Deploying ASP.NET Applications

Chapter 11

What Are You Going To Learn?

- At the end of this lesson, you will be able to:
  - Apply 3 different approaches to deploy a Website
    - XCopy
    - VS Copy Website
    - Deploy Precompiled Web application
    - The Windows Installer (not in the syllabus)

Deployment

- deployment entails the process of getting your final application onto the production server and making it live.
- But prior to that, it also consists of setting up your project to work in various environments before you actually host it on the production environment.

Test Environment

- involved with the testing phase, which rigorously scrutinize every aspect of the code, its functionalities and its performance
- perform various testing processes on the application on an environment that closely resembles the production environment
- Plan performance, load or stress test

Web Server

- development environment tools, extra software and unnecessary applications that are not expected to run on the production environment SHOULD NOT be installed
- Check
  - Unforeseen dependencies inherited from your development environment
  - hardware mismatch
  - Configuration mismatched (database, mail setting, etc.)

Web Server Requirements

source: http://documentation.commvault.com/

Hardware Specifications

Based on scalability requirements, we recommend the following hardware specifications for the Web Server:

Environment	Specifications
Large	This configuration is oriented for large environments to support up to 10,000 laptops or 1,000 mailboxes. Recommended configuration: • 200 GB of disk volume • 16 CPU cores • 32 GB RAM memory • Cache size based on the amount of recalls and downloads performed from Web Console and Outlook.
Medium	This configuration is oriented for medium environments to support up to 2,500 laptops or 400 mailboxes. Recommended configuration: • 200 GB of disk volume • 8 CPU cores • 16 GB RAM memory • Cache size based on the amount of recalls and downloads performed from Web Console and Outlook.
Small	This configuration is oriented for small environments to support up to 500 laptops or 40 mailboxes. Recommended configuration: • 100 GB of disk volume • 4 CPU cores • 8 GB RAM memory • Cache size based on the amount of recalls and downloads performed from Web Console and Outlook.

Staging Environment

- to actually run the system as if it would run on a live production environment
- is the place where you might want to do last minute tests on everything
- perform stress and load tests, run lots of other real-time scenarios and see if it works as expected

Production Environment

- Products are actually put into operation for their intended uses by end users, i.e. the products are being used by end users in a real-time situation.

Transferring Files to Server

```
<configuration>
<system.web>
        <!--
            Set compilation debug="true" to insert debugging
            symbols into the compiled page. Because this
            affects performance, set this value to true only
            during development.
        -->
        <compilation debug="false“>
        </compilation>
</configuration>
</system.web>
```

Deployment steps

XCopy

- We can copy the assembly to another server and we do not need to stop/start IIS while copying.

