IDC4OW
INTERDISCIPLINARY STUDIES – SCHOOL WEB PAGE II
COURSE OUTLINE AND EVALUATION

1. **TOPICS**

a) Intro
- Analysis of employment opportunities
- The role of a Webmaster
- Ongoing skills and competencies assessment portfolio

b) Web Page Creation
- Advanced Flash Animation and Actionscripting
    - loading external assets
    - using components
    - dragging and dropping
    - developing a Flash based website (the production workflow)
- Javascript
    - hand coding using the 1st page editor
    - event handling, functions, selection and repetition
    - rollovers, using arrays, working with combo boxes, creating pop-up windows
- Advanced DreamWeaver applications
    - ***review*** of tables/frames and navigation bars/CSS/library items/templates
    - Advanced CSS (Descendent Selectors and page layout)
    - Behaviours (DW javascript)
        - using the behaviour panel
        - using behaviours with layers (Draw AP Div)
        - using Spry widgets (menu bar/tabbed panel/accordion)
    - Dynamic HTML (DHTML – a combination of HTML,CSS,and Javascript)
- Creating Dynamic Web Applications using Microsoft Visual Web Developer Express
    - intro
    - manipulating data base records using Access
    - adding dynamic data to a web site using ASP.NET with Visual Web Developer Express
- SJB/Feeder School Web Site Creation Project

2. **EVALUATION**

Ongoing Assessment and Evaluation 70%			
Knowledge and Understanding 15%	Thinking and Inquiry 15%	Communication 10%	Application 15%
- unit tests - quizzes	- lab work - projects - independent study	- journal/blog - presentations	- assignments - projects - lab work
Final Evaluation 45% - Formal Exam: Written and Application (15%) - Final Course Project (30%)			

NOTE: (1) **Any <u>missed tests</u>**, for whatever reason, must be written the morning of the day the student returns to school, between approximately 7:15 - 8:15

(2) If suspended and a test/assignment is missed due to suspension a mark of zero is assigned

(2009-2010)
IDC40W
INTERDISCIPLINARY – SCHOOL WEB PAGE **II**

a) **INTRO**
- The role a webmaster
- SJB/Feeder School Web Site Creation Project (30%)
  - ongoing throughout semester

=> Go through examples in FlashReview Folder in preparation for upcoming lessons

---

b) **FLASH Review and Extension**

(1) **Review**
- Drawing Basics/Text Basics
- Understanding layers
- Symbols and instances
- Animation Basics
  - Frame by Frame (keyframetext.fla/faceframebyframe.fla)
  - Motion tweening (skateboarder.fla/sample3.fla/finalexplode.fla/effectsFinal.fla)
  - Shape tweening (shapecircle.fla/texttweenfinal.fla)
- Timeline Effects (Insert/Timeline Effects)
- Sound (SoundSyncFinal) (in Motion Tween folder)
- Motion guides (bouncingball.fla/motionguide.fla/multipleguides.fla)
- Mask layers (ccspottext.fla/tweenedshape.fla) -> set masked then mask layer

=First Animation Exercise (in Tweening/NotesAssignments folder)
- import rainbow.gif
- break apart using trace bitmap
- convert to symbols
- distribute to layers
- animate

- Animating using movie clips (rotatingwheel.fla/animsymbolfinal.fla)
  - first make instance of wheel, this puts shape into library
  - then make movie instance of same wheel and name it rotwheel
  - this now puts symbol into library we can now motion tween
  - double click to edit (motion tween in new timeline)
- tweening within tweens (tween.fla)

- Buttons basics
  - animated buttons
    - -newsimplebutton.fla/animatedhome.fla/flowerpowerfinal.fla
    - multi-layered buttons (instead of modify – use insert-> new symbol)
      - capsule.fla/surprisebutton.fla
  - adding actionscript to buttons (play/stop/geturl) (actscrptplaybttn.fla/actscrptlnkbttn.fla)
    - make sure animated text initially added to btn is static text
    - use on (release) event
  - image maps (invisible buttons) (invisiblebuttonfinal.fla) – remember to use layers

=> Minor Assignment (create an animated button within an image map application – See studentcouncil.fla)

- Putting it altogether
  - Combining HTML and Flash (CatToysSolution) using DW
    - inserting flash/flash text/flash buttons
  - multiple scenes (gotoAndPlay("scene 2",1)) (multiplescenes.fla)
    - (Window->Other Panels->Scene)
  - Targeting scenes
    - adding scenes (window->scene)/ duplicating scenes
    - gotoAndStop("scene 2",1)
    - Examples - targetingscenes.fla / gotoSceneFinal
  - using **Movie Explorer** (Windows menu) to view entire project

  - The goto Action to create a slideshow (nextFrame/prevFrame)
    - slideshow.fla -> slideshowFinal.fla

---

(2) **The Next Step ActionScripting**

(i) Intro
- the 3 places you can put actions
- keyframes (consist of a ***list of cmds*** that are executed as soon as the keyfrm is reached or a ***func***
- button instances (can't just enter a series of cmds instead you react to an event)
- movie clip instances (also uses an event handler)

(ii) Frame Actions
- using the trace command attached to a keyframe (trace("Hello");) – demo with simple mot tween
- Using actionscript to create non-linear movies (gotoAndPlay)
    - make action that loops part of movie (BeforeNonLinearMovie.fla/NonLinearMovie.fla)
    - Using a frame label as the destination of a goto (NonLinearMovieUsingFrameLabel)

(iii) Button Actions
- adding buttons to your animation to stop and continue playback
    - NewActionscriptPlayButton.fla (note use of separate layer for frame actions)

(iv) Movie Clip Actions
- **Using Instance Names**
    - NewMovieClipActionsInstanceName.fla
        - **instancename.method** ->eye.play()/eye.stop()
        - Note: frame action (frame 1) eye.stop() stops rotation /stop() stops movement across screen
    - mcplayback.fla
    - changingimages.fla (note: use of mclip.gotoAndPlay("frame"))
- **Properties**
    - object property syntax -> **object.property** (theclip._alpha)
        - where theclip is the **instance name** of the object
        - all built in prop are preceded by an underscore
    - some properties
        ._x (x pos), ._y (y pos), ._alpha (current alpha), ._rotation (curr clip rot)
        ._width, ._height, ._xmouse,._ymouse, ._xscale, ._yscale
        ** ._x,._y -> relate to centre of object NOT left,top
    - only instances of button, clip and dynamic or input text can be changed during run
    - Examples
        - properties1.fla (modify and experiment with other properties)
            - eg. theclip._xscale= -theclip._xscale (performs horiz flip)
        - movieclip_rotation.fla
        - Mcproperties.fla
- **Nested Instances** (paths to objects)
    - Intro
        - levels2.fla (movie clip inside a movie clip)
        - do with students/use double movie clip technique for rectangle and name on main timeline
            - then create circle symbol inside rectangle
        - format is clip1.clip2.method()/clip1.clip2._prop
    - Practical application
        - InstanceActions1.fla (stop btt stops the car but not the wheels from spinning)
        - InstanceActions3.fla (name front and back wheels on **main** timeline)

=>Problems (NestedInstancesProblem)
- note: menu buttons were created by duplicating in library
- nose and mouth anim are shape tweens /first draw objects and make into movies (only once) then enter mClip library where clip is a shape and tween

---

(v) Programming in Flash A Closer Look
- **variables** (using dynamic text and input text) show border
    - examples
        - using a variable to get the users name (variables1.fla) var box in prop panel
        - math application (mathinput.fla)/ incrementing a variable (variables2.fla)
        - sample application (netpay.fla)
- **Decisions (Newestif1.fla)**
    - intro (if2.fla) (note use of ==/ &&/||)
        - add age=parseint(theage.text) where theage.text is the instance name of the textbox

- **Dynamic Text** (DynamicTextExample folder)
    - static vs dynamic text web pages
    - loading dynamic text fields using external files (no scroll bar appears-> fix in More Assets lesson)
        - loadVariablesNum(filename,0)
        - var name of dynamic textbox must match name used in external text file (content=)

=> Dynamic Text Problem (3 image buttons which when pressed display corresponding info in a dynamic text field)

(vi) **Advanced Flash Programming**

- **Using Frame Events** (Review/Extension)
    - main_800.fla (note use of scrollbar component)

- dynamic text box was inserted first with properties (multi-line/selectable/show border)
- then the scrollbar component was attached (just drag onto textbox and it should snap to it)
- then text was COPIED and pasted into it

- FrameEvents.fla
  - 200 frame scene with each pic (mc) visible for 10s (20 fps)
  - see last actionscript for pictures.nextFrame();
  - text descriptions are added from pic mclip timeline action panel
  - note use of -> on(rollover)/on(rollout) for buttons

- **Using Sounds**
  - Intro (MoreSounds folder)
  - The **Sound Object** ***
    - Trigging a sound from the library (sound.fla) sound not on timeline
    - make sure to set it's symbol linkage properties (right click in lib)
    - newbounce.fla (uses multiple instances of mclip with embedded actions)
    - note use of Stage.height/Stage.width
    - SoundOnOffFinal (MoreSounds)

- **Loading More External Assets (*ExternalAssets.fla/external.htm*)**
  - loadMovie("file","_root.mclip") used to load .swf files/images .jpg
    - need to first create a placeholder which is an empty movie clip instance
      - insert/new symbol/mclip
      - then drag empty mclip instance on stage
      - this will hold the external content
  - loadVariablesNum(file,0)
    - used to load text files
    - variable name of dynamic textbox must match name used in external text file (content=)
  - loadMovieNum("file",1) (optional)
    - content of movies on higher level appears above content from levels below
    - don't use level 0
    - top left corner of movie loaded by default to position 0-x,0-y
      - need to add code to each mclip in frame 1
      - _x=_level0.place._x; (place is the name of mclip in ExternalAssets.fla)
      - _*y*=_level0.place._y;
  - unloadMovieNum(level) – added to SJB Opening (to remove movie added to layer 1 by last btn)

- U**sing More Components**
  - Intro (**component_finish.fla** from bookdemo folder)
  - Scroll Bar (CS3ScrollBarFix.fla)
    - note the use of *event handler method* in frame actionscript and instance name use
  - Combo box (**UsingComponentsCS3.fla/UsingComponentsCS3.htm**)
    - note use of two mclips and use of and unloadMovie(_root.clip);
    - set label parameters for combo box (bottom of CS3 screen - property area)
    - highlight use of *function* **change** and cb.addEventListener("change", this);

=> Combo Box Assignment (create a combo box with 3 choices and then display a corresponding image and text description using a scrollbar

---

- **creating drop down menus**
  - **introduction** (highlight key components - from CBT CAFE)
    - FlyOutDemo (note the invisible button with the cut out used with on(roller)-last layer)
    - MenuFinal (in Bookdemo folder) - 3 layer technique (buttons/labels/actions)
    - DropDownMenu (note: use of _root.content="")
  - **We do Example (3 choice menu with 2nd and 3rd choices having sub menus)**
    - type text that will be used for menu items
    - convert each to buttons and include desired *over* effect
    - convert buttons to one movieclip
    - now work within movie clip
      - label initial layer **buttons**
      - add new layer called **labels**
      - select the first keyframe and name it **begin**
      - on the labels layer select frames 10,20,30 and press F7 to add blank keyframe
      - label the frames one,two,three
      - select frame 40 and press F5 to add frames so that you can see label three
      - add a new layer called actions and add stop to the keyframes
        - note: code for some buttons will be right in the frame actions and for sub menus right on button
      - now create key frames in the button layer to match the labels layer
        - add the view of the button at that moment
        - make sure to move the playhead to the correct frame first
        - add actionscript for **all** buttons at that moment including begin frm

```
on (release)
{
   gotoAndPlay("one");
}
```

        - if button has no submenu then add to frame actionscript code
          eg. loadMovie("sjbspring.jpg",_root.place);
        - if button has submenu then add new text below it and convert to buttons and then add code directly to buttons

---

- **Dragging**
  - dragging a button using startDrag(_,_) and stopDrag()
    - drag1.fla
  - dragging a movie using an invisible button (ie. Invisible button inside movie technique)
    - dragcomplex.fla
      - note : **_root.circle**
    - draganddrop.fla
      - after dropping make *root.green*._alpha=0 or ._x, ._y back to orig pos
    - draganddropMovie.fla
      - use trace(_droptarget) in on(release) to det exact target name ()
      - target and drag obj can be on diff layers
      - target mc and mc inside mc should be given instance names

- **Event Handler Methods**
  - used for same purpose as regular event handler
  - script is placed on Frame rather than attached to instance
  - allows instances to react to mouse and clip events (usually cant attach both kinds of evt)
  - eg. on(press) {
    gotoAndPlay(5); **this is a regular event handler attached to a Button**
    }
    myButton.onPress = function () {
    gotoAndPlay(5); **these are event handler methods**
    } **placed on main timeline frame**
    myButton.onRelease = function() {
    myMC._xscale=50;
    }
  - Examples (movieclip_droptarget.fla)
    - can also add_*root.circleBig*_mc._alpha=0; to droptarget if statement to make large mclip disappear

=> Traffic Signs Problem

WORK PERIOD

WORK PERIOD

=> Practical Test

c) **Developing a Flash Based Website**

(1) **Sample Sites**

- FinalHalfSite.fla/swf - in sjbwebsiteFlash
  - note: The movie clips that display the images are right underneath the tear drop images
- XboardingSiteFinal
  - Login:guest Password:gotsnow

