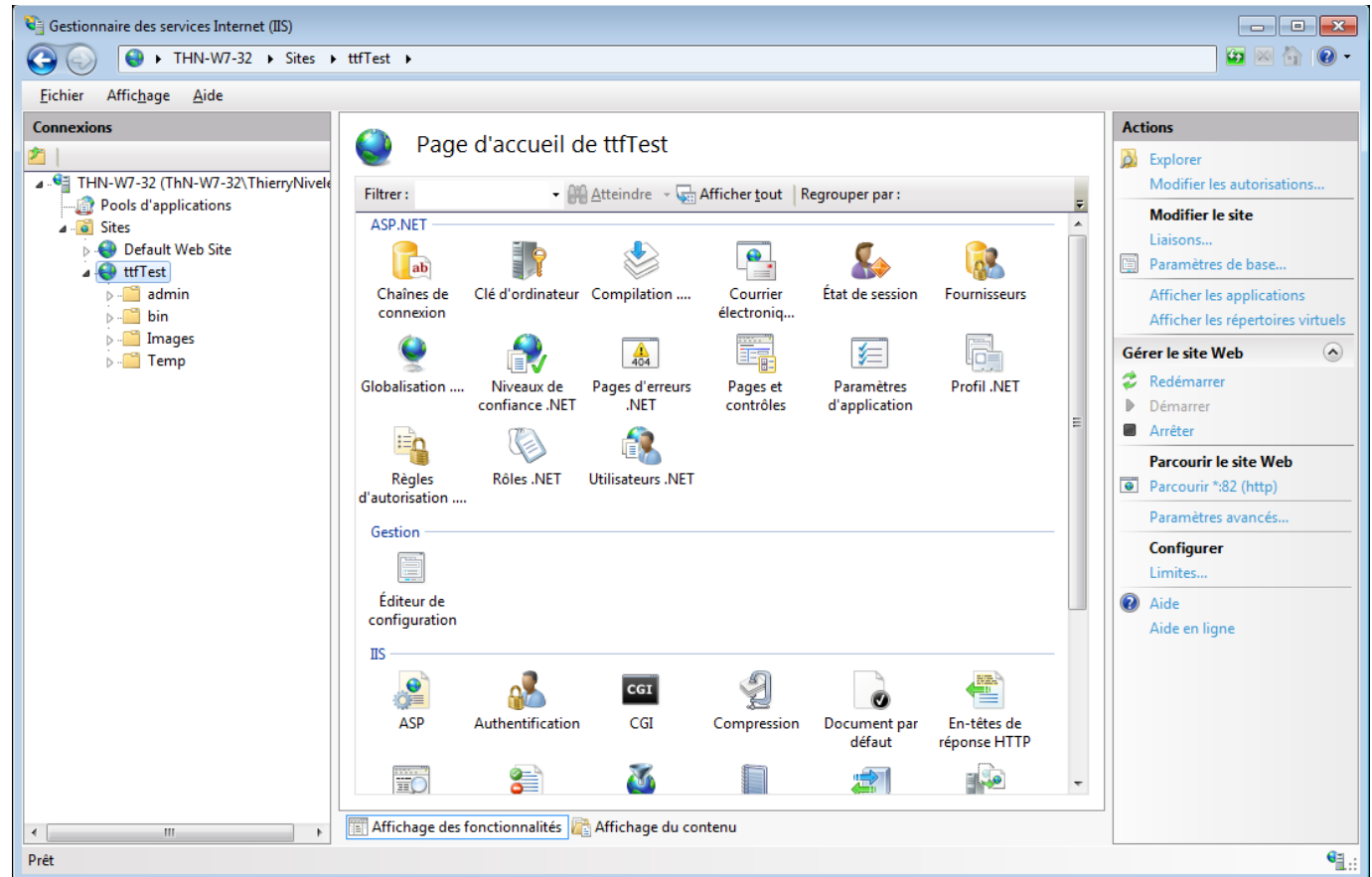
Site responds
(default home page)

Default home page is `index.htm` that FAA has build for you in step 3-publish;

(will be replaced in a next step)

Create FoxInCloud script folder

In
`inetpub\`,
add a
folder
named
`awScripts`.

Following FiC's naming convention, 'awScripts' means 'standard FoxInCloud CSS and JS'
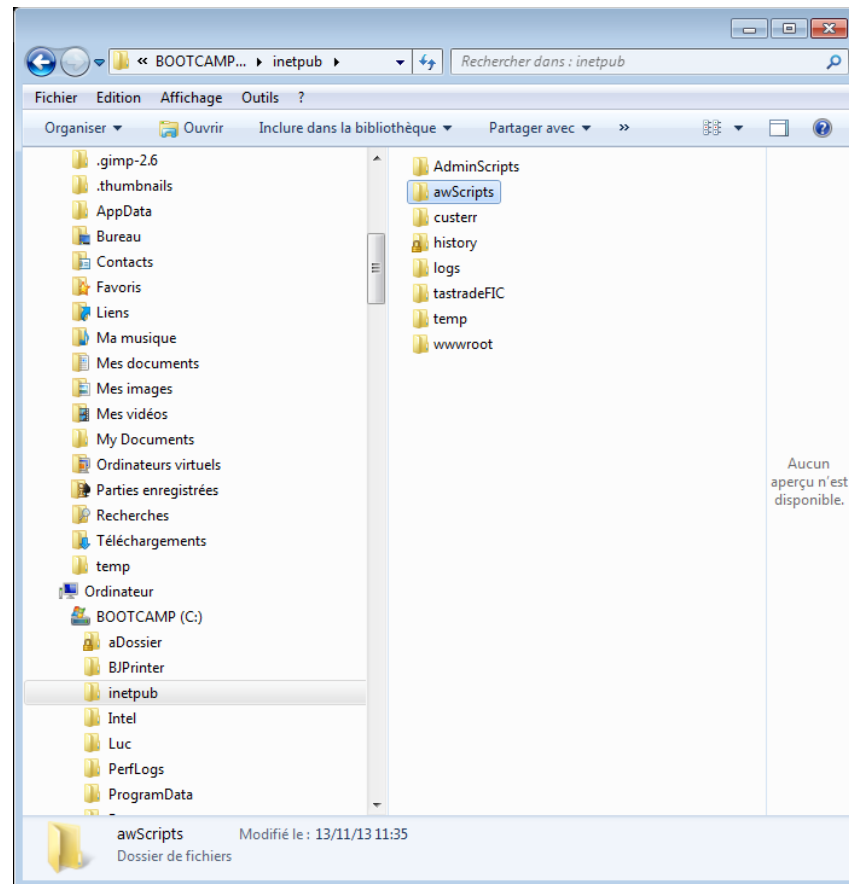
Copy FoxInCloud `*.css` & `*.js` into `inetpub\awScripts\`