- Allows us to move
  - Files
  - Directories
  - Entire drives

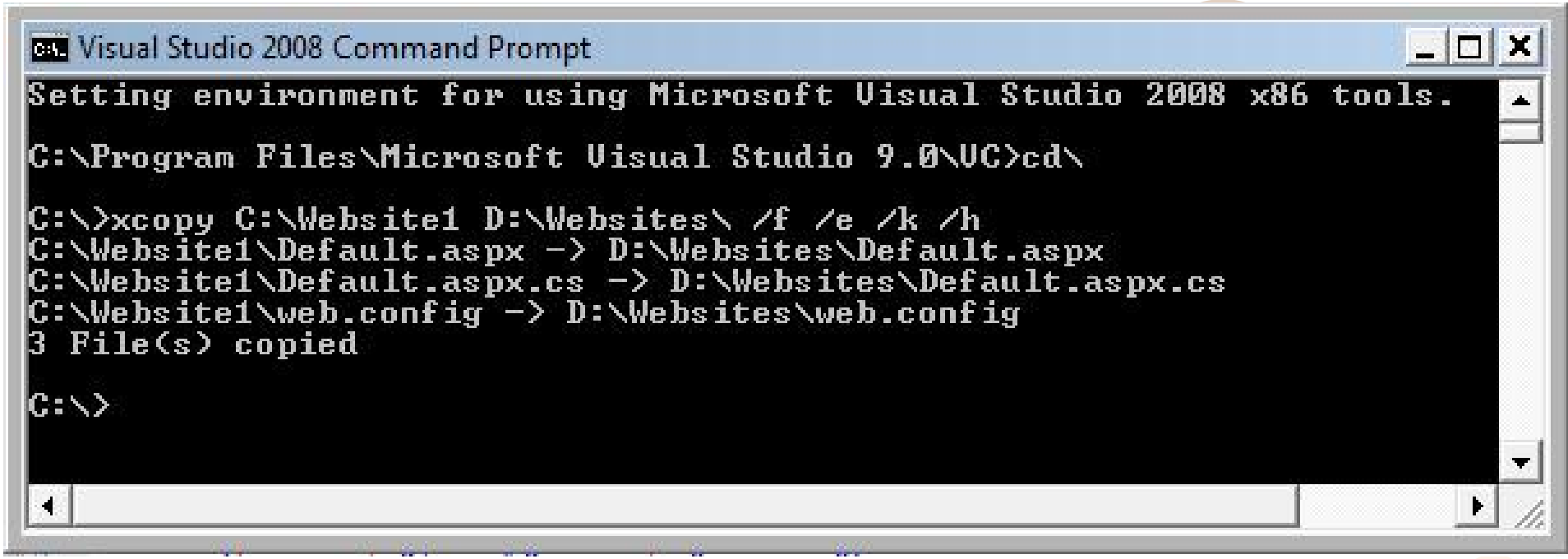
XCopy Syntax

- xcopy [source] [destination] [Parameter]

Parameter	Description
/f	Displays the file names from the source and destination files while copying process is occurring
/e	Copies all subdirectories regardless whether these folders contain files
/k	Ensures the read-only settings remain in place during the copying process
/h	To include the hidden and system files

For more details, read reference [1]