(2) **The production workflow (**DaisySite)

- create an extendable site structure (multiple scenes/consistency/navig)
- plan the basic site structure
- HIPO chart (see flowchart)
- functional specifications (see functionalSpec)
- establish key elements (visual site layout – see sitemap)
- organize the document layers
  - layer1 Actions/layer2 Labels/layer3 Logo/Layer4 Main Menu
  - layer5 Captions (for loaded and static text)/Layer6 Loaded images/clips
  - layer7 Background
- prepare graphics for static or dynamic loading
- place static elements and text
- place movie clips
- work on navigation

---

(3) Demo OtherSites (folder) Flash Templates

- demo all sites but focus on ***Atoms_an_Gujan/Fullsite***

(4) Flash Website Assignment (use production workflow techniques)

- create from scratch or use a Flash Template

---

10 WORK PERIODS

d) **Javascript**

(1) Intro

- what you can do with javascript (see Week1 Intro to Javascript at www.sislands.com/coin70)
- sjb website
- where to put javascript
  - ***monkeyalert1.htm***/students type in after demo using script tab in 1st page
  - external scripts (<script language="Javascript" src="filename.js"> … </script>
  - note: comment enclosures no longer required

(2) Output

- document.write/alert (***kittointro1.htm/ketsex11.htm***)
- alert box with line feed (alert("Hello" + '\n' + "there"))

(3) Variables

- ***ketsex13.htm/newmonkeyvariables.htm***

(4) Input

- ***newKittointro2.htm/ketsex14.htm/ketsex15.htm*** (instead of eval can use parseInt())
  - prompt/confirm

=> Exercise (jex1) (demo to class first) + then go through Javascript Examples (www.w3schools.com/js/default.asp -> Basic/Statements/Variables)

=> take up ex

(5) Events (including FORMS) – use summary from Missing pg 335 (Mouse/Keyboard/Body/Frameset/Sel/Form events)

- onLoad (put in body tage -> onLoad = "alert('welcome')"
- onClick (***vdeventsintro.htm***)
  - students type in after demo to acquaint themselves with forms and **submit bttn**
- onMouseOver (***monkeyonclick.htm/ webteacheronmouse.htm*** + add another image link)
- onMouseOut (***webteacheronmouseout.htm/newKittointro3.htm***)
- onBlur/onFocus (***newwebteacheronblur.htm***)

(6) Creating a form in Dreamweaver and hand coding simple Javascript events

- must first insert the basic form container then add elements like submit buttons
- note: button "value" is in property bar along bottom of screen
- go into design mode and hand code

=> Exercise (jex2) demo to class first + Intro to Javascript Week 2 (Event Handlers)

=> Take up before end of class

---

(7) Functions

- intro
  - ***vdfunctionintro.htm/vdfunctioneventintro.htm/winmagdemo.htm***
    - move myfunction call into body -> observe error
    - function call must be part of an ***event*** -> eg. attach to a button
      - when it is used within the body
- naming hierarchy (document. __)
  - ***ketsex21.htm***
  - ***newketsex21.htm*** (uses document.getElementById("x").value)
- functions with parameters
  - ***webteacherfunction.htm/webteachercolor.htm***
- Recap Examples of functions and events
  - ***Kittotextbox*** (introduces onBlur/also discuss onFocus event)
  - ***vdstatusbarnew***
    - Displaying info on the status bar/ reviews functions and events
    - modify function to have 2 parameters
    - statbar(txt,col) {document.bgColor=col;
    - onMouseover="statbar('d','blue')" onMouseout="statbar('','red')"

=> jex3a/newjex3b/jex4/newjex4-uses getElementById (note **form button** *NOT* **submit** for jex3a/b) then Javascript Tutorial (Chapter 1,2)

---

(8) Local and Global variables

- a closer look at **var and functions that return values** (***webmonkeyvar1/webmonkeyvar2***)
- by default variables have a global scope
- by using **var** you localize tht variable to the one function that houses it

(9) Selection Statements

- Basic Selections
  - ***webmonkeyif1/webmonkeyif2*** (note use of x==y / && / || / !x)
- Simple Applications
  - ***webmonkeyif3/ifelse1/webteacherifthen***(note use of Javascript:password()***/ketsex22)/quiz***
- Checkboxes and RadioButtons
  - ***webmonkeyifthen2*** (**checkbox**)
  - ***webmonkeyradiobttnsnew.html*** (**radiobutton**)
- Creating checkboxes and Radio Buttons in Dreamweaver (***DWRadioCheck***)
  - add javascript code (on common tab-script)
- Using getElementById (***RadioGetElement***)

=> Exercise (newjex5.htm/jex6/jex7) + Read JavascriptFormsFunction.pdf

---

=> Take up Problems

(10) Image swapping (Rollovers)

- basically involves image links with multiple images
- ***-monkeyimageswapnew.htm/voodooimages.htm***
- demo how to create images in Fireworks then students do flashing name in DreamWeaver
  - The canvas
  - Drawing objects
  - click and hold displays a menu of shapes (shft ellipse ->circle)
  - Add Text/adding effects (shadow..) to objects
  - Modifying the Canvas (trim) * canvas is reduced to exactly fit objects
  - Saving (.png)
  - Exporting Fireworks doc using Image **Preview** then Export
    - select index transparency /choose format gif/jpg /save as image)
    - opening and importing images/cropping (crop tool followed by double click)
- ***ketsex25.htm***

=> Problem (rolloverimagemap.htm)

- More Rollovers
  - ***script1***
  - ***script2*** preloading images using Image object
  - disjoint rollovers
    - triggering rollovers from a link (***script3***)
    - multiple images changing a single rollover (***script4***)
    - multiple rollovers (image that triggers rollover is a rollover) (***script5***)
      - note use of ";" to separate rollovers
    - using a function to simplify coding multiple rollovers (***script6***)

=> Minor Assignment (***rollovermenu***) -> create a rollover menu + Intro to Javascript (week 3 image swapping overview www.sislands.com/coin70)

- Using Array Objects
  - Intro (Intro to Javascript – week 3 Arrays)
  - creating cycling banners (vs animated gifs)
    - ***newscript8.html*** -> note use of setTimeout()
    - ***newRotatingBanner.html*** -> note: use of .length
  - adding links to cycling banners (***newscript10.html***)
    - note use of Javascript:newLocation()
  - building slide shows (***newscript11.html***)

=> Problem (create a banner that cycles through 5 images and links to each)

---

(11) Working with Combo boxes

- Intro (Javascript Tutorial – Chapt 7)
- Using the **options** array to access elements within the list (***javatutorscombo1new.htm***)
  - copy document.George from last onClick into 2nd alert options [ ] (value property)
  - *then* students type in sample combo box in 1st page (select list) (***comboproblem.htm***)
    - change choices to colors and add a button to determine which color chosen
- Creating combo link boxes (***newjavatutorscombo2.htm***) (notice type="button" not submit)
- Frames (***javatutorscombo3.htm/page1/page2***)
  - note use of Parent.Jane (where Jane is right frame name) in ***page1.htm***
- Using an image instead of a form button as the go button (***newjavatutorcombo4.htm***)
  - using a javascript url
- Creating a combo box that jumps to a URL upon selecting (***newjavatutorcombo5.htm***)
  - using onChange event
- Sample application (***kittocombo.htm***)
- Creating pull down menus (jump) in Dreamweaver
  - located in forms tab
  - values are placed in URL
  - demo linking to other pages or other external sites
  - click in behaviour tab to edit
  - hand coded pull down menus vs DW jump menus
    - more control using hand coded method eg. application similar to ***kittocombo.htm*** hard to do in pure design mode in DW

=> Minor Assignment (newjex8) (demo to class first)

---

(12) Repetition Statements

- While statement (***webmonkeywhile1.htm***)
- For statement (***for1.htm/for2.htm***)

(13) Arrays Revisited

- ***arrayintro.html*** (uses repetition)
- Sample application
  - ***ketsex23.htm***
    - illustrates the efficiency of arrays
    - introduces the idea of a control array (all radio buttons named group)
  - ***newketsex23.htm***
    - adds a second array and uses a **for** loop
  - slideshow on school website (***main.html***) demo and look at code

(14) String Functions (***JavaScriptStringFunctions.htm***)

Recap Application

- ***main.html*** (slideshow from school website)
- demo javascript free code sites (http://Javascript.internet.com) The Javascript Source

=> Problem (modify main.html)

---

(15) Creating Windows (window.open)

- Intro demos
  - ***windows1.htm/windows2.htm/windows3.htm***
    - note use of -> left=,right=
    - note page2.html use of Javascript:close()
  - ***script01.html/script02.html***
  - ***voodoowindows1new.htm***
- Sample Application
  - go through VirtualScienceFair2004 (note use of resizeWin() function)

(16) Making pages Dynamic

- demos from javascript2 (chapt 7)
  - ***script01.html/02/03***

=> **Review for Test**

JAVASCRIPT TEST

---

(16) Using DreamWeaver Advanced

Review and Extension

- **Tables**
  - Introduction (the two ways to use tables)
    - table.html (inside chapter07_finished within DWTables folder)
      - used to display rows and columns of information
    - FINAL.html (inside DWTables)
      - a table based web page layout
    - demo sample site (www.lacountyarts.org)
  - standard mode (layout tab)
  - layout mode (View Menu/Table Mode)
    - first set layout table then cell (on very right side of layout tab)
    - draw table in layout mode then switch back to standard view to add bkgs
    - flexible width table (autostretch)
      - you set one column in the table to autostretech
      - click the down pnt triangle next to the pixel width num in col header
      - creates a spacer image file
  - using a tracing image (page property – in modify menu)
  - importing data into a table
    - first export in tab delimited format (Access –in More Tables folder)
    - then import tabular data
    - new in CS3 Excel imports directly – no need to export in Excel first
    - sorting data in a table (Commands -> sort table)
  - importing Word documents (new in CS3 Word imports directly)
- **PhotoAlbum Creation**
  - Using Dreamweaver (In Commands Menu)
  - Using Porta
  - demo both using the SampleImages folder/in Porta choose Edit Album/options

(HTML/SimpleView)

=> Exercise 2

- **Frames and Navigation Bars**
  - review **frame** settings (***IntroFrameLesson*** folder)
    - overview/review creating a DW site (covered in detail in a later lesson)
    - create a site (have it point to ***CompleteNonFrameSite***) then
    - start by opening or creating a page that will appear within the main frame
    - then choose view/visual aids/frame border
    - then choose one of the pre-made framesets (on layout tab – far right)
    - select frames by setting Windows-> Frames
    - save the frameset and the frames (for subsequent saves use save all
    - set frameset properties (borders)
    - set frame properties (name/scroll)
      - choose frame from frames panel first
    - add more pages/ target links
  - **iframes** (in DW access from layout tab in code view)
    - intro
      - <IFRAME> works like <IMG> only instead of a picture on the page it puts another web page
      - ***iframe1/iframe2***
    - incorporate in above example used to review frames
      - in <IFRAME> need to set src= and name= /then set target in links
  - inserting a **rollover image** (common tab)
    - intro (demo ***finished.html*** in DWImages folder)
    - must create a new site which points to DWImages folder (work thru tutorial pg 136)
  - inserting a **nav bar** (see pg 216 Missing Manual)
    - Intro (tsn example ***BasicNavBar***) – discuss diff between rollovers and nav bar
      - then work through the creation of this nav bar
      - first create graphics for navbar in FW (up to 4 different states)
      - then exporting for use in DW
    - Applications -
      - ***framedNavBar***
      - demo Frames Tutorial ("When Cabbage Attack" – DWFrames pg 220)
      - illustrate nesting framesets
      - while in centre/main frame (modify/frameset/split-left/right)
      - note disjoint rollover (techn revisisted during behaviour lessons)
  - Extension (***TwoFramesOneClick***)
    - how to load 2 or more frames with one mouseclick (Advanced Tutor)
    - introduces how to add Javascript in DW codeview
      - -insert/script obj (for internal)/right click tag to access external
    - using function multiLoad(d1,d2)

- **CSS**
  - **Review** (IDC3OW)
  - intro
    - Sample Sites (Zen Garden/CSSPlay)
    - Why Use ? (***beforebasic1.html/basic1.html/basic2.html***)

  - creating styles using **TopStyle Lite and 1st page** (use internal/external templates)
    -internal(***webmonkey1***)/ external(***webmonkey2-mystyles.css***)

  - adding styles in **DW** (use same demos as above)
    - use **CSS styles (right tab)** and choose **All** then **New CSS Rule**
      - choose Selector Type tag then
      - apply style / editing style/ also view HTML code
    - **attaching external style sheets** (created in TopStyle Lite/DW)
      - attach style sheet to **NEW** page

  - defining styles using **classes and ID's** (TopStyle-1st Page vs DW)
    - Intro (***classintro.html***)
    - you can apply an ID only once per page
      - note: to add ID use Advanced Selector type
    - ***basic4/newdoc2(basicstyle.css)/newwebmonkey5/chapt_02***
      - highlight -> background: repeat/A:Link/A:Hover
      - .bottom a:hover (previews descendent selectors)