Copy the contents of
`<VFP9>`
`\Tools\AB`
`\AW`
`\Scripts\`
from your dev. machine to your server,
EXCEPT
_Source
folder containing © code.
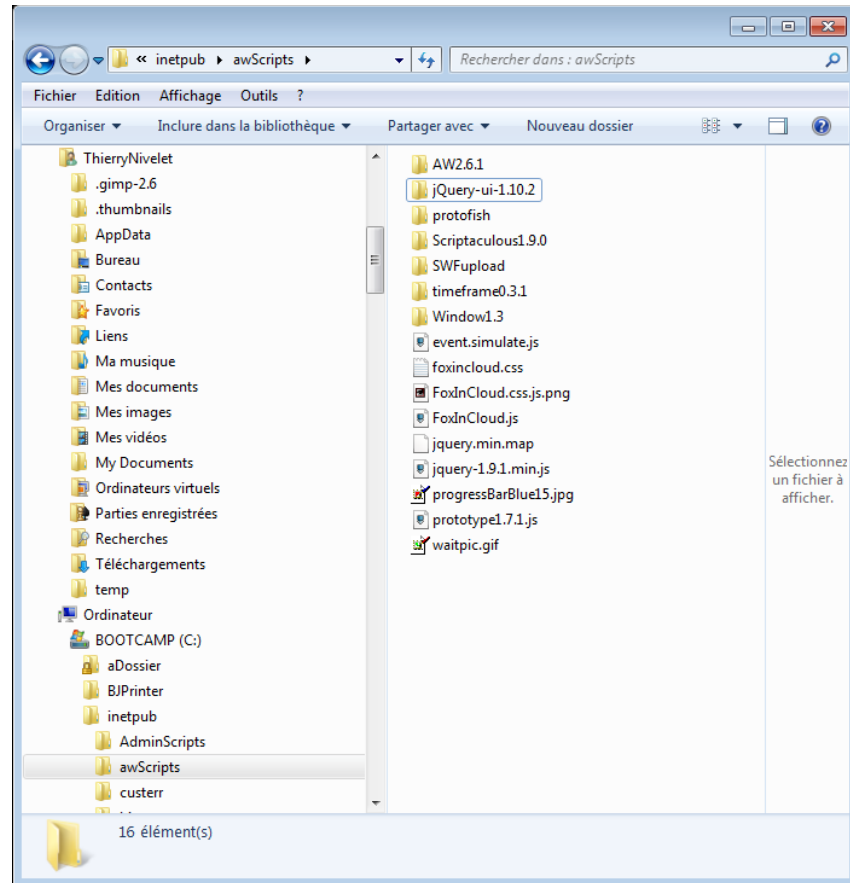
Create /awScripts/ virtual directory at site root

Create a virtual directory alias `awScripts` at the root of your site.

(though you can configure another address, this default location is recommended)
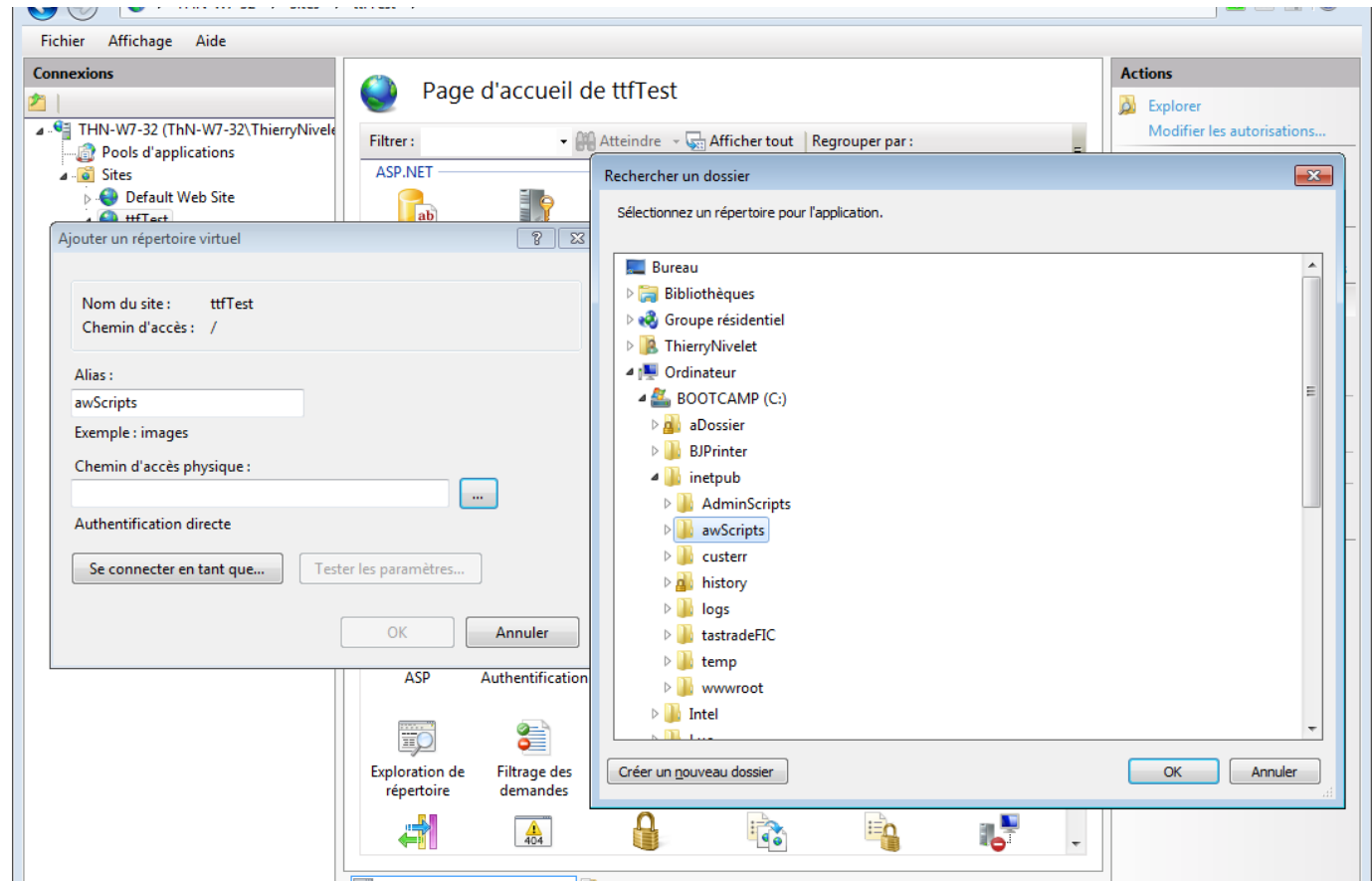
Allow requests on /bin/*