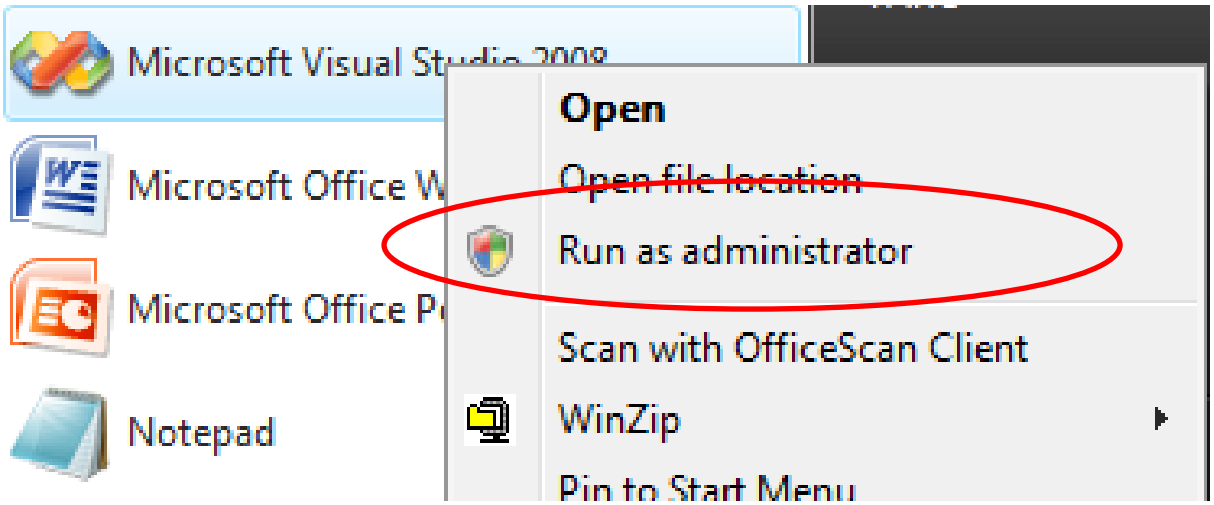
DEMO



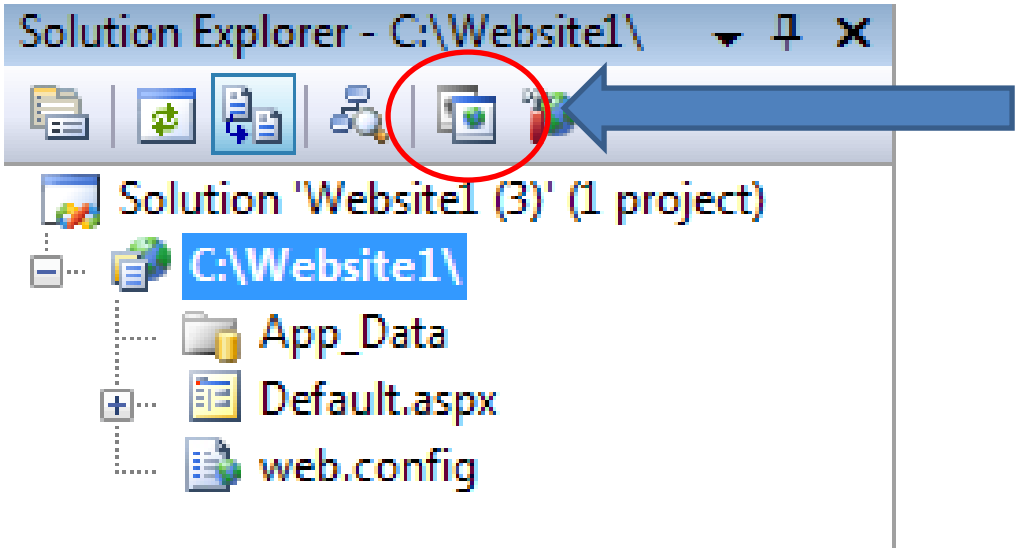
Xcopy Command