    - note: in DW highlight area to style and click apply style in CSS panel
      - demo with ***basicstyle.css*** and ***apple.htm***

  - using positioning (pos1/ layer1/3dlayer1/layer2/webmonkey6)
    - recap main ideas using 1st Page 2000 then
    - demo briefly how to create positioning commands in DW
      - now called Draw AP Div in CS3 (Layout tab)
      - creating layers/adding text or images
      - properties (bgColor/Images/z-index/visibility)
      - add a dashed border around some sample text by using a class

  - go through Tutorial pg 139 (**CS3DWMissingManualChapt4**) with class

=> CSSMissingManualChapt2 (students compare original to finished)

=> Take up problem

- Controlling images with CSS
  - intro (Finished.html -> highlight CSS)
  - **CS3DWMissingManualChapt 6**
    - box (float)/border (dashed)
    - note: floated element must appear before anything that you wraparound
    - note application of styles to images
  - Note the use of #wrapper to center web page (surround all content after the body tag)
    width:760px
    margin : 0 auto or margin:0 auto 0 auto (top/right/bottom/left)
    border-left: 2px solid black
    border-right: 2px solid black
    padding 0 20px

  - Demo how to insert **DIV** tag (covered in detail in next lesson)

=> **CSSMissingManualChapt8** (compare original to finished – add wrapper id)

- **Advanced CSS**
  - **Descendent Selectors**
    - example: .MainStay td li
    - meaning -> apply style <li> tag but only when inside a <td> tag and only when <td> tag inside another tag that has the class MainStay applied to it.
    - Another example: p.intro a {color:yellow}
    - meaning -> apply style to every link (a) that's a descedent of para (p) that has the intro class applied to it
    - Recap (**CSSMissingManualChapt3** pg 61-70) Selector Sampler
      - reviews basic selectors/classes/ids using div tag/descendent select
      - demo how to directly type css code in codeview
        - show when entering id selector code
  - **Page Layout**
    - float layout basics
    - Basic structure (similar to frames idea)
      - #header
      - #sidebar1
      - #mainContent
      - #sidebar2
      - #footer
    - Example (work through with students see **CS3DWMissingManualChapt9**)
      - uses the new CS3 CSS layouts
      - 2 column fixed left sidebar header and footer
      - remove some of the pre-set visual formatting
      - add new content
        - add images or html files to each structure
        - header/sidebar/content
        - select file and copy and past in required area
      - add some style
        - attaching an external style sheet
      - skip fine tuning but do finishing touches
        - add navigation bar after banner image

=> **CSSMissingManualChapt7** Tutorial pg 158

- **Using CSS Templates** (compare original to modified versions)
  - in depth analysis of :
    - the use of the div tag to structure the html page
    - the style sheet code: note use of the **descendent selectors**
  - Using FireFox with FireBug add-on to analyze CSS code of websites

---

- **Site Management**
    - properly defining a site in DreamWeaver
    - create folders before defining site, including an images subfolder
    - go to Site->Manage Site -> remove any old sites still in memory
    - Site -> New Site -> choose none for remote server
    - try to make site name and folder name the same
    - check out advanced tab settings (set default images folder)
    - Demo a simple 3 page site (index/page1/page2) -> use pictures in **images** folder
    - To work on an existing site (eg ***BELLUR***) -> reviews navbar and css
        - first save a copy of all site files in one folder and then
        - when defining the site choose this folder for the local root folder

- Libraries/Assets
    - Creating and using **Library** items (Window > Assets Panel – should appear at bottom right corner)
        - only works for properly defined sites (demo with SCUBA site)
            - to add a library item to a web page, simply drag it from assets panel
            - drag item (table, text etc) to library section (bottom panel)
            - edting library items (the true power of the library)
                - double click library item
                - mini page .lbi appears
                - make changes then save and update
                - also need to save all so that each page updated is also saved
        -** Note: problem at school may occur after saving, assets panel library item disappears
            - still exists in site files

- Templates
    - intro (why use?)
    - **creating a template** (turning a web page into a template) -> File-Save As Template
        - use the sjb template example (work through with students)
        - template is based on a pg which is designed with tables in layout mode (view/table mode)
            - switch back to standard view to add bkg colours etc
            - call this page ***dummytemplate***
        - once it has been used as the basis for template (saved as template) it can be discarded and all subsequent pages will be based on this saved template
    - **defining editable regions**
        - by default all regions are locked initially
        - select region you want to make editable/right click or use template tab
        - you can re-lock regions (remove template markup – in Modify/Templates)
    - **creating a new page based on a template** (File/New/Page from a template)
    - **updating a template** (go to Assets/File panel and double click on template)
        - make changes/additions then Save -> Update then Save All
    - **unlinking** a page from a template (Modify -> Templates -> Detach)

    - Using Premade Dreamweaver Based Templates
        - demo Intro (t2 template)
        - then highlight templates in UsingSampleDWBasedTemplates folder

    - Other DreamWeaver Based Templates

Behaviours (pre-packaged Javascript in DreamWeaver)

- Behaviour Elements
    - HTML tag
        - where you apply events and actions to
        - **link** is most common also buttons,menus
    - Event
        - trigger that causes action to happen
        - usually begins with **on (click/mouseover etc)**
    - Action
        - what behaviour is suppose to do
        - eg. open new page/display message
- The Behaviour Panel
    - Select HTML **tag** or object
    - click **action** (+)
    - set options for action
    - change the **event** if required
    - you can add multiple behaviours to an HTML tag
    - double click to edit

- Examples (work through with students)
    - **pop up message** (illustrate underlying code)
        - add to body tag -> onLoad
    - **set text of status bar** (use link #)
        - add two behaviors – one for onMouseOver and the other for OnMouseOut
    - **set text of text field** (make sure to name text field – set ID and use onClick)
    - **open browser window** (use a submit button with onClick)
    - **goto url**
        - opens a new page in the current window or specified frame
        - useful for changing content of two or more frames with one click
            - demo ***TwoFramesOneClick***
        - otherwise it would be like using a regular link
        - Work through a 3 frame (row style) setup where menu is bottom frame with two links
            - create site that pts to demo folder TwoFramesOneClick
            - can use already created test pages to place on frames

- Using Behaviours with layers (now called Draw AP Div in CS3)
    - Intro (demo sample in DrawAP folder)
        - modify demo to illustrate NEW **Effects** Behaviours
    - Go through simple example with students
        - note use of **layer id**
        - set text of layer (use a text/image link)
        - show and hide Elements
            - initially set visibility to hidden
            - to unhide go to window AP Elements

Problem (Using Behaviours with AP Elements)

- jump menu revisited (drop down combo box)
    - in forms tab (don't use jump menu **behaviour**)
    - like a drop down menu (list) but it sends the user to a new URL as soon as the selection is made
    - should be in a form but DW takes care of this
    - click in behaviour tab to edit
    - demo linking to pages or other sites
- Creating a Navigation Menu / ********** New ****** *SPRY* **MENU BAR ********** **
    - intro (basic setup) note: can't add spry menu to a frame setup
    - Link tutorial pg 185 Chapter 5
        - reviews basic links,CSS and using the Spry menu bar
        - introduces styling the menu bar
            - you must edit/add to actual CSS code
            - ul.MenuBarHorizontal a/ul.MenuBarHorizontal li
    - demo other menus in (MenuExamples)

- Other *Spry* widgets (demo ***OtherSprys*** folder)
    - **Tabbed Panel**
    - **Accordion/Panel**
        - demo styling other Sprys eg. .TabbedPanelsContentGroup

Problem (create a simple page that incorporates several Spry widgets)

- Swap images Revisited
    - Basic
        - rollover(***script02***)
            - note: code originally created in 1st page
            - rollovers can be created by using rollover on common tab of DW
            - do simple example right in ***script02*** and note code DW creates
            - make sure comment lines in ***script02*** are removed
        - vs swap images (***script03/04***)
            - with swap images a mouseclick or mouse pass over image is not required to trigger them. You can trigger them from any tag and event
            - again note code originally created in 1st page then
            - do simple example in ***script03*** using DW with behaviours
                - put a new link below arrow and have it trigger image swap
        - using behavior swap image a practical example (***leaders1.htm***)
            - place starter image on web page and name it
            - image to be swapped should be same size
            - select tag
            - attach swap behaviour (onmouseover)
            - from images list click name of starter image
            - set source (image to swap in)
            - then attach swap image restore (onmouseout)
                - set automatically if checkbox checked in swap image dialog
    - Swapping more than one image using a single swap image behavior
        - ***leaders2.htm*** -> illustrates disjoint rollovers
            - from image list choose image holder
            - an asterisk (*) next to the name of an image in the swap image dialog box indicates that the behaviour will swap in a new image for that particular grph
    - Setting the contents of more than one object (image/text) using multiple behaviours and events
        - ***leaders3.htm***
            - set text of textfield behaviour added
            - needs two extra events just for text update (onmouseover/onmouseout)

- Setting Navigation Bar Images
    - recall nav bar and frames lesson (DWFrames)
    - some advanced features of nav bar only available if you use the behaviors panel to edit
    - this behavior can let the state of a nav bar image dictate the state of other images on page (swapping)
    - ***leader4.htm*** (using *Set NavBar Image* Behaviour)
        - first create navbar using icon in common tab *then*
        - click on one of the images in the navbar
        - double click any of the Set Nav Bar Image actions in the behaviour panel
            - that have been automatically placed
        - click **advanced** tab
        - choose **over state**
        - from **also set** click name of image you want this action to replace with another graphic
        - specify image of swap in

Assignment => Create a web page which uses disjoint rollovers via links and navbar (eg. music/sports)

WORK PERIOD

---

(17) DHTML

- combination of HTML,CSS,and Javascript
    - introduce/recap with bookdemos (folder) chapter 23,24 serves as a nice lead into Dyanmic-Drive
- highlight *Dynamic Drive* Site (www.dynamicdrive.com)
    - just demo dw_tooltip
    - demo how to modify -> NavBarMenu (topmenu3)
        - note: readme.txt/most configurations done in custom.js
        - frame version (use master/page1/page2 – menu added to top.html)
        - table version (use table1/table2/table3)

=> Assignment: Show and Tell (DHTML)

- Dynamic Drive
- Dynamic Web Coding (www.dyn-web.com)
- dhtmlgoodies (www.dhtmlgoodies.com)

WORK PERIOD

WORK PERIOD

WORK PERIOD

WORK PERIOD/PRESENTATIONS

WORK PERIOD/PRESENTATIONS

---

e) **Creating Dynamic Web Sites**

(1) Intro

- What is a dynamic website (Amazon.com example)
- Web Server handles requests for Web Pages (Internet Information Server IIS)
- Application Server (usually built in to the Web Server)
  - this is the part that understands the code
  - types: ASP.NET,PHP
- Database Server
  - MS Access (only supports a few simultaneous users)
  - MySQL
  - SQL
- Dynamic web sites contatin HTML and programming code (C#/ASP.NET)

- The application server processes the code and sends a complete HTML page to the Web Server which in turn sends that on to the visitor.
- Often the programming code requires the application server to retrieve info from a database and then merge it with the HTML page.

=> **We do Quiz 1**

(2) ASP.NET using C# and Visual Web Developer Express

DON'T DO FORMAL LESSONS ON C# BUT RATHER TEACH SIMPLE CONCEPTS ON A NEED TO KNOW BASIS

OTHER MATERIAL COMES FROM MAKTUB CASESTUDY (eg. images)

a) Getting Started using Visual Web Developer (chapter 2 Dummies)

- ***CreatingAWebsite*** (Creating a new website -> creates empty page Default.aspx)
  - browse to H: drive and add new folder name
  - creating and using folders in the solution explorer pane
    - eg. Image folder (copy files to this folder)
  - selecting and formatting text
  - adding images
  - design/source view
  - editing in source view with intellisense
  - titling pages
    - in properties -> choose <DOCUMENT>
- ***SimpleCoding1*** (label/button) ASP.NET controls vs HTML
  - ASP.NET button control is coded in C# (when you double click on it)
    - ***code behind file***
      - not HTML but rather C#
      - .aspx.cs
  - HTML button control is coded in Javascript
    - when you double click on it , you are put into HTML source onClick event
    - you then must provide the code for the corresponding function eg. alert (" welcome")
- ***SimpleCoding2*** (adding a second page/start page)
  - note option to place code in a separate file when creating a new page (this adds pg.aspx.cs)
  - add hyperlink that goes to 2nd page (highlight text then click link button on toolbar or drag from expl)
  - you can make the second page the start page by right clicking on it and then setting it as the start page
  - adding HTML elements (HR) from the HTML controls in toolbox

Notes: See vedio-01 Getting Started with VWD ***

- should re-open created **website *folder*** from ***within*** Developer
  - solution file may not be in same location
  - at home it may default to user\my documents\visual studio 2005\projects
  - at school projects should default to H:\visual studio2005\projects

- using the style builder on an object (right click and choose style)

=> We do Quiz 3 (03_teach.pdf)

**** NOTE *******

Visual Web Developer has a built-in viewer for ASP.NET pages you create within the program ***BUT***
You **can't** just open a dynamic page on your own computer (without VWD), before its posted online, as you can a regular HTML page. You must view a dynamic page through a Web Server that has an appropriate application server running.