At your server or site root, open the 'request filtering' feature and remove '`bin`' from the list of hidden segments.
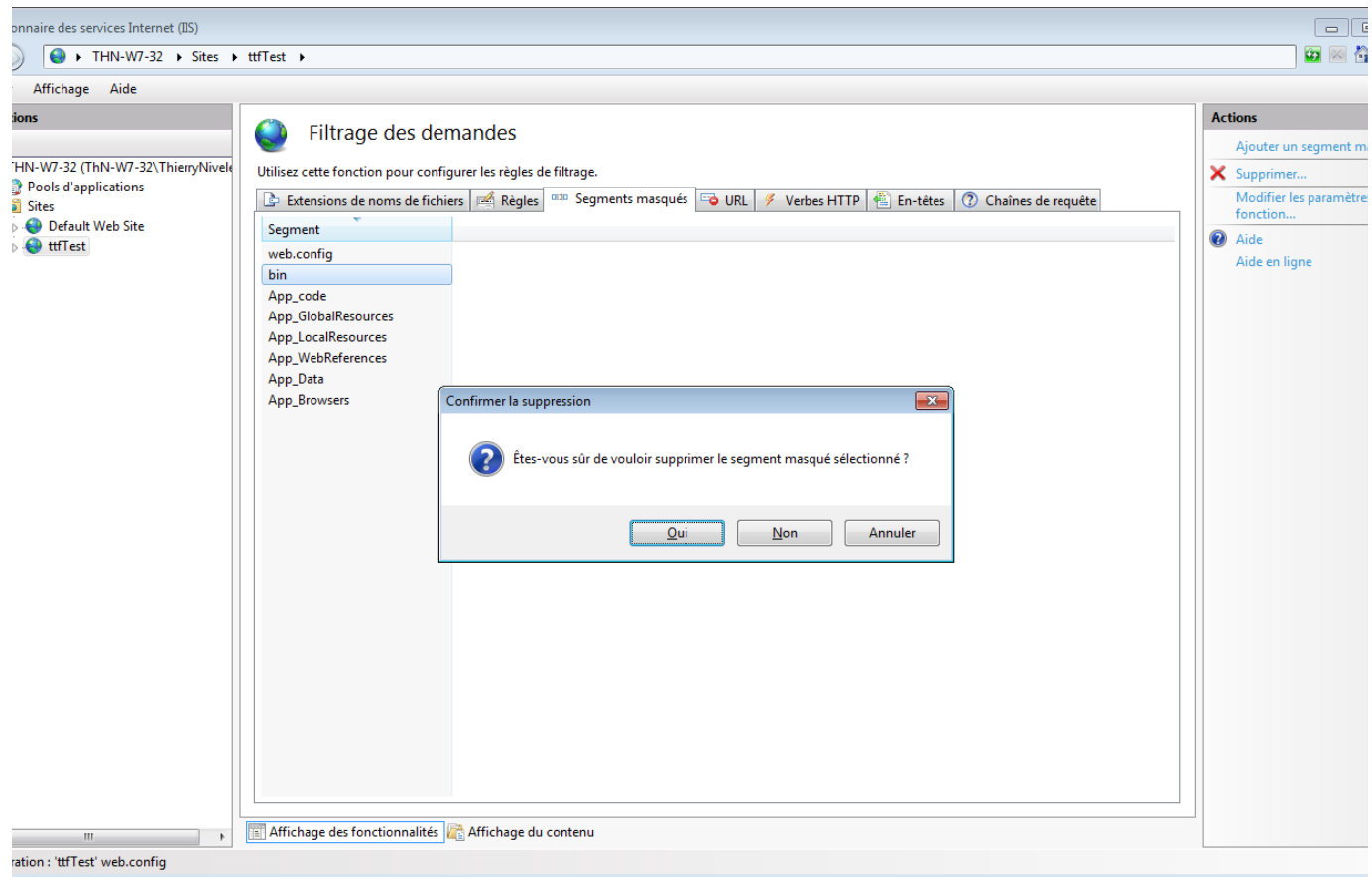
Session 4, episode 2
Creating your Web Application
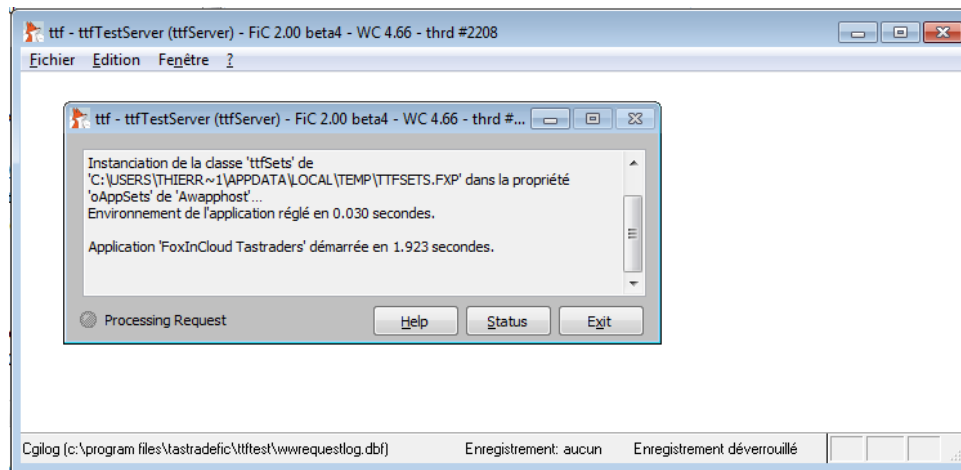
Copy program files