Using Visual Studio Tools

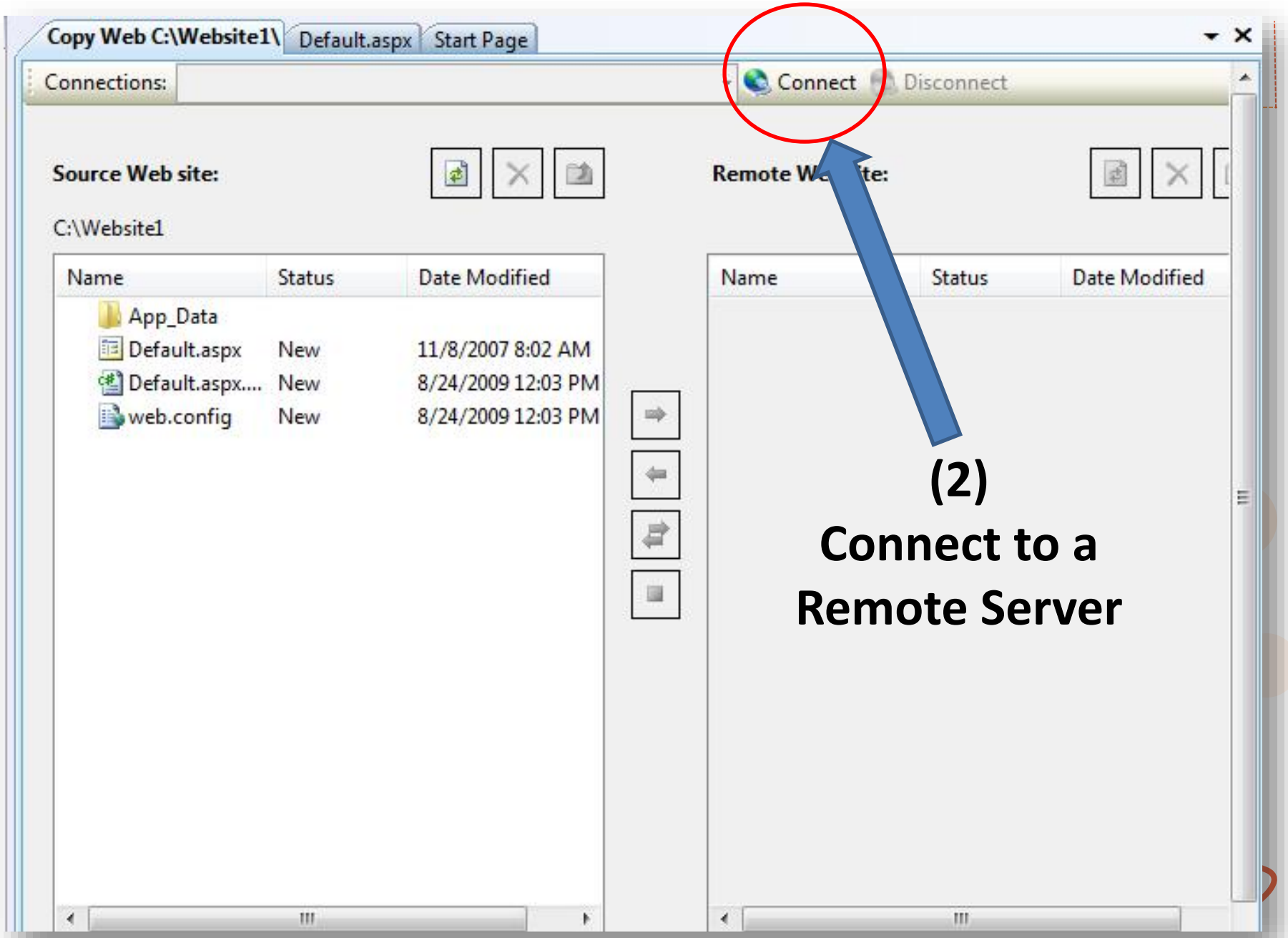
- Run VS in the context of administrator account