ASP.NET is the server model we use. It runs in conjunction with Microsoft's IIS Web Server (XP PRO)
- http://localhost

=> Video (Beginners-01) + Then Chapter 2 and 3 tasks (02_teach.pdf/03_teach.pdf)

b) Creating Web forms a Detailed Look

- **Intro**
  - can contain HTML and ASP.NET server controls
  - filename extension is .aspx
- ***Controls*** (see video-02 Creating a web form user interface)
  - adding different web controls (as opposed to HTML)
    - buttons/labels/textboxes/radiobuttons/hyperlinks/dropdownlists/
      - note: hyperlinks can be text/image (change property ImageURL/NavURL)
      - note: **group property** for radiobuttons/button within layer
      - note use of **autopostback true** for DropDown.SelectedIndexChange
    - required FieldValidators/ValidationSummary
  - controlling layouts(DO BEFORE WEB CONTROLS)
    - tables
    - CSS
      - Using Layout Menu/Position-Auto Position/CSS Pos/Absolute
    - Layers
- ***MakTubBandMembersBio1***

=> **We do Quiz 5 (05_teach.pdf)**

c) Events and Postback (see Videos – Beginners 1st 8 minutes) video-03 Understanding more about Events -Postback

- ***Postback***
  - why use IsPostBack property (demo in prg by commenting code in Page_Load)
  - in Page_Load
  - if (Page.IsPostBack= = false)
    - use for page loaded the **first** time
- ***MakTubBandMembersBio2 (display message on opening subsequent requests ban mem info displayed)***

d) Passing Data to Another Page (See pg 8 Chapter 8 - 08_teach.pdf)

- ***NavigatingBetweenPages*** (various techniques)
  - using a linkbutton (Response.Redirect ("page.aspx")
  - creating a sign-in screen
    - query string ?
    - using Request.QueryString[ ]

=> Videos (Beginners-02) + (Minor Assignment – Create a two page site highlighting your favourite band – Main is login and Second pg is Bios)

---

e) Working with style sheets and master pages (for HTML controls)

- Creating a **Style Sheet** (***StyleSheet1***) (Chapter 6 Dummies)
  - create a folder called stylesheet
  - right click folder and add new item => stylesheet
  - using the CSS editor
    - position cursor just to the right of the opening curly brace of the selector you want to style
    - choose add style rule (element/class -> left toolbar)
    - build style (style builder)
  - Linking to a style sheet
    - drag stylesheet to upper left corner of page (eg. Master page)
      - document property (stylesheet)
      - element property (class)
    - to apply class selector must go into source and type <span class= >
- Master Pages (similar to template)
  - ***MasterPage1*** (Chapter 4 Dummies)
    - create a folder called master
    - add item => master page (note ContentPlaceHolder)
    - design layout (tables -> template/custom)
      - choose template (Header/SideLayout)
      - may need to move PlaceHolder into appropriate cell on page
    - styling master page panes (using stylebuilder)
      - style each pane
      - ContentPlaceHolder pane
        - make sure mouse is in same cell as placeholder but not on placeholder
        - then choose text vertical alignment (top)
        - also pad pane (edges)
    - using a master page
      - create a new web form
      - choose select master page checkbox
    - editing a master page (note page updates)
  - ***MasterPage2*** (See Video Beginners-06)
    - note use of Navigation controls
    - Menu control
    - Treeview control
      - must first add new item -> **Site Map**
      - then go to data source and bind to site map file
      - then click site map and refresh schema

=> Video (Beginners-06) + Create a style sheet for the BandBios.aspx page within the NavigatingBetweenPages site -> NavigatingBetweenPagesStyled

---

f) Using Data in Web Pages

Introduction: Manipulating database records using Access

1) Intro (terminology)
- Database, tables (files),records,fields
- relational vs flat file databases

2) Creating a Database (people.mdb)
-startingAccess and creating a new database table
-Table Design View
-Defining Fields
-primary key field - eg. SIN (no duplicates)
-data types (text/number) /field properties (size,format for numbers,indexes)
-modify the structure of a table (change/remove/add fields)
-Data Sheet View (working with data)
-adding,delete new records/editing records/finding records /resize,reorder columns
- cursor must be in correct field then use binoculars
-Saving a Database/Reopening

3) Opening existing databases from ccdemos (demos97)
- when using demo97 database, students should first create a new database in h: drive then import specific tables from within it.

4) Creating Select Queries from a single table (selquery.mdb) (students name db selquery.mdb and import req tables)
- Intro (**people** table)
- simple queries and dynasets
- display name and SIN (using ! to run query)
- Using Selection Criteria (**emp**)
- CA
- employee # < 6850
- shift 1 AND zip 12345
- CA OR NY / CA or SHIFT 2 (Put 2 on line below criteria OR)
- BETWEEN ... AND (employee # 6840-6860)
- using wildcards (*)(cities A*/*A/*A*/soc sec 987*)
- creating parameter queries (first name) -> use criteria [Enter First Name]
- Using (**sales17**) (optional)
- list sales between the dates of 2/1/92 and 5/1/92
- # signs around the element of an expression indicate that the element is a date/time , not a text string
- automatically put in by Access

5) Sorting Records(**emp**)
-Ascending/Descending (state) /grouped(shift-descending,last name-ascending)

=> We do Chapter 9 Quiz

=> Chapter 9 Task

---

6) Designing Forms and Reports from a single table (formrpt.mdb)

- intro (nwind.mdb) (see also employees form)
- students name db people.mdb and import people table from demo97.mdb
    - view people table in datasheet view
    - then use **formrpt.mdb** to illustrate the various concepts
- Creating a new form using a Form Wizard (single column/tabular/new tabular)
    - Note: the info in a form/report may be from either a **table** or a **query**
    - name,address,city,postal code (people)
- Modifying the generated form design (form design window)
    - Anatomy of a form (header/detail/footer)
    - Modifying the tabular form of the people table
        - moving and resizing controls
        - changing label prompts, text appearance (using properties/right mouse)/toolbox
        - adding fields/changing properties
        - changing colors (using palette) of labels/fields/backgrounds
- Adding/Deleting records in form view (new➔*)
- Printing datasheets and forms
- Creating a Simple Report using a Report Wizard (single column/tabular/mod tab/sorted mod tab/Kitch resid - based on query)
    - Anatomy of a Simple Report (report hder/page hder/detail/page fter/report fter)
    - modifying the generated report (report design user interface)
    - align,size controls/changing properties
    - inserting pictures (size mode - stretch)
    - adding labels in the header with the toolbox/inserting pictures - image control (size mode - stretch)/adding fields
    - sorting (see sorted modified tablular report in report design view)
    - choose preview then choose print

7) Calculations and Summary Statistics

Calculated Query Fields (artificial fields) in select queries (calc2.mdb)(use **employ**)

- Net pay calculation
    - use only the name field from the employ table
    - then create calculated field net pay-> net pay:[salary]-[deductions]
    - change format of net pay to currency (in query design view)
- Querying a query (that contained calculated fields)
- can then be used as the basis to a form/report (**net pay 25000- 30000**) **OR** in **report** properties **recordsource** (in design mode right click the empty box in the top left corner of the report window)you can perform a required query(using the query builder) on an existing table/query (eg. display only people with **more than 20 yrs experience**.

Making a Queries That Calculate several Totals at once Σ / Subtotals (calc3.mdb) (**models** table/ **sales** table)

- query using count,sum (total model count/total value sum)
- query using **group by** and **sum**/ **group by** and **count**
    - models table (sub-total of value based on manu/count of computers by manu)
    - intro with values sorted by manu (no grouping)
- sales table (sub-total amount sold based on salesperson)
    - intro with amount sorted by salesperson (no grouping)

Intro to SQL

```
SELECT [FIRST_NAME],[LAST_NAME],[STATE],[SHIFT]
FROM EMP
WHERE [STATE] = "CA" OR [SHIFT]=2;
```

Using Expressions (calculations) and Totals (summary statistics) in Forms and Reports

- Building Calculated Controls and Summary Statistics in Forms(calc4.mdb)
- use tabular form wizard (standard)
- based on a query (net pay query calculation from employ table)

****** **this is the best method** *********

- add text box (unbound field) enter formula into control source (using expression builder)
    - change autolabel to NO/click textbox on toolbox then prop on menu bar
- total net pay => =sum([net pay])
- average age => =avg([age])
- maximum experience => =max([experience])
- note the summary statistics are placed in the footer

- iif (immediate if command)
    - =iif([net pay]>=25000,"high","low")
- sorting and filtering by form (don't forget to apply filter - press button)
    - net pay(descending)/age(ascending)
    - net pay (25000-30000) (filter by form)

=> Student Averages (Students can import stdav table from stdav.mdb)

**** Create the form based on a query calculation that determines the student averages *****

8) Relational Database Concepts

- Flat file vs Relational (customer order example : 1 table vs 3 tables -> customer info/parts -static /orders-dynamic)

- Intro (generic example with two tables: [sin,name,address,telephone,city]/[sin],[amount])
- define primary key/define relationship/enforce ref int/perform simple query
- Then create form based on dynamic and add combo using query builder idea

- Forms with Multiple linked tables
- advanced combo boxes (Do on same form)
- using multiple columns allow you to display related data from other tables
- use query blder in row source/ **col cnt->3/col wid-> 1;1;1**
- display names but store SIN (column width 0) (query linked combo box)
- col cnt 2/col width 0;1/list width 2

- chocolate bar campaign
-Master (#,name, hr, tel) /Dynamic (#,date,amt)
- query
- report (summary/detail)
- form (combo box – name,#)

- The School Board Computer Inventory (schcomp.mdb) (items/models/schools)
Students are to take the 3 tables items,models and schools and **define** all the necessary **primary** keys and **relationships** so that a relational DB is created. Then a **Form for item entry** is to be created which makes use of the relational aspects of the newly created DB. Also produce a **reports** for **model value** totals grouped by **school** - detail and summary, and model total values summarized by school-summary only.(Based on 3 table query of school name,model,value)
Make location a combo box,displays full school name but stores location #, use query builder to create row source from school table (loc,school name fields). Make model combo using query builder to create row source from models table (model field) /Have value automatically appear when model chosen

******** **(Base form on a query betw *items* and *models* with all fields from *items* and just value from *models*)**

Computer Science Day Problem (Remember rate/hours number fields of type *single*) **HW Assignment**

Database Driven Web Pages

1) ***DataBinding1*** (See Videos-Beg-07)
- adding a sql database within Visual Web Developer- Must create on C:drive at school -> call it **Customers**
- note database explorer/solution explorer tabs
- add a new table then add fields and data types
  - customerID/FirstName/LastName/CustomerSince/CreditLimit
  - set identity specification for customerID and make keyfield
- save table (customer.mdf) and enter data (right click/show table data)
- add data source to web form (SqlDataSource)
  - conifigure data source (customer.mdf)
  - configure select statement (*)
  - creating views (queries)
- adding a **gridview** to display data (like datasheet in Access – choose data source SqlDataSource1)
- adding a **details view** to a new web form (like form in Access)
  - after initial SqlDataSource is added to a web form that is part of the Site this addition is not required on any further forms. You only need to add the desired control . VWD will automatically add the SqlDataSource
  - new data source (DB) + connection string
  - autoformat/enable paging
- **gridview and detail view** together
  - configure gridview first then details view (SqlDataSource1)
  - enable selection in gridview
    - remove CustomerID/CreditLimit/CustomerSince columns
    - no paging
  - write code for gridview control's SelectedIndexChange Event
    - DetailsViews1.PageIndex=GridView1.SelectedIndex

=> We do+ Chapt 10 DB-Driven Band Bio Page Using Gridview with premade Access DB (place in AppData folder)
=> remember to add AccessDataSource first BandDataBase.mdb (Maktub folder - artist table)
=> Video (Beginners 07) + Chapter 11 Task (tours table)

---

2) ***DataBinding1A*** (See Videos – Beg 08)
- **SQL data source** (customers.mdf)
  - need to make proper connection string
- use ***formview***
  - allows editing of data instead of just viewing data (edit/delete/new)
  - configure data source (only for formview not SqlDataSource)
  - choose *
  - then go into advanced when databinding
    - generate insert/update/delete statements
    - use optiminstic concurrency`
    - database must use **key fields** if you want to be able to edit the table thru web pg
  - demo how actual db is updated when changes are made to web form (show table data)

3) ***DataBinding2***
- **Access data source** (use people2000.mdb)
  - using the Data Configuration Wizard
  - when configuring data source use Access DB and name specific table
- use **gridview**
  - illustrate sorting/paging (in prop can change pagesize to 3 per page)
  - demo auto updated data by updating data base and refreshing the web site page
  - problems with updating Access DB from web page (write permissions – locked file)

=> Video (Beginners 08)

---

4) ***DataBinding3*** (See Chapt 12 pg 280 Dummies - Using DataInWebPages.pdf)
- using a DropDownList to filter records in another data bound control (DetailsView)
  - populating a DropDownList
  - enable postback
  - DetailsView1.PageIndex=DropDownList1.SelectedIndex
- when adding a second control to a page choose <New Data Source> and don't use same source the first control uses. By default the new data source is called SqlDateSrc2
- when you choose the connection string you want to reuse the same connection string you use for everything else. That's because the connection string only tells the control where the database is located, it doesn't specify any tables or views within the database.

g) Configuring, Building and Deploying a Website (See Video – Beginners 10)

h) Sample ASP.NET Sites
- Start Kit – PersonalDesign1

=> Quiz

SJB/FEEDER SCHOOL WEB SITE CREATION

22 WORK PERIODS

To search for a term, click the Search button. Then type the term and click Search. You can then choose from the list of topics.