Copy your VFP `*.exe / *.ini`, a couple of `*.dll`, your app resources (data, images, reports, …) and a `Temp\` folder.
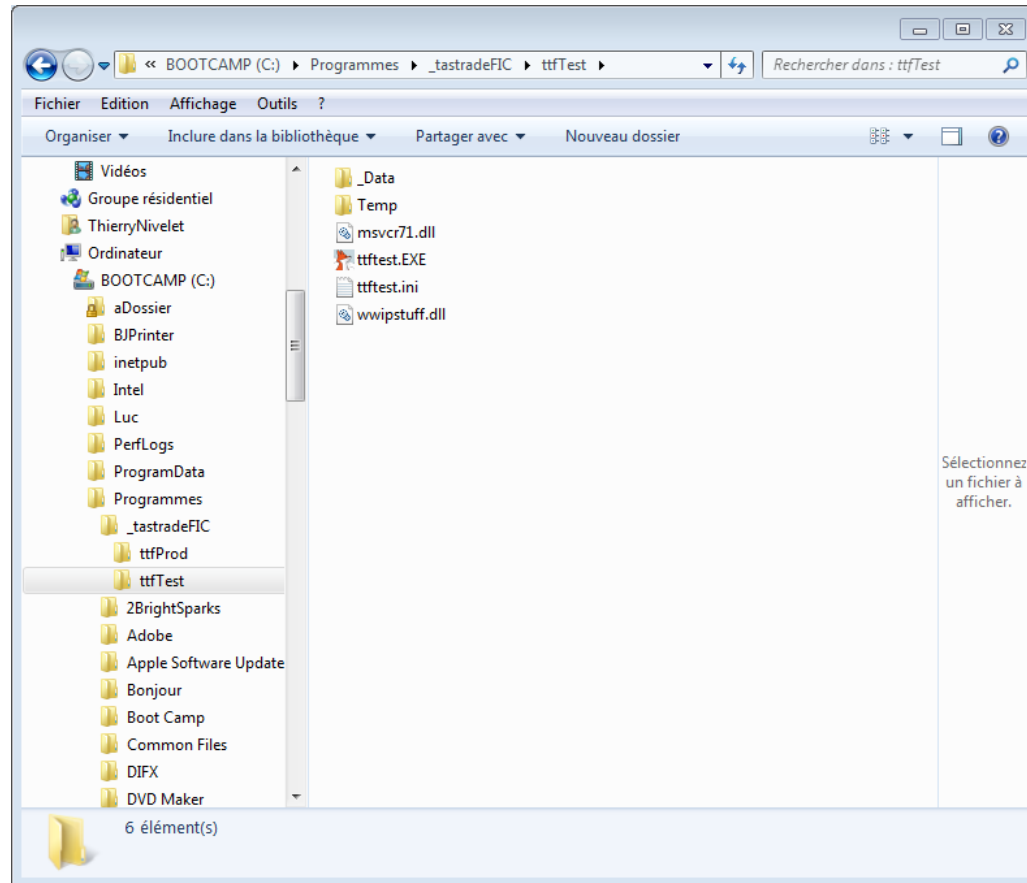
Install VFP9*.dll

You can deploy `VFP9*.dll` either your application's folder, on in the standard shared folder (recommended)
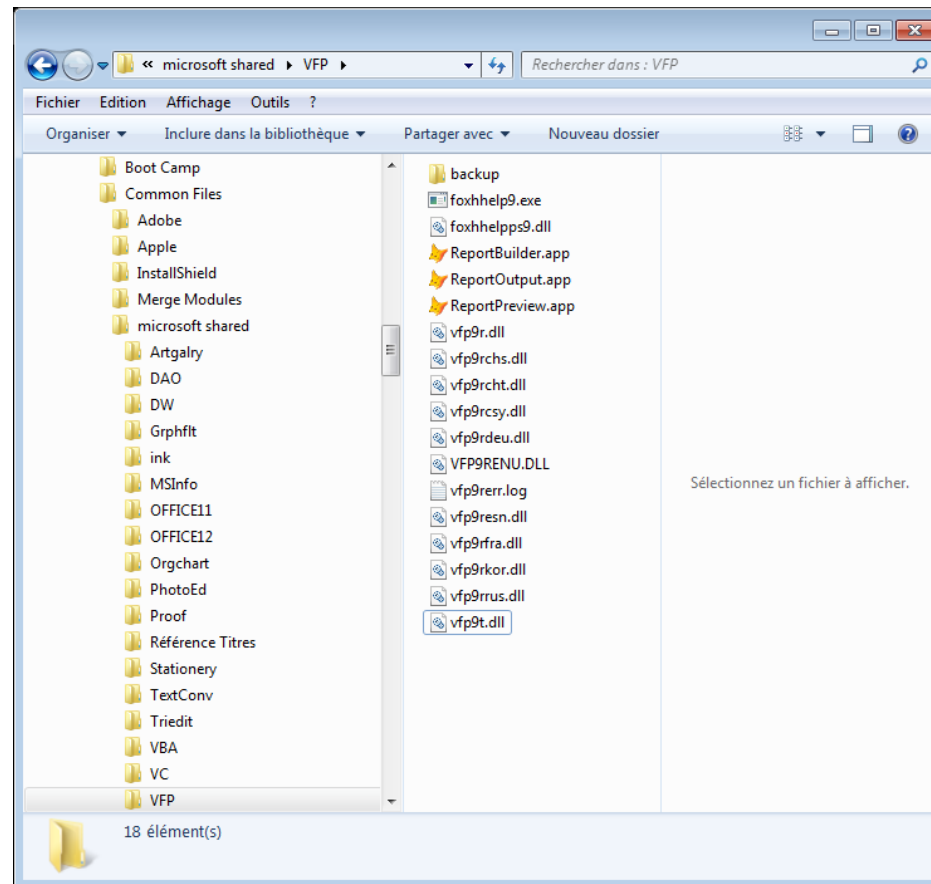
Register your VFP exe as a COM server

Run
`your.exe /regserver`
to register your app's COM object.