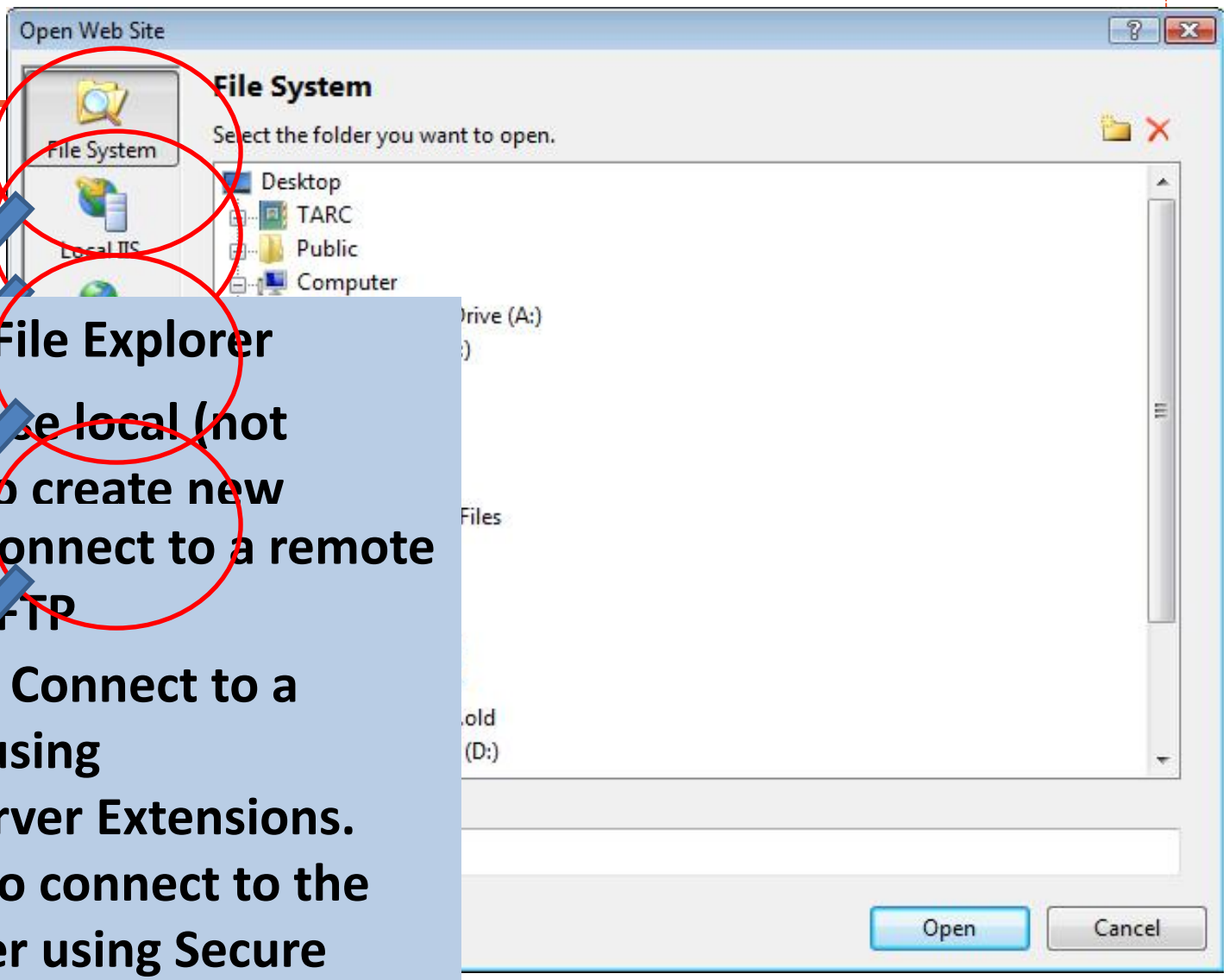


VS Copy Website Option



(1)
Copy Website


Copy Web C:\Website1\
Default.aspx
Start Page
Connections:
Connect
Disconnect
Source Web site:
C:\Website1
Name
Status
Date Modified
App_Data
Default.aspx
New
11/8/2007 8:02 AM
Default.aspx....
New
8/24/2009 12:03 PM
web.config
New
8/24/2009 12:03 PM
Name
Status
Date Modified
(2)
Connect to a
Remote Server