Finding more help on the Web

Adobe offers support on its Web site. Choose Help➪Flash Support Center, which takes you to `www.adobe.com/support/flash`. There you can search the knowledge base and tutorials for answers to your questions.

Try It; You'll Like It

Perhaps by now you're getting impatient to try out Flash. Getting started is easy. You collect a few ideas, put together some art, add animation, save your movie, and publish it. Then you view it in a browser either online or offline. That's the gratifying part. In the following sections, you get to try out Flash by working through a basic animation. The rest of the book explains these concepts in more detail.

Conceiving your first animation

Suppose that you want to add an animated logo to a home page that you've already set up. You want the animation to run when the page loads and then stop. Figure 1-4 shows the Rainbow Resources company logo — unanimated, of course — that you can find on this book's Web site, at `www.dummies.com/go/flashcs3`.

Figure 1-4: A company logo that could stand some animation.

Suppose that you want the word *Rainbow* to fly into your page from the right and the word *Resources* to fly in from the left. At the same time, you want the graphic to rotate 180 degrees. The following section shows you how to create this animation.

Creating flashy drawings

You can use Flash to create a company logo, but importing one from this book's Web site is simpler. Often, you import existing graphics (such as a company logo) from a file rather than create them from scratch. (Chapter 3 explains how to import and manipulate graphics.)

If you're going through the steps and make a mistake, choose Edit➪Undo (or press Ctrl+Z or ⌘+Z) and try again. You can use Undo repeatedly to undo several steps, if necessary.

To import the Rainbow Resources logo into Flash, follow these steps. (The steps might vary if you're importing some other graphic in a different format.)

1. **Start Flash.**

   See the instructions in the section "Starting Flash on a PC" or "Starting Flash on a Mac," earlier in this chapter, if you need help.

2. **In the Create New section of the Startup screen, choose Flash File (ActionScript 3.0).**

   You see a spanking-new movie on your screen.

3. **Go to `www.dummies.com/go/flashcs3` and download the `rainbow.gif` image file.**

   You'll need to unzip the file and extract the image file. Extract this image file to the location where you plan to save your Flash movie.

4. **Choose File➪Import➪Import to Stage.**

   The Import dialog box opens.

5. **Browse the dialog box until you find `rainbow.gif` in the location where you saved it, and then double-click the file to open it.**

   You see the logo on your screen. You need to break the logo into pieces and make it a vector graphic so that you can animate sections of it separately.

6. **Choose Modify➪Bitmap➪Trace Bitmap.**

   The Trace Bitmap dialog box appears.

7. **In the Trace Bitmap dialog box, set the color threshold to 100, the minimum area to 1, the curve fit to Pixels, and the corner threshold to Many Corners. Click OK.**

   In our example, we chose to use settings that reproduce the bitmap as faithfully as possible. Flash creates a vector graphic and breaks up the graphic into individual components. The entire graphic, however, is selected.

8. **Click anywhere outside the graphic to deselect it.**

You've got your logo! Now you need to set it up for animation.

Turning your objects into symbols

In the logo that you imported in the preceding section, each letter is a separate object, which can get pretty confusing. Each line in the logo's design is also separate. But you want your words — and the little design — to stay together. So you must combine each word and the logo into a symbol. A *symbol* helps keep objects together and is required for some kinds of animation. (See Chapter 7 for the scoop on symbols.)

To turn the words and the logo into symbols, follow these steps:

1. **To get a better view of your image, click the Zoom Control drop-down list (at the upper-right corner of the Stage area) and choose 400%.**

   If you don't see the Zoom Control drop-down list, choose Window➪Toolbars➪Edit Bar to display it.

   Use the scroll bar to scroll the words of the logo into view, if necessary.

2. **Click the Selection tool on the Tools panel if it's not already selected.**
3. **Click the upper-right corner of the word *Rainbow* (just above and to the right of the *w*) and drag to the lower-left corner of the first letter, *R*.**

   Dragging from right to left makes it easier to avoid selecting the logo at the same time. The entire word should be selected. If it isn't, click outside the word and try again.

4. **Choose Modify➪Convert to Symbol. In the Convert to Symbol dialog box, click Graphic for the Type and then click OK.**

   It's usually good practice to name the symbol, but doing so is not necessary for this exercise. When you click OK, Flash places a box around the word so you can see that it's one object.

5. **Repeat the procedure outlined in Steps 3 and 4 with the word *Resources*.**

   In this case, you might want to start clicking and dragging from the upper-left area of the word; then choose Modify➪Convert to Symbol again and click OK. Now all the letters of the word *Resources* are a single object.

6. **Click the Zoom Control drop-down list and choose 100% so that you can see the entire logo.**
7. **Click above and to the left of the logo and drag to the lower right to select the entire logo.**
8. **Hold down the Shift key and click each word to remove both words from the selection.**

   Now the design portion of the logo is selected.

9. **Press F8 (the keyboard shortcut to create a symbol) and then click OK in the Convert to Symbol dialog box.**

   Flash creates a symbol from the lines of the logo's design.

See Chapter 7 to find out more about what symbols are and how to use them. Symbols are important building blocks in Flash movies.

Putting your graphics on layers

You need to place different pieces on different layers when you're animating. You use layers to organize your movie and keep shapes separate so that they don't interfere with each other. (See Chapter 6 for the complete story on layers.)

To split your three symbols onto three separate layers, you can use a convenient feature of Flash CS3: distribute to layers. Follow these steps:

1. **Click the Selection tool on the Tools panel if it's not already selected.**
2. **Drag diagonally across the entire logo, including the two words, to select it.**

   You should see two rectangles inside one bigger rectangle. All three objects in the logo are selected.
3. **Choose Modify➪Timeline➪Distribute to Layers.**

   You now have three new layers, named Symbol 1, Symbol 2, and Symbol 3. The three objects of the logo have been distributed to Symbol 1 through 3 and removed from Layer 1.
4. **Click outside the Stage to deselect any objects.**

You're now ready for the animation process.

Making graphics move

We explain earlier in this chapter that your goal is to have the word *Rainbow* fly in from the right and the word *Resources* fly in from the left. You also want the graphic to rotate 180 degrees at the same time. What you see now is how the animation will end — the last frame of the movie.

Follow these steps to create the last frame of the movie and save the file:

1. **For *each* of the three symbol layers, click frame 30 of the Timeline and choose Insert➪Timeline➪Keyframe.**

   You may have to scroll down to access frame 30 on the lowest layer. You can find out more about keyframes in Chapter 9.
2. **Choose File➪Save and choose the same location you used for the `rainbow.gif` image file.**

   We don't recommend saving the file in the Flash CS3 program folder — it might get lost among your Flash program files.

3. **Give your movie a name, such as *Movie of the Year,* and click Save.**

   Flash creates a file named `Movie of the Year.fla`. Flash adds `.fla` for you because that's the filename extension for Flash movies.

Go back and create the beginning of your movie. Flash can fill in all the blanks in between. Follow these steps to create the beginning of the movie and the animation:

1. **If the Property inspector isn't already open, choose Window➪Properties➪Properties to open it.**

   If the Property inspector is open but collapsed, click its title bar to expand the panel.

2. **Select the word *Rainbow.* Click the first frame of the Timeline in the highlighted row.**

   When you select the word *Rainbow,* you can tell which layer it is on by looking at the highlighted layer.

3. **Press and hold down the Shift key while you drag the word *Rainbow* to the right, just off the Stage into the gray area.**

   You might need to use the horizontal scroll bar or choose a lower zoom percentage in the Zoom drop-down list to see the gray area. Pressing Shift keeps the object from moving up or down while you drag to the right. By clicking the first frame and moving the word, you set its position at the beginning of the animation.

4. **Click the word *Rainbow*'s layer, anywhere between the 1st and the 30th frame.**

5. **On the Tween drop-down list of the Property inspector, choose Motion.**

   You now see a solid arrow on the Timeline between the 1st and 30th frames. Choosing Motion in the Tween drop-down list creates animation that moves the symbol from the position in the first frame to its position in the next keyframe, which is frame 30 in this movie.

6. **Repeat Steps 2 through 5 for the word *Resources.* However, in Step 3, drag the word *Resources* to the left, just off the Stage.**

7. **Select the logo design and click the first frame of the Timeline in the highlighted row.**

8. **Choose Modify➪Transform➪Rotate 90° CW to rotate the design 90 degrees clockwise.**

9. **Repeat the Modify➪Transform➪Rotate 90° CW command to rotate the design a total of 180 degrees.**

10. **Click the logo's highlighted layer, anywhere between the 1st and 30th frames, and choose Motion in the Tween drop-down list of the Property inspector.**

11. **If necessary, drag the horizontal scroll box until the Stage is in the center of your screen.**

    Otherwise, you won't be able to see the entire animation — and you don't want to miss this one!

12. **Click the first frame of any layer.**

    This takes you to the start of your movie. Your screen should look like the one shown in Figure 1-5.

13. **Press Enter (Return) and watch the animation. (Start writing your Academy Award acceptance speech.)**

14. **Save your movie again by choosing File➪Save.**

Publishing your first animation for posterity

You can't watch the animation in a Web browser until you publish it and insert it into an HTML document. To do so, follow these steps:

1. **Click the Stage to change the display of the Property inspector.**

    You should see the Settings button next to the Publish label.

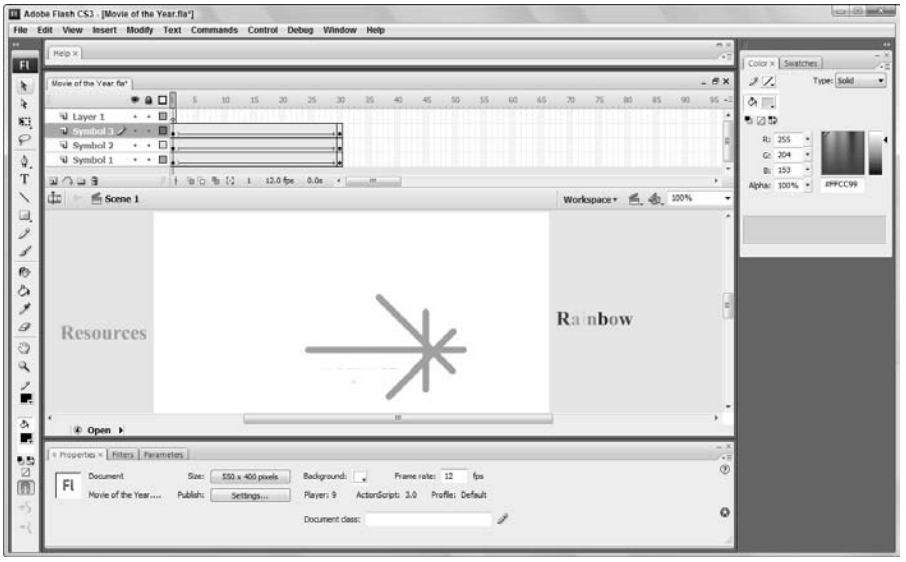


Figure 1-5: Before you run the animation, *Rainbow* appears to the right and *Resources* to the left, and the line logo is rotated.

2. **Click the Settings button in the Property inspector.**

   The Publish Settings dialog box opens.

3. **Click the HTML tab.**

4. **Deselect the Loop check box in the Playback section.**

   We want the animation to play only once.

5. **Click the Publish button, and then click OK to close the dialog box.**

   With scarcely a blip, Flash publishes your movie and creates two files, named `Movie of the Year.swf` (assuming you used that name) and `Movie of the Year.html`. They're in the same folder as your `.fla` movie file. `Movie of the Year.swf` is the file your browser reads to play the animation. `Movie of the Year.html` contains the HTML code required to display your movie on a Web page.

6. **Open your Web browser.**

7. **Choose File➪Open (or Open File) and find `Movie of the Year.html` (or whatever you named your movie file).**

   You might need to click Browse and navigate to the file.

8. **Double-click the file.**

   Your browser opens the HTML document and reads its instructions to play the Flash movie.

9. **Sit back and watch it roll.**

   Don't blink or you'll miss it. (If you do miss it, click the Refresh or Reload Current Page button in your browser.) You can see the movie in Figure 1-6.

10. **When you finish watching the movie, close your browser.**

You can find the Movie of the Year files (`.fla`, `.html`, and `.swf`) on this book's Web site, `www.dummies.com/go/flashcs3`.

Exiting Flash

When you finish creating something in Flash, choose File➪Exit (Windows) or Flash➪Quit (Mac).