(on your dev machine, VFP auto registers your COM server upon each `BUILD EXE`)

Create a dedicated user

Each instance of your web app. should run under a dedicated user account named similarly.
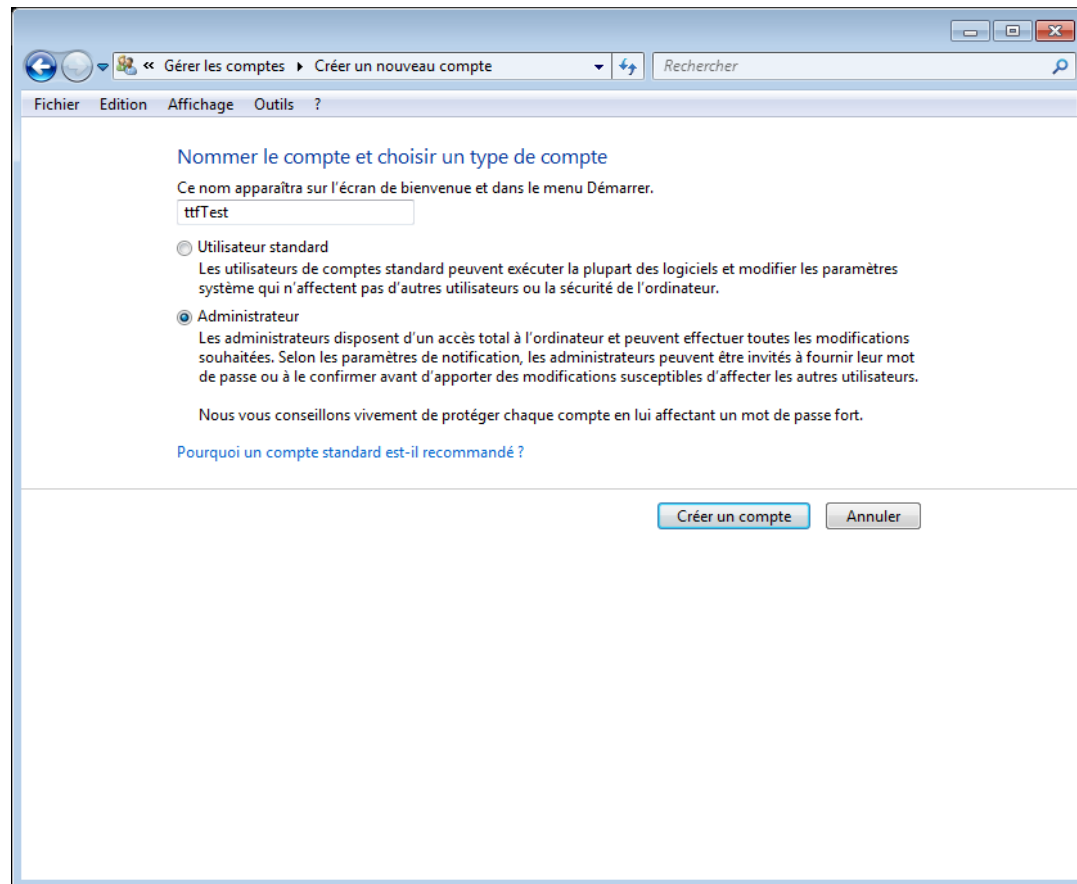
Open a session for this dedicated user