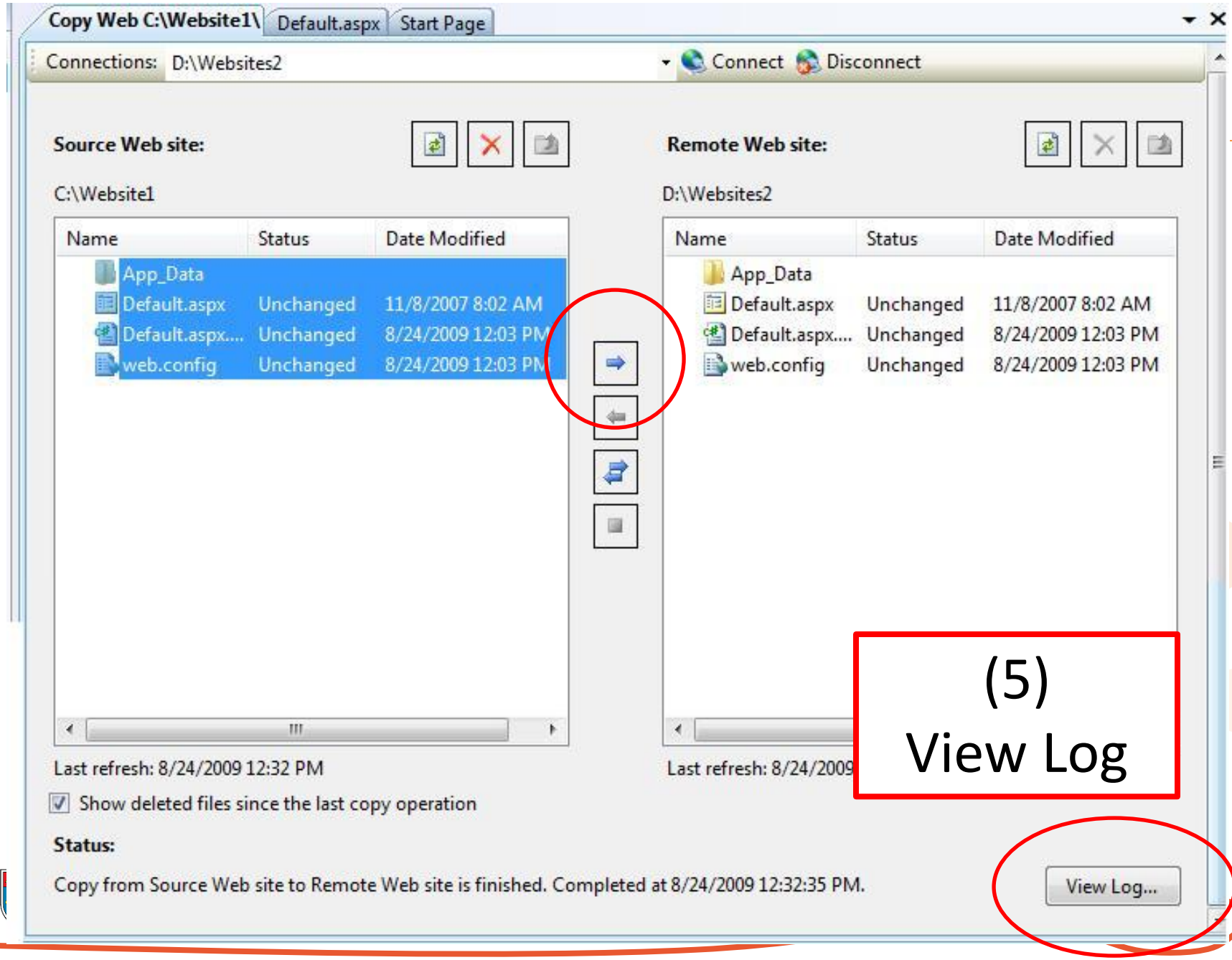


File System: File Explorer

Local IIS: To use local (not remote) IIS to create new

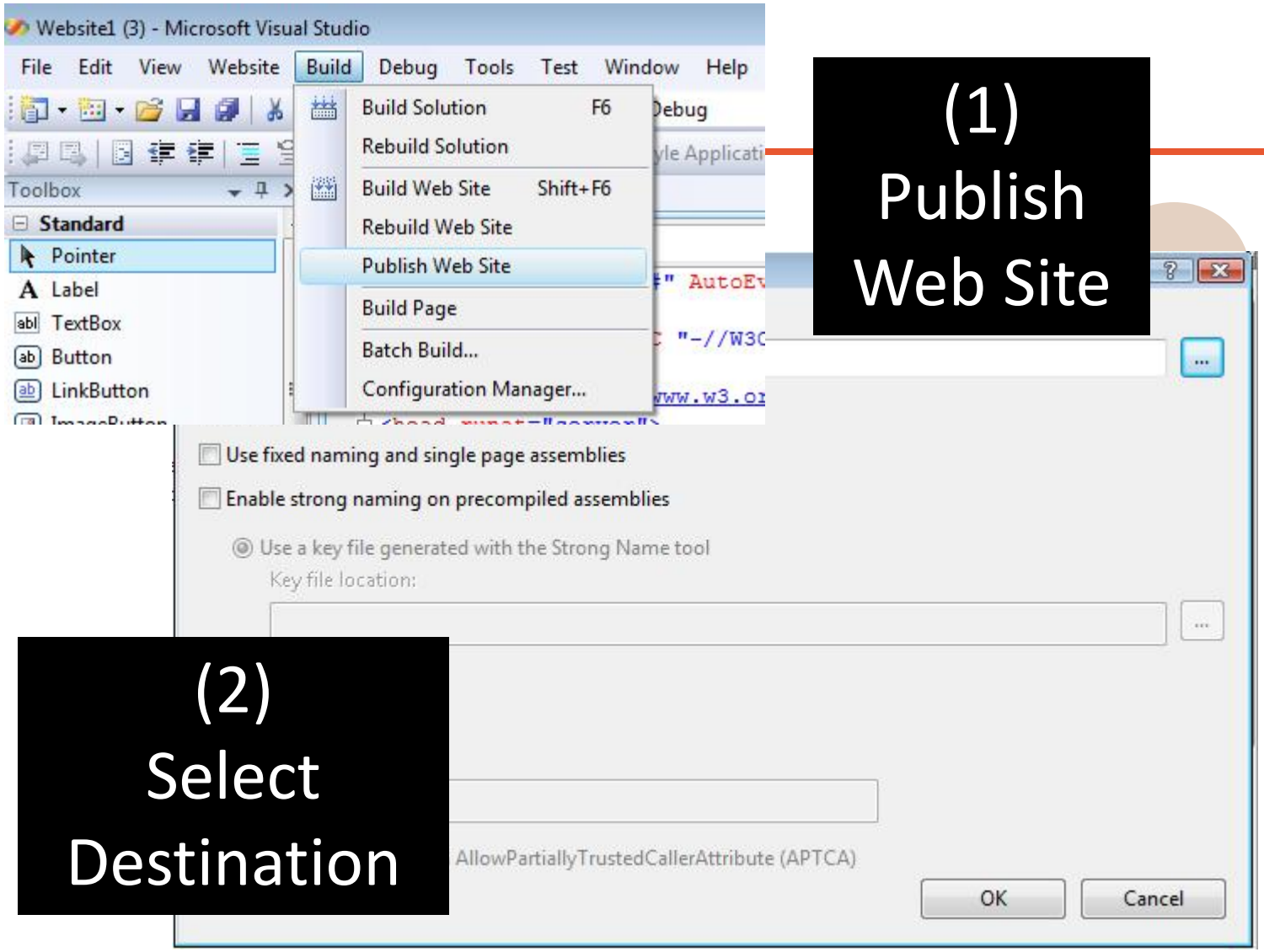
FTP Site: to connect to a remote server using FTP