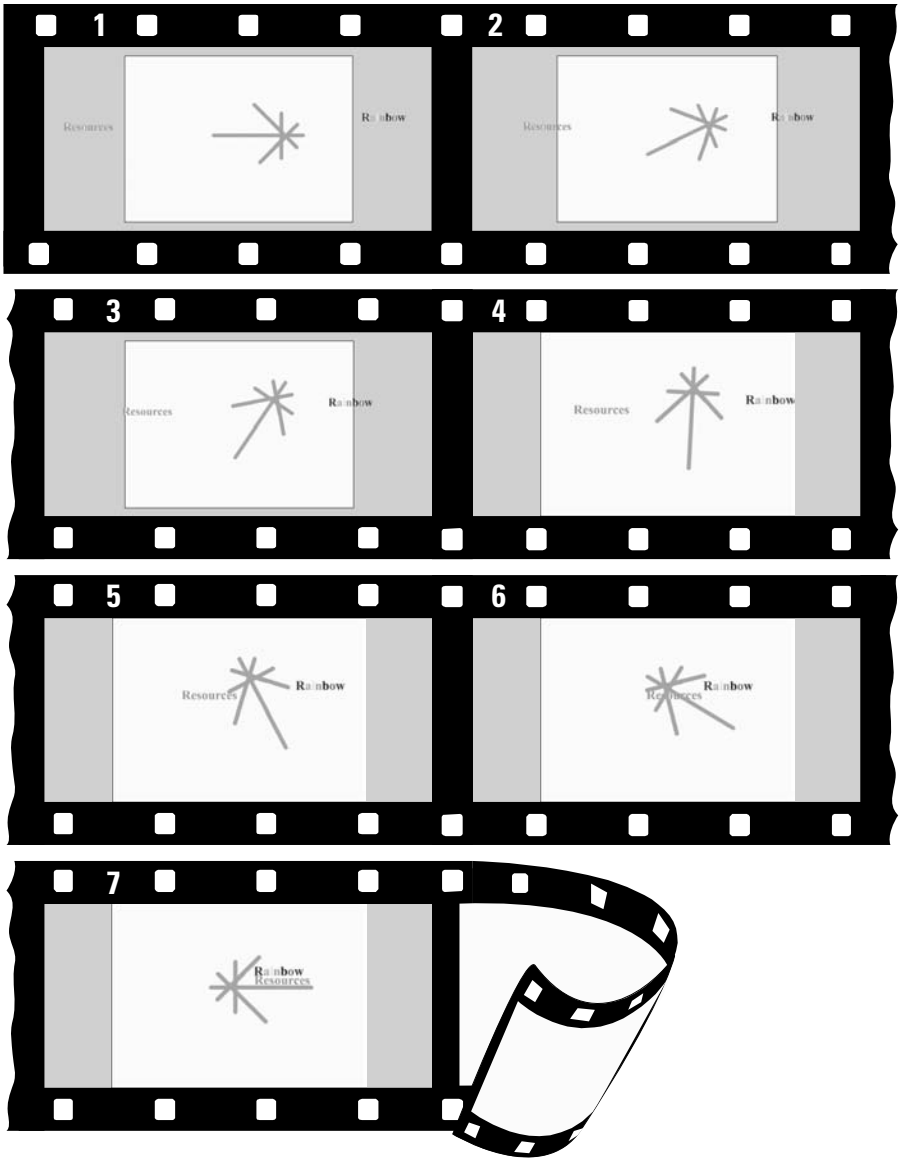


Figure 1-6: The Movie of the Year animation in detail.

Exercise 2 — MEASUREMENTS

The MEASUREMENTS Web site was last modified in Chapter 8, Exercise 2. Modify the MEASUREMENTS Web site by completing the following steps:

a) In Dreamweaver, open the MEASUREMENTS Web site for editing.

b) Modify the index.htm Web page document as follows:

1. In the empty cell in the right side of the third row, place the insertion point.
2. Insert tabular data, using the si_base_units.txt tab-delimited file, a table width that fits the data, a cell padding of 6, no cell spacing, a border of 1, and format the top row of data as bold.
3. Near the bottom of the Web page document, place the insertion point after the period in the text table at right, and press Enter.
4. Type the following text, allowing the text to wrap:
   For more information on the International System of Units, visit the Web site of the Bureau International des Poids et Mesures (BIPM), which is based in France.
5. Link the text Bureau International des Poids et Mesures (BIPM) using an external hyperlink to the following URL:
   http://www.bipm.fr/enus/welcome.html

c) Modify the prefix.htm Web page document as follows:

1. Near the bottom of the Web page document, place the insertion point below the text SI Prefix Chart.
2. Insert tabular data, using the si_prefixes.txt tab-delimited file, a table width that fits the data, a cell padding of 6, no cell spacing, a border of 1, and format the top row of data as bold.

d) Modify the derive.htm Web page document as follows:

1. Near the bottom of the Web page document, place the insertion point below the text SI Derived Units Chart.
2. Insert tabular data, using the si_derive.txt tab-delimited file, a table width that fits the data, a cell padding of 6, no cell spacing, a border of 1, and format the top row of data as bold.

e) View each Web page document in a browser window.

f) Print a copy of each Web page document from the browser.

CHAPTER
22

Getting Started with Dynamic Web Sites

So far in this book, you've learned to build and maintain Web sites using Dreamweaver's powerful design, coding, and site management tools. The pages you've created are straightforward HTML, which you can immediately preview in a Web browser to see a finished design. These kinds of pages are often called *static*, since they don't change once you've finished creating them (unless you edit them later, of course). For many Web sites, especially ones where you carefully handcraft the design and content on a page-by-page basis, static Web pages are the way to go.

But imagine landing a contract to build an online catalog of 10,000 products. After the initial excitement disappears (along with your plans for that trip to Hawaii), you realize that even using Dreamweaver's Template tool (Chapter 19), building 10,000 pages is a lot of work!

Fortunately, Dreamweaver offers a better and faster way to deal with this problem. Its dynamic Web site creation tools let you take advantage of a variety of powerful techniques that would be difficult or impossible with plain HTML pages. With Dreamweaver, you can build pages that:

- Display listings of products or other items like your record collection, your company's staff directory, or your mother's library of prized recipes.
- Search through a database of information and display the results.
- Require login so you can hide particular areas from prying eyes.
- Collect and store information from visitors to your site.
- Personalize your visitors' experience: "Hello Dave, it's been a while since you've visited. Did you miss us?—Hal."

Visit Amazon.com, for example, and you'll find more books than you could read in a lifetime. In fact, you'll find more products—DVDs, CDs, and even outdoor lawn furniture—than could fit inside a Wal-Mart. In just an hour, you could browse through hundreds of products, each with its own Web page. Do you really think Amazon hired an army of Web developers to create each Web page for every product they sell? Not a chance.

Note: Luckily you aren't limited to *either* "static" *or* "dynamic" Web pages. Web sites frequently contain both–static pages for custom designs and handcrafted content, and dynamic pages for mass production of a thousand catalog pages.

Instead, when you search for a book on Amazon.com, your search triggers a computer program, running on what's called an *application server,* which searches a large database of products. When the program finds products that match what you're searching for, it merges that information with the HTML elements that make up the page (banner, navigation buttons, copyright notice, and so on). You see a new Web page that's been created on the spot—perhaps for the first time ever (Figure 22-1).

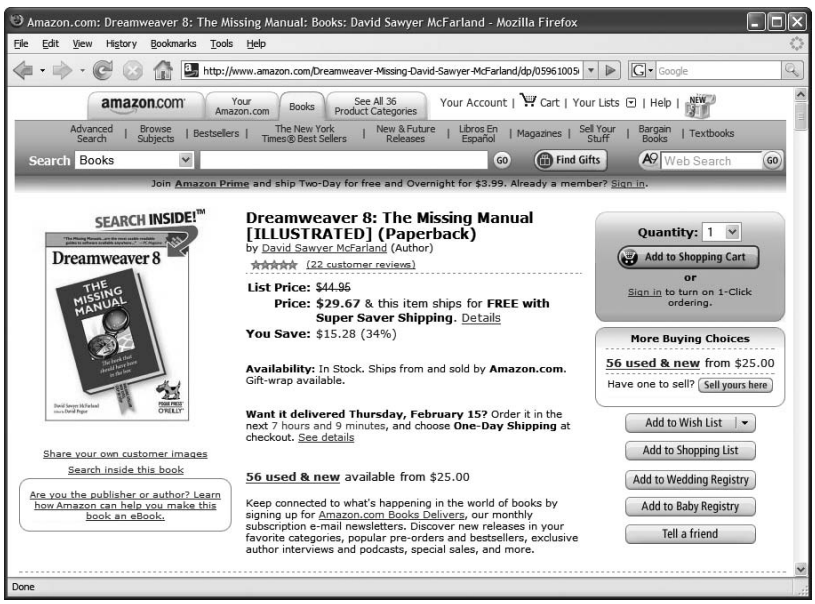


Figure 22-1: *An infinite number of monkeys couldn't create all the Web pages for all the products Amazon sells. The solution? A dynamic Web site, which takes your programmed instructions (cooked up with a little help from Dreamweaver), and automatically creates pages made up of content chunks pulled from a database. That's the way to go if you've got a site with loads of pages that all present similar information.*

Dynamic Web sites are usually the realm of professional programmers, but Dreamweaver can simplify routine tasks like viewing information from a database and adding, updating, and deleting data. Even if you don't have a programmer's bone in your body, this chapter and the next few give you the basics.

Pieces of the Puzzle

You may be thinking, "Yeah, that sounds fantastic, but so did that time-share in the Bahamas. What's the catch?"

The catch is that dynamic Web sites are more complex and require more technologies to get off the ground. Simple static Web sites require only the computer you use to build them, and a Web server to dish them out. In fact, as you can see by previewing your site from your own computer with a Web browser, you don't even need a Web server to effectively view a static Web site.

Dynamic Web pages, by contrast, require more (see Figure 22-2). Not only is there a Web server that handles requests for Web pages, two other types of servers enter the equation: an *application server* and a *database server*.

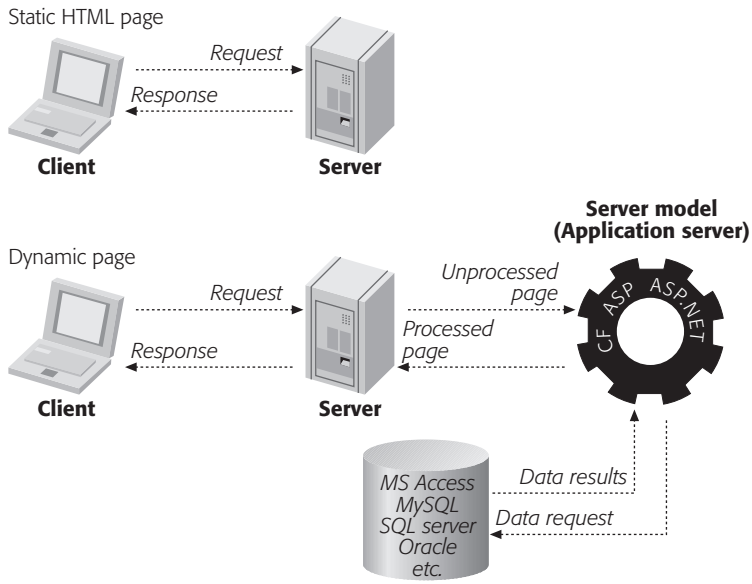


Figure 22-2:
Top: The technology behind a Web site that uses plain old static HTML pages is straightforward: When you come to the site, the Web server sends your computer (the client computer) the Web pages, graphics, and other elements of the page.

Bottom: When you come to a dynamic page, on the other hand, the Web server must communicate with an application server, which in turn communicates with a database server. The final results come back to you through the Web server.

You'll still be using a lot of HTML (and CSS) to build a dynamic site—for example, to provide the layout, add banner graphics, and navigation bars. But you'll augment this mix with some form of programming code. The application server processes this code and sends a complete HTML page to the Web server, which, in turn, sends that onto the visitor. In many cases, the programming code requires the application server to retrieve information from a database, and then merge it with the HTML of a page.

Note: In this context, a *server* refers to software that dishes out particular types of information–Web pages, database queries, or a program's output. It doesn't necessarily mean a separate computer; Web hosting firms can (and frequently do) have Web, database, and application servers all running happily together on a single machine.

Because dynamic Web sites require more technology, you can't just open a dynamic page in your Web browser as you can a regular Web page. You must view a dynamic page through a Web server that has an appropriate application server running.

You also have to set up a database, and connect that database to your application server. Although this can be quite complex, it's not difficult to set up a basic Web server, application server, and database on your own computer, so that you can build and test database-driven Web pages. It's also easy to connect to other computers that are already configured to serve up dynamic, database-driven Web pages.

And once you or your company's system administrator have set up the Web server and other assorted components, Dreamweaver can easily create complex Web pages that access databases, and let you build powerful Web applications, all without ever learning any programming.

Note: The term *Web application* refers to Web pages that work together to complete a task. All the various pages that come together to form an online shopping site–which lets visitors do things like search a database of products, view individual product pages, and add products to a shopping cart–would be considered a Web application.

FREQUENTLY ASKED QUESTION

The Dynamic Duo

How does a dynamic Web site differ from dynamic HTML?

Dynamic is a word that's thrown around a lot in Web circles, and it has a variety of uses.

For starters, *dynamic* sometimes refers to the power of JavaScript. For example, Dreamweaver CS3's Spry Framework uses JavaScript to create interactive page elements such as the Spry menu bar discussed on page 175, or the animated effects described on page 511. The result is sometimes called "Dynamic HTML," because the elements on the page *change*.