Opening a session ensures the creation of a workspace & temp folder where the app. will work into.
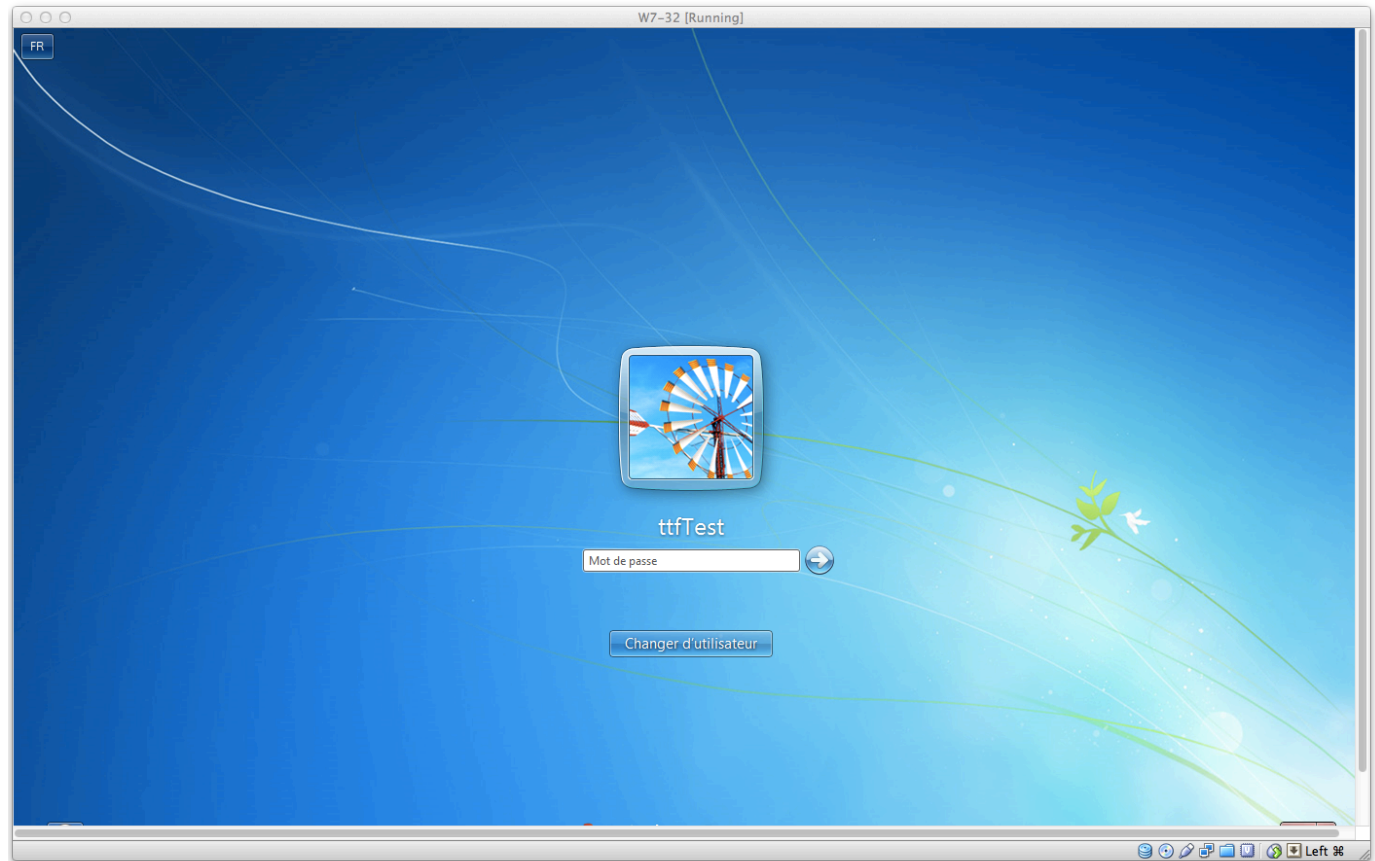
Dedicated user's workspace is created
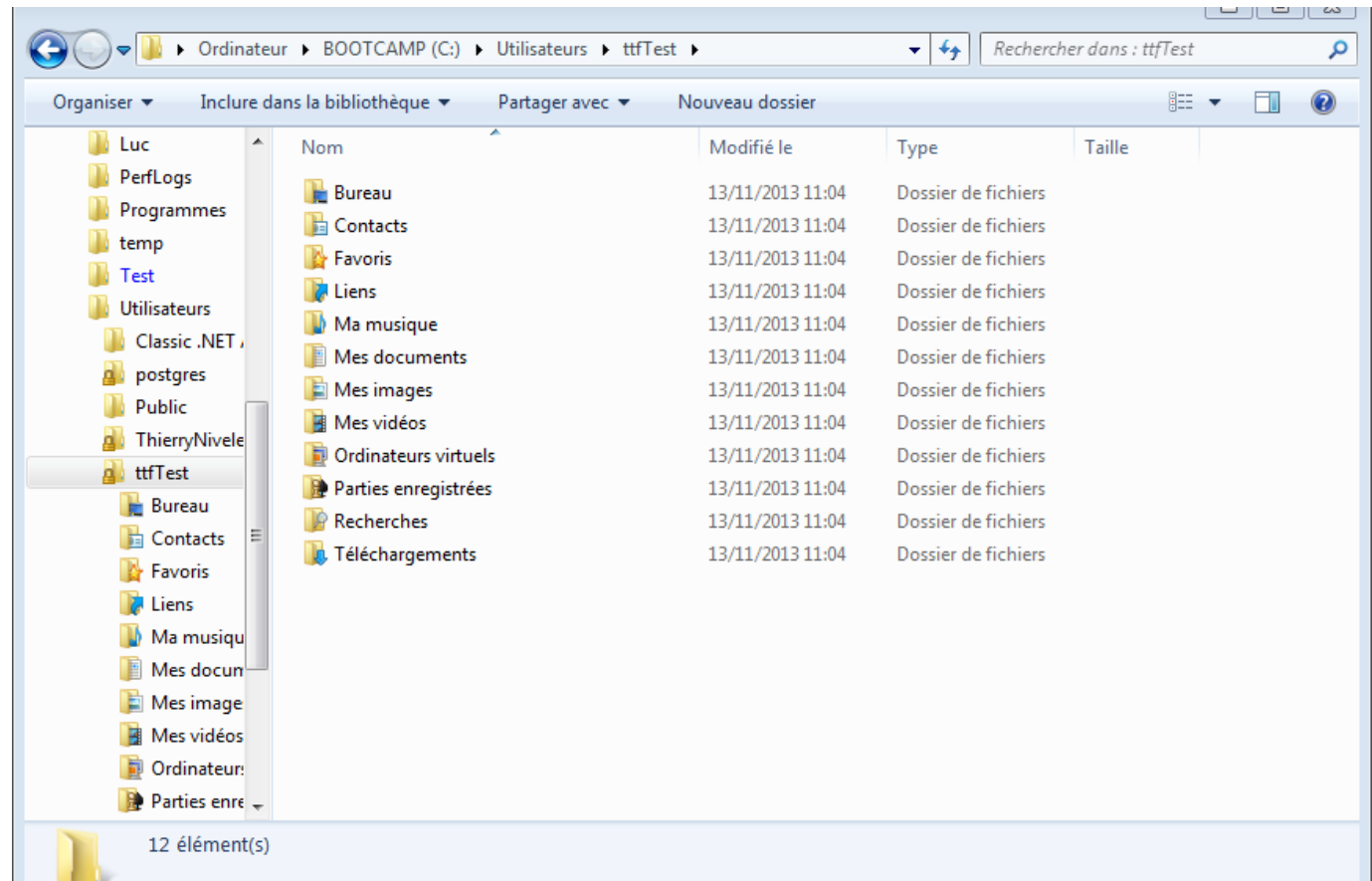
Get back to your session
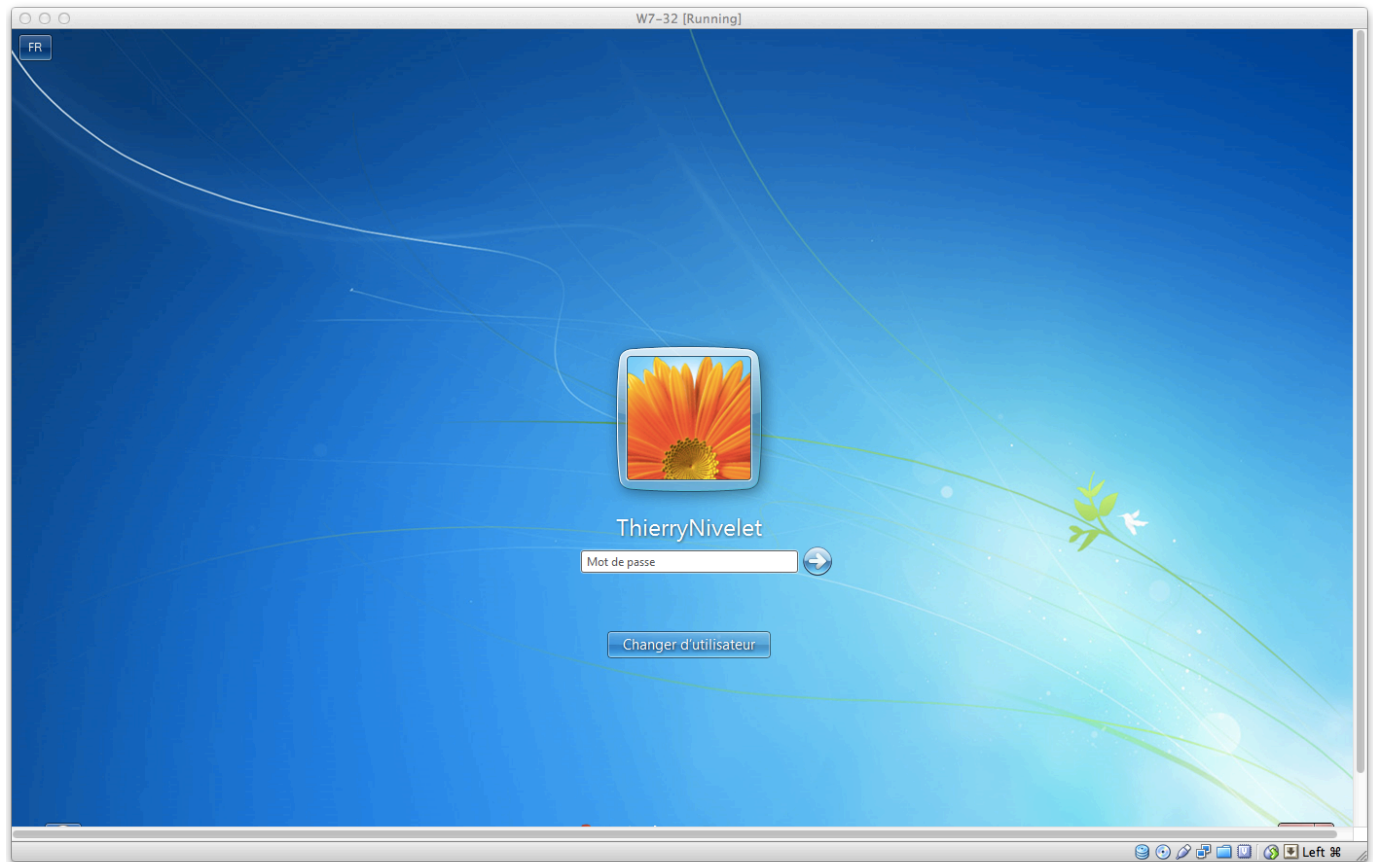
Set your app COM object to run as the user just created