Remote Site: Connect to a remote site using FrontPageServer Extensions. Can choose to connect to the remote server using Secure Sockets Layer (SSL)


Copy Web C:\Website1\
Default.aspx
Start Page
Connections: D:\Websites2
Connect
Disconnect
Source Web site:
C:\Website1
Name
Status
Date Modified
App_Data
Default.aspx
Unchanged
11/8/2007 8:02 AM
Default.aspx....
Unchanged
8/24/2009 12:03 PM
web.config
Unchanged
8/24/2009 12:03 PM
Remote Web site:
D:\Websites2
Name
Status
Date Modified
App_Data
Default.aspx
Unchanged
11/8/2007 8:02 AM
Default.aspx....
Unchanged
8/24/2009 12:03 PM
web.config
Unchanged
8/24/2009 12:03 PM
Last refresh: 8/24/2009 12:32 PM
Last refresh: 8/24/2009
Show deleted files since the last copy operation
Status:
Copy from Source Web site to Remote Web site is finished. Completed at 8/24/2009 12:32:35 PM.
View Log...
(5)
View Log

Deploy a Precompiled Web App

- Precompilation
  - Each page is built and compiled into a single application DLL and placeholder files.
  - Make difficult for your code to be stolen or changed
  - WHY precompilation????


Website1 (3) - Microsoft Visual Studio
File Edit View Website Build Debug Tools Test Window Help
Build Solution F6
Rebuild Solution
Build Web Site Shift+F6
Rebuild Web Site
Publish Web Site
Build Page
Batch Build...
Configuration Manager...
Toolbox
Standard
Pointer
Label
TextBox
Button
LinkButton
(1)
Publish
Web Site
Use fixed naming and single page assemblies
Enable strong naming on precompiled assemblies
Use a key file generated with the Strong Name tool
Key file location:
AllowPartiallyTrustedCallerAttribute (APTCA)
OK
Cancel
(2)
Select
Destination

ASP.NET .NET Framework
MasterPage Navigation Control
Event Handlers Client-event Server-event
Database Data Access Controls Advanced database-handling
Authentication Authorization

WHAT HAVE YOU LEARNT?

State management QueryString Cookie Session Application Cache
Validation Controls
User controls custom controls
Exception handling try-catch-finally Page_Error Application Error Custom Error Pages
Configuration Optimization
Deployment