However, in this section of the book, *dynamic* refers to any Web page that's processed by an application server–pages that undergo some form of transformation on the Web server's side of the Internet, like connecting to a database, or collecting information from a form.

What's important to remember is that JavaScript, used for Dynamic HTML, Spry, and Dreamweaver Behaviors (Chapter 13), is a *client-side* programming language. It runs in someone's Web browser, and is limited to changing the way a Web page looks and behaves *after* it's been downloaded over the Internet.

Dynamic Web sites, on the other hand, use *server-side* programs–those that run on an application server, out there on the Web somewhere. The dynamic part (responding to a form or accessing a database, for example) happens *on the Web server*. The visitors to your site never see any programming code, and their computers never have to run the program. They merely enjoy the results of the application server's hard work: a finished HTML page.

Even so, there are literally hundreds of combinations of Web, application, and database servers, and Dreamweaver doesn't work with all of them. However, it's capable of working with five of the most popular and powerful combinations, using seven different programming languages!

Understanding Server Models

In Dreamweaver lingo, the different application servers combine with a programming language to create a *server model*. Dreamweaver recognizes several server models, including ASP and .NET (pronounced "a-s-p" and "dot net"), ColdFusion, PHP, and JSP. Each server model has its own set of unique requirements, and its own methods of performing identical tasks.

Each server model also works with one or more programming languages. For example, you can create an ASP page using either VBScript or JScript. In some cases, the server model understands only a single programming language: JSP pages, for instance, use the Java programming language. And to make things just a bit more confusing, PHP can refer both to a programming language named PHP *and* to the application server. Likewise, CFML, or ColdFusion Markup Language, is a programming language, and ColdFusion server is an application server. If your head is hurting trying to make sense of all this, just keep this in mind: An application server processes programming code and carries out various actions, like talking to a database or spitting out a Web page.

Which server model you use depends on which resources you have available: which type of Web server hosts your site, the operating system it uses, and which application server is available. If you're hosting your site on Linux or Unix, you'll most likely end up using PHP; if you're hosting on Windows, meanwhile, you've already got access to ASP and probably .NET. It all comes down to what you have on your computer, what your company uses, or (if you're using a Web hosting service) what the host computer understands. Here's a brief description of each server model.

PHP

PHP (PHP Hypertext Preprocessor) is a programming language that was created specifically for building dynamic Web pages. It's the most popular and widely available option at Web hosting companies—in other words, when it comes time to place your finished Web site on the Internet, you're most likely to find a Web hosting company that supports PHP. (PHP is also quite often the least expensive hosting option.) The *PHP interpreter*—that's the application server—works in conjunction with a variety of Web servers, including Microsoft's Internet Information Server (IIS), but was initially created for the Apache Web server. PHP can also work with a variety of different database servers, but Dreamweaver understands only the MySQL database server (which is also available at nearly every Web hosting company).

Note: Because Apache, PHP and MySQL are so commonly used together, you may encounter the abbreviation frequently used to describe them–AMP.

Apache, PHP, and MySQL are free (one of the reasons they're so popular), and you can find simple installation programs that let you install all three programs on your own desktop computer. The tutorials for this section of the book use the PHP server model.

.NET

.NET is a Microsoft server technology. It's actually an entire suite of technologies intended to integrate many activities over the Internet.

.NET is also called ASP.NET because it was created as a replacement for Microsoft's older ASP (Active Server Page) technology. .NET can be programmed in numerous languages, including Microsoft's VB.NET, C# (pronounced "see sharp"), and JScript.NET, as well as more than 20 other languages. Dreamweaver recognizes only the C# and VB languages.

.NET runs in conjunction with Microsoft's Internet Information Server (IIS) Web server. IIS comes with Windows XP Professional and some versions of Windows Vista. Unfortunately, it doesn't come with Vista Home Basic, the most common version of Vista. You can find the .NET framework at *http://msdn2.microsoft.com/en-us/netframework/*.

.NET can work with a variety of different databases, with Microsoft's SQL Server being the most common.

ASP

ASP (Active Server Pages) used to be one of the most common ways to start building database-driven Web sites. It's a bit long in the tooth now, and is probably not the best choice if you're just starting with database-driven sites. ASP understands two different programming languages: VBScript and server-side JavaScript, both of which Dreamweaver speaks fluently. ASP also works with Microsoft's Web server—IIS.

ASP can work with a variety of databases. For small projects, you can use Microsoft Access (the database program that comes with some versions of Microsoft Office), since it's fairly easy to use. For more demanding projects, where you need to store lots of data, and many people will access your site, Microsoft's SQL Server is a better choice.

ColdFusion

ColdFusion is an application server from Adobe (the maker of Dreamweaver) that's programmed using CFML (ColdFusion Markup Language). ColdFusion works in conjunction with several different Web servers, including IIS and Apache, and uses its own programming language, which resembles HTML. For this reason, some Web designers find it easier to learn than other programming languages.

The downside is that this application isn't free. You *can* download a developer's edition—a free version that runs on your computer—so you can build and test ColdFusion Web pages. But if you want to host the Web site on the Internet, you have to either buy the ColdFusion Server package (which isn't cheap) or find a Web hosting company that offers ColdFusion hosting. Fortunately, in recent years, more and more Web hosting companies have started offering ColdFusion as an option, at rates that are close to or match regular hosting plans.

The Developer Edition of ColdFusion is available for download at *www.adobe.com/go/devcenter_cf_try*.

Like ASP and ASP.NET, ColdFusion works with many different databases.

JSP

JSP (JavaServer Pages) is based on Sun's popular Java programming language. It requires a Java application server, like Adobe's JRun server.

This approach isn't for the faint of heart, however, since setting up a Java Server and connecting it to a database can be tricky. Java is one of the more difficult languages to learn, too. If you don't have a knowledgeable guide, you're better off starting with a simpler technology like PHP.

A widely used version of JSP is Tomcat, available at *http://jakarta.apache.org/*. It's open source (that is, polished by a worldwide population of volunteer programmers), free, and it works with the Apache Web server and many different databases.

Picking a Server Model

With so many choices, you're probably wondering which server model to choose. If you've never built a dynamic Web site before, your best bet is PHP. You can easily set up a fully operational Web server (Apache), application server (PHP), and database server (MySQL) on your desktop computer, and quickly begin building dynamic pages with Dreamweaver.

In fact, since this is the easiest method, this book's tutorials concentrate on building PHP pages. Once you get the hang of Dreamweaver's dynamic Web-building tools, you can always try any of the other server models to build your sites. (However, switching a single site from one server model to another is difficult and not recommended.)

However, when you're building a real-world Web site, the final decision on which server model to use may be out of your hands. You may be working for a company that's already using ColdFusion for its Web site. Or, if you've already got a Web site up and running, but want to add some database-driven content, you have to use what's installed on that server. If your site is currently hosted at a Web hosting company, you should contact the company to find out which operating system, Web server, and databases it uses. If they're Windows-based, odds are that they use IIS, meaning that you can use ASP, or ASP.NET, and either Access or SQL Server databases. On the other hand, if they're a Unix operation, you'll most likely find the Apache Web server, PHP, and MySQL database.

Fortunately, the tools Dreamweaver provides for the different server models are largely the same. Essentially, you start to build dynamic Web pages using the same techniques you've learned in the earlier sections of this book. For the heavy lifting (like retrieving data from a database or password protecting a Web page), you'll turn to Dreamweaver's menu-driven database tools. They'll help you add the

Dynamic Web Development with ASP.NET

The Dynamic Web Development with ASP.NET Course teaches students how to create a web application using Microsoft's ASP.NET technology. ASP.NET is the web component of the .NET Framework and is an evolution of Microsoft's Active Server Pages (ASP) technology .

The course uses Visual Web Developer a free, light-weight program for designing web applications based on HTML and ASPX pages. Visual Web Developer includes a personal web server that can be used to serve web pages to the local computer. Visual Web Developer also includes utilities for designing and connecting to Access and SQL Server databases that be used to create database-driven web applications.

The Dynamic Web Development with ASP.NET Course teaches the basics of web page layout and design using standard HTML elements. Interactive ASP.NET pages are created using web controls provided in the Visual Web Developer Toolbox and Visual Basic.NET code. Formatting web controls using the Properties Window is emphasized. Students are taught how to write code to respond to various web control events such as button clicks and page loads. A brief review of key Visual Basic.NET programming concepts is provided.

The course emphasizes data-base driven ASP.NET applications. Visual Web Developer provides several controls and code wizards that make it easy to connect an ASP.NET

application to a database. The student is taught how to use the utilities provided with Visual Web Developer to design and populate an Access database. Using the MXDataGrid and DataGrid controls and Code Wizards the student creates a number of data-base driven applications. Again the emphasis is on creating a working application, rather than learning the intricacies of SQL statements and ADO.NET connection syntax. The Code Wizards are used to create a web-based database administration tool that can be used to display, edit, add and delete records from a database.

The Dynamic Web Development with ASP.NET Course also emphasizes the use of list controls such as dropdown lists and radiobutton lists. These controls can be automatically populated by binding a field in a database table to the control. Databinding reduces development and maintenance time and decreases errors associated with manually populating these controls.

Advanced topics include using login pages, providing security and data validation. The student is taught how to download and install custom controls that can be added to the Visual Web Developer Toolbox, expanding its functionality. A simple custom control is created and used in an application. The use of Web Services is demonstrated by creating an Amazon.com search page.

Your Task

Purpose

Demonstrate the ability to create a simple ASPX web page.

Instructions

Create a new web form (File>>New File. Select Web Form and check the box, Place code in separate file.) Name it 02_Welcome and add an empty label and a button. A sample follows. The label in this example still has "Label" in the Text property so that you may see it here. You should remove the word "Label" from the Text property on your actual page.

Use the Property Window to empty the Label Text Property, and to place "Click Me" into the Button Text property.

Double-click the button to generate the code section. Use Visual Basic code to place "Welcome to the Maktub website" in the Text property of the label.

Change the button BackColor property.

Test your program.

Challenge

Try placing an ImageButton as well. Copy the C:\WebDeveloper\Images folder to C:\Maktub. Choose a picture from the Images folder. Display a different message when the picture is clicked.

Teacher's Tip

This Task is simply intended to make the student more comfortable using Visual Web Developer.

Your Task

Purpose

Demonstrate the ability to create a web page using HTML controls.

Instructions

Create a web form in the Maktub website that shows information for one band member. This is a static page. In a later task you will create a dynamic page that retrieves data from a database. You will also enhance this page in Task 04.

Choose one of the five members of the band. You may find information about the band at www.maktub.com. Use the information from the band journal pages to create content for this page.

Name the page 03_ArtistBio.aspx.

You may use your design skills to create the layout of the page. You are expected to use a table to organize the presentation of the data. In this task, the fonts and most formatting is not required. That is done in Task 04. Here, you focus on putting data and controls into a table.

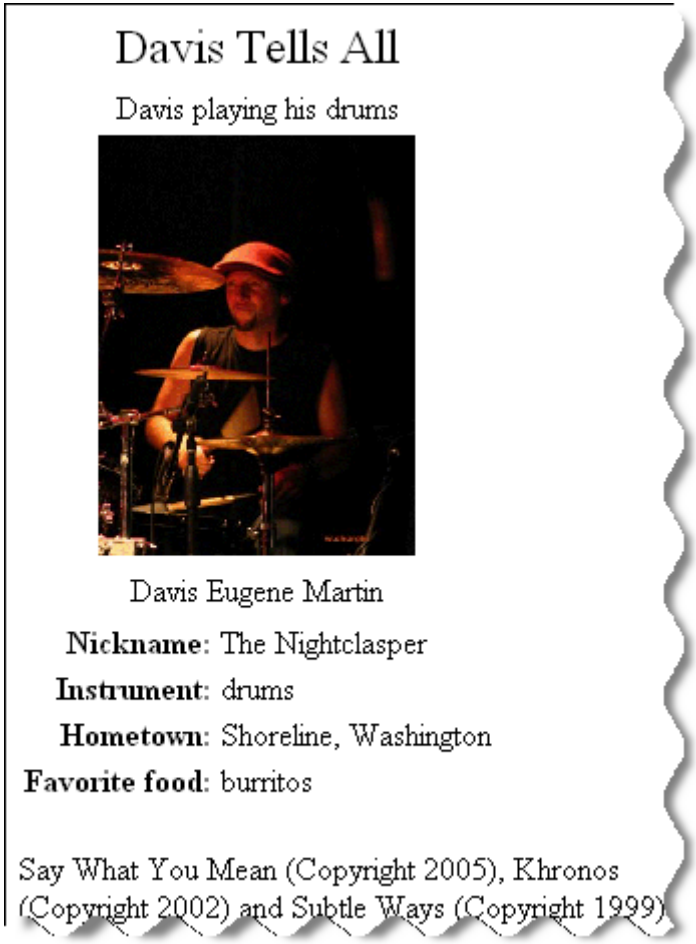
Include these elements on the page:

- Page header with text such as “Davis Snippets” or “Daniel Tells All” or whatever you like. It should be something that will not offend the band.
- Image control with a picture of the artist. Make it an appropriate size by dragging the “size handles.” Artist pictures are found in the Image folder that you have copied to your Maktub website folder. Put a caption over the picture and the artist’s full name under it.
- Show at least five types of data, with a title followed by the appropriate data for the artist you chose. Some possibilities include Name, Email address, hometown, biography, instruments, and nickname. You may find more data in the artist’s journals. Titles should be bold.
- Include the 01_CopyrightNotice.aspx file at the bottom of the page.