Run `dcomcnfg`, select your COM server and set identity to the user you just created.
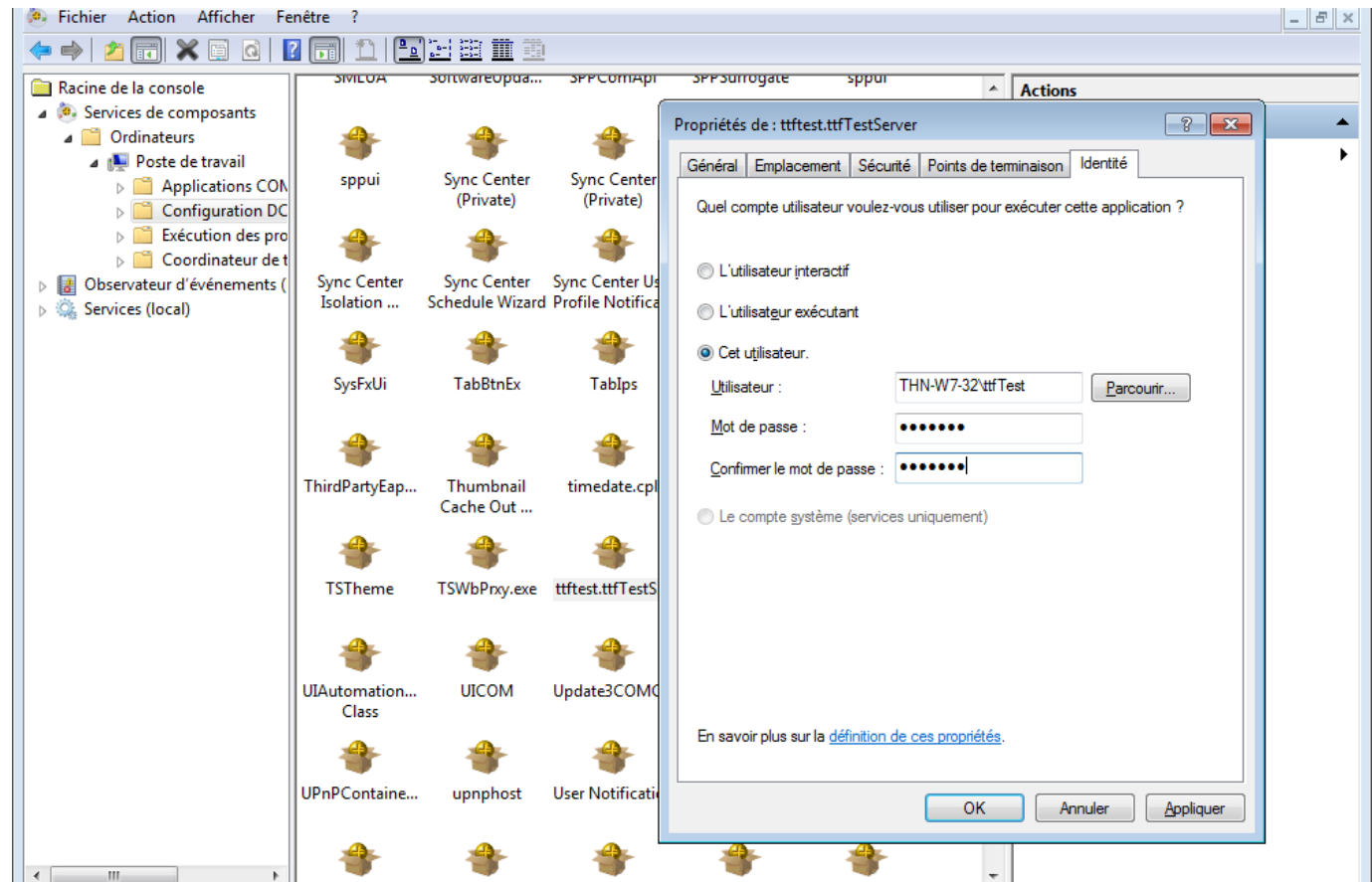
Session 4, episode 3
Linking Web site and Web app

Adjust contents in wc.ini & xxxTest|Prod.ini

Adjust the the various pathes to your production server install, fill out your customer, admin and FAS IDs and keys


```
File  Edit  Selection  Find  View  Goto  Tools  Project  Preferences  Help

 ttftest.ini          x      wc.ini              x

 1   ; .INI FILE FOR WC.DLL
 2   ; ==========================
 3
 4   ; wc.dll reads this file once at startup, and each time you press the
 5   ; 'Update DLL INI File' button in the 'Editing Config Files' form
 6   ; (admin.asp > 'Edit configuration files' .../wc.dll?wwMaint~EditConfig)
 7
 8   ; Comments in this file originate partly from web connect documentation,
 9   ; partly from FoxInCloud to ease server setup. You obviously can delete or
10   ; simplify these comments when your application goes into production.
11
12   ; wConnect documentation about this file is titled "The wc.ini configuration file"
13
14   ; Everywhere in this file, you need to replace 'ttf' by your application
15   ; code as explained in the FoxInCloud documentation:
16   ; http://foxincloud.com/How-to.php?HowTo=awServerDev
17
18   ; ttfTest.exe means either the Test or Production version of
19   ; your FoxInCloud Server.
20
21   ; 'Test' and 'Development' are 2 different concepts for FoxInCloud:
22   ; you should have a Test server in your production environment to test
23   ; changes in your application before putting them into production for real users.
24
25   [wwcgi]
26
27   ; ================================
28   ; BASE DEVELOPMENT SETTINGS
29   ; ================================
30
31   Path=C:\Users\Public\FoxInCloudTemp\ttf\
32   ; Full path to wConnect Temp files directory.
33   ; REQUIRED
```

Any question? Post with screenshot in the 'FoxInCloud' section of http://west-wind.com/wwThreads/ or http://www.universalthread.com/

Create script-mapped extension

Make sure ISAPI is allowed at server level. If you run Windows 64 bits, allow 32-bit appls at server level.

Script-mapped extension created

When clicking OK, allow the ISAPI module to execute.

Redirect index.htm to index.xxx

'xxx' is your application code / that you've chosen during FAA step 3- Publish

Run your server as plain exe (file mode)

Running your xxxTest | Prod.exe is just like running your server from the IDE: both use wConnect 'file' mode
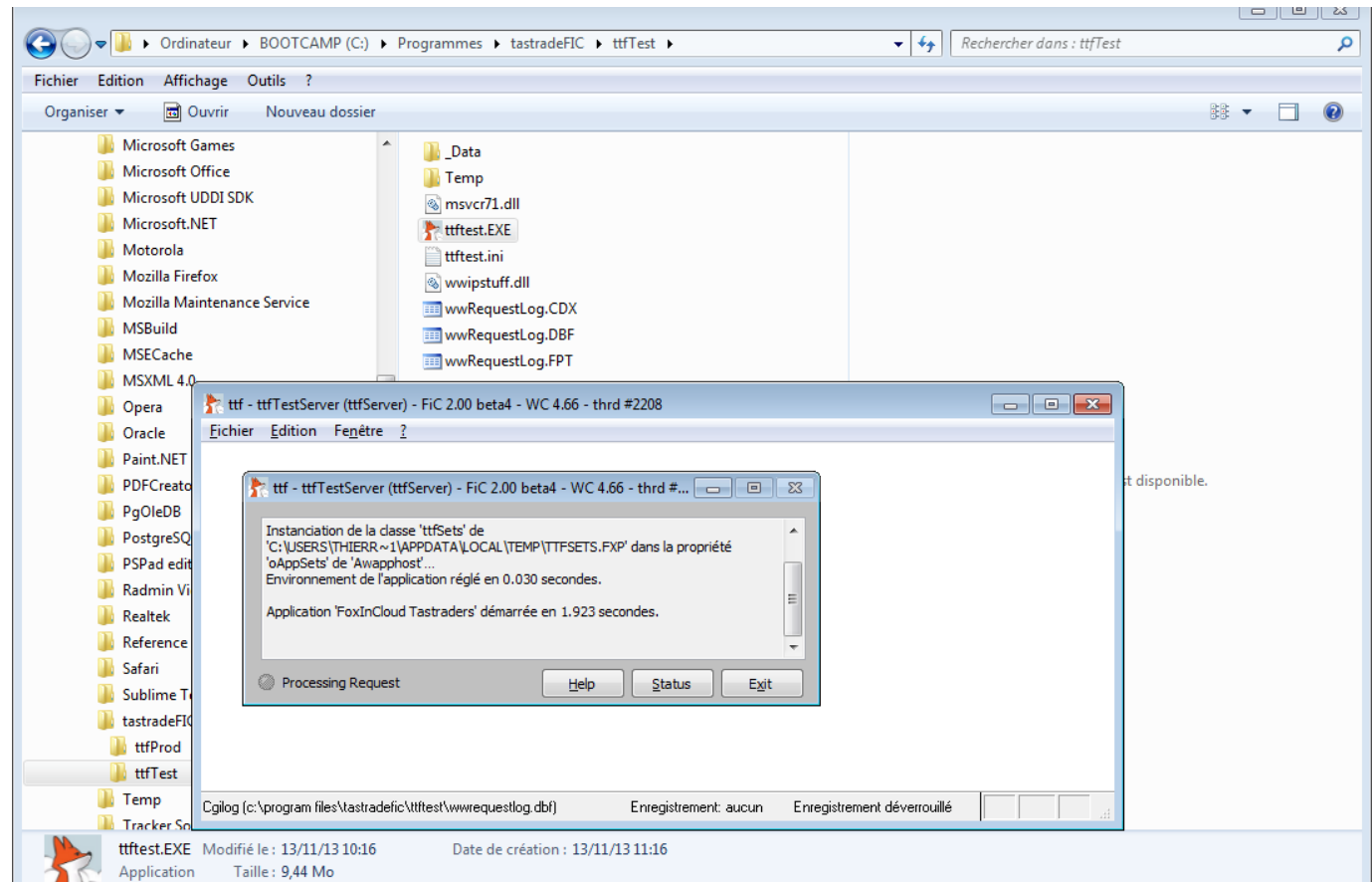
Site home page responds

Form displayed from menu

Switch to COM mode

Form displays in COM mode

Our VFP exe runs in COM mode as our dedicated user

FoxInCloud app status

This page will include more and more functionalities along future FAS versions, including optimization.