Teacher’s Tip

This Task will be totally replaced by a future Task that uses information stored in the database instead of “hard-coded” information.

Here is one possible solution.



Challenge

Include extra data items. Experiment with other HTML elements, such as the Horizontal Line. Try the alignment buttons in the tool bar. Use a width parameter on the table element to center the table on the page.

INTRODUCTION TO DATABASE PROGRAMS - USING ACCESS

1. A **DATABASE** is a collection of information organized into one or more files (tables).

2. A **FILE** (TABLE) is a collection of related information.

3. Files (tables) are organized into **RECORDS**. Each record contains information on one entity in the file. Each piece of information occupies one space called a **FIELD**.

EXAMPLE:

NAME	FORM	STREET ADDRESS	CITY	POSTAL CODE	SIN
Bill Ament	10B	25 Arthur Street	Kitchener	N3B1K6	433043149
Larry Devitt	11C	24-12 Heathcliff Ave.	Waterloo	N1A2K3	234332121
Martha Stauch	12D	69 Appleway Cres.	Kitchener	N4F204	221323432
Richelle Brown	9A	5 South Street	Kitchener	N6G3B2	443546676
Tracy Bering	9D	14 Howland Road	Kitchener	N7B2S1	433034148
Nworb Salguod	11C	25 King Street	Kitchener	N8D2E4	443221870
Nworb Eniale	13B	25 Arthur Street	Kitchener	N8D2E4	876345232

The **file (table)** above contains **7 records**. Each record contains information on one entity, in this case a person. Each **record** has **6 fields**: name,form,street address, city,postal code and social insurance number. The social insurance number could be a link to another file (table) containing more information on each person. These **two files** (tables) could be called a **database.**

EXERCISE:

1. Create a new file (table) called 'friends'
2. Set up the file (table) to store the following information about a number of people: name, telephone number, age, and address.
3. Once you have set up the fields , enter at least six records. Use your friends or classmates to generate the data.
4. Close the file (table) when done.
5. Reload the file (table) back in.

Your Task

Purpose

Demonstrate the ability to create and populate an Access database table.

Instructions

Create one table called Fans.

The table should have these fields with these characteristics:

Name	Data Type	Size	Required	Primary Key	Indexed
ID	Text	20	Yes	Yes	Yes(No Duplicates)
Password	Text	20	No	No	No
Name	Text	50	No	No	No
Birthdate	DateTime	0	No	No	No
Gender	Text	1	Yes	No	No
City	Text	50	No	No	No
State	Text	2	No	No	No
Email	Text	50	No	No	No
FavoriteSong	Text	20	No	No	No
StreetTeam	Text	1	Yes	No	No
Comments	Text	250	No	No	No

Once the table is completely designed, put data in the table. Add 3-4 fans to the table. Have fun with the data while using good taste.

Show the resulting table to your instructor.

ACCESS- STUDENT AVERAGES EXERCISE:

Listed below is information and various marks for a number of students:

Name	Grade	English	Math	French	History	Student Average
Carson James	9	76	72	73	65	
Sleman Douglas	9	67	58	94	62	
Turner Ethel	10	34	43	55	67	
Brown Elizabeth	10	67	74	92	66	
Cartwright Mary	10	67	68	93	70	
Harris Ginny	10	78	67	32	55	
Alan Norman	11	75	89	78	56	
Stenn Charlotte	11	53	59	68	69	
Wilson Marion	11	78	88	89	82	
Lobb William	12	68	59	86	83	
Jackson Mark	12	45	86	56	55	
Course Averages						

Step 1

Create a new database called **averages**. Then import the table 'stdav' from stdav.mdb located on the v: drive.

Step 2

Create the above form based on a query calculation that determines the student averages. The form should also include a grade status (eg. A,B,C,D,F)

F student average <50
D 50<=student averge <60
C 60<=student average<70
B 70<=student average<80
A 80<=student average<=100

Step 3

Produce the following reports

a) The students' results sorted by student average in descending order
b) Just the results of students with averages greater 80 or less than 60

COMPUTER SCIENCE DAYS

PROBLEM

Your local fast-food restaurant wants to keep track of the students working for them and some information about each student. They also need separate records of which students are working each day and how many hours they worked. Finally, they need to issue pay cheques at the end of the week.

Your job is to enter and maintain the data in their Watfile system to enable fast and easy access to all information.

The first thing you need to do is to decide what information is necessary for each student. The following gives an example of data you may wish to record when the student starts work.

NAME	TELEPHONE	POSITION	RATE
Smith, John	745-7998	Busboy	$4.85
Brown, Sue	768-9034	Busboy	$4.85
Jones, Harry	455-7654	Cashier	$5.65
Clark, Joan	344-8734	Sweeper	$5.00

To keep track of which students are working on which days, time cards such as the following may help.

DATE	NAME	HOURS
11/04/88	Smith, John	10
11/04/88	Brown, Sue	5
11/04/88	Clark, Joan	8
12/04/88	Brown, Sue	7
12/04/88	Clark, Joan	5
12/04/88	Jones, Harry	10
13/04/88	Jones, Harry	5

You now have all the information you need to issue pay cheques. Combining information from the above two files, you can prepare cheques as follows:

NAME	RATE	HOURS	CHEQUE
Smith, John	$4.85	10	$48.50
Brown, Sue	$4.85	12	$58.20
Jones, Harry	$5.65	15	$84.75
Clark, Joan	$5.00	13	$65.00

10 DB-Driven Band Bio Page *Using the GridView Control*

Our first database-driven site feature is a new Band Bio page. Using a database and the GridView control, we'll display a lot more information about each band member. The GridView control makes connecting to a database as easy as drag and drop. Then we'll add an Artist of the Month feature, which lets users select their favorite band member. We'll also add some filtering capabilities so users can limit the information displayed on the web page.

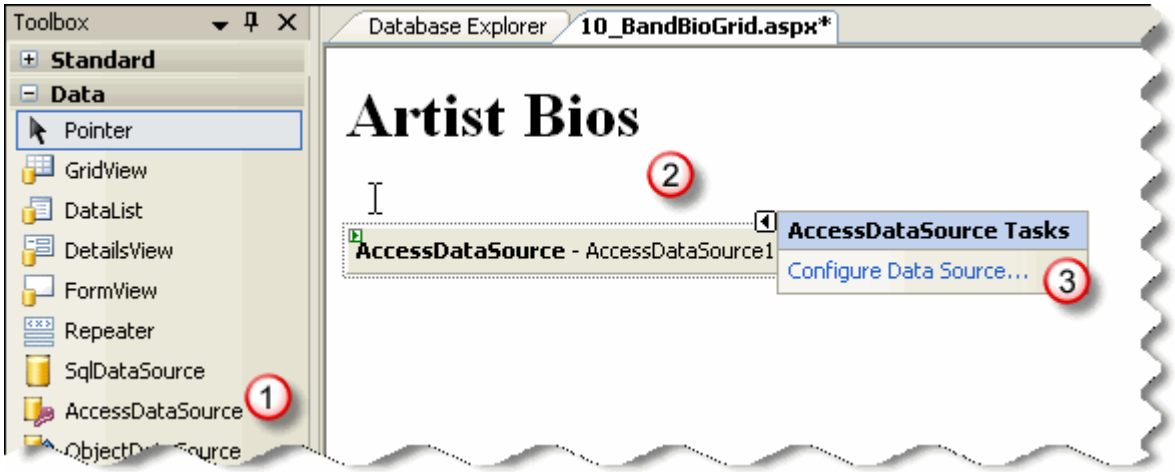
Create a Band Bio Page

Open Visual Web Developer and add a new ASPX page called 10_BandBioGrid.aspx. On the Design tab, type Artist Bios at the top of the page. Change the paragraph style to H1. Press the Enter key to create a new line.

Your page looks like this:

Your Database Connection

After you create your ASPX page, you need to add a Data Source and make a database connection. If you haven't copied the BandDatabase.mdb file to your Maktub website, do so now.

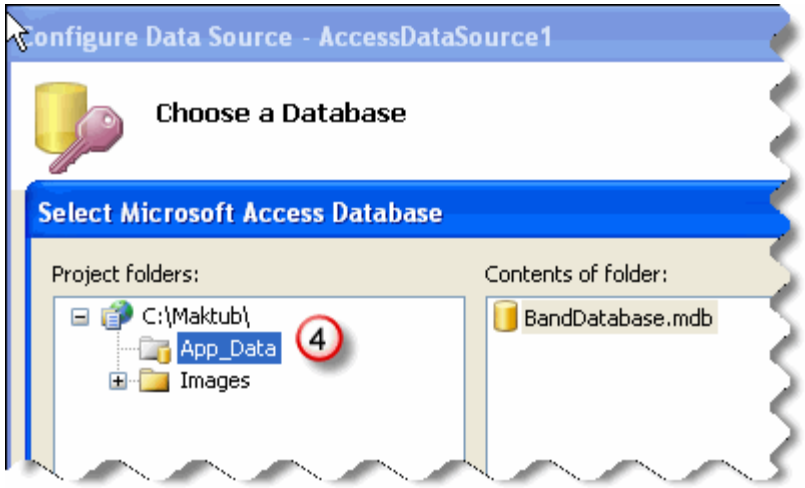


Teacher's Tip

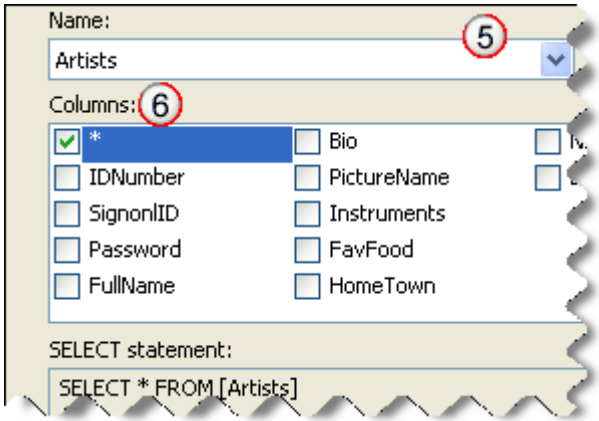
The course is using the Artist table to show how to add a table. The Artist table already exists in both the student and teacher versions of the band database.
The band members are called artists.

It should be in your C:\Maktub\App_Data folder.

1. Select the Data tab on your Toolbox and find the AccessDataSource control.
2. Drag the AccessDataSource control to web page.
3. Select Configure Data Source and the Choose a Database window will open. Click Browse.
4. Double click on the App_Data folder. The BandDatabase.mdb file is highlighted. Click OK and then Next.

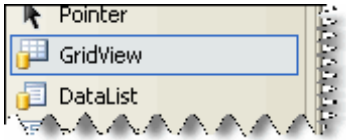


5. In the Configure the Select Statement Window, select the Artist table from the dropdown list in the Name section.
6. In the Columns section, check the asterisk (*) box to select all columns. Click Next and Finish.

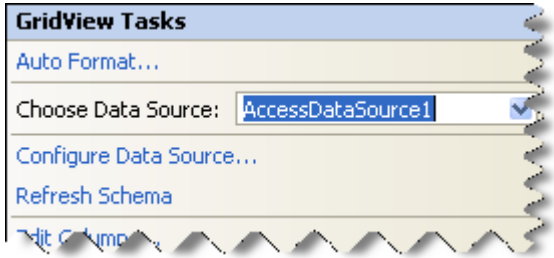


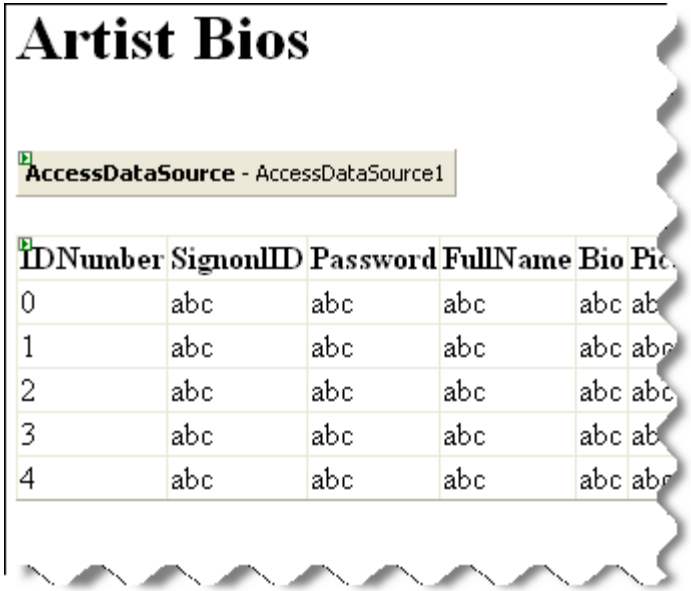
Add the Artist Table

Now that we have a database connection established, we are ready to add data to our ASPX page. From the Data tab in your Toolbox, drag a GridView control onto our ASPX page.



The GridView Tasks Window will be displayed. Select the AccessDataSource1 from the dropdown list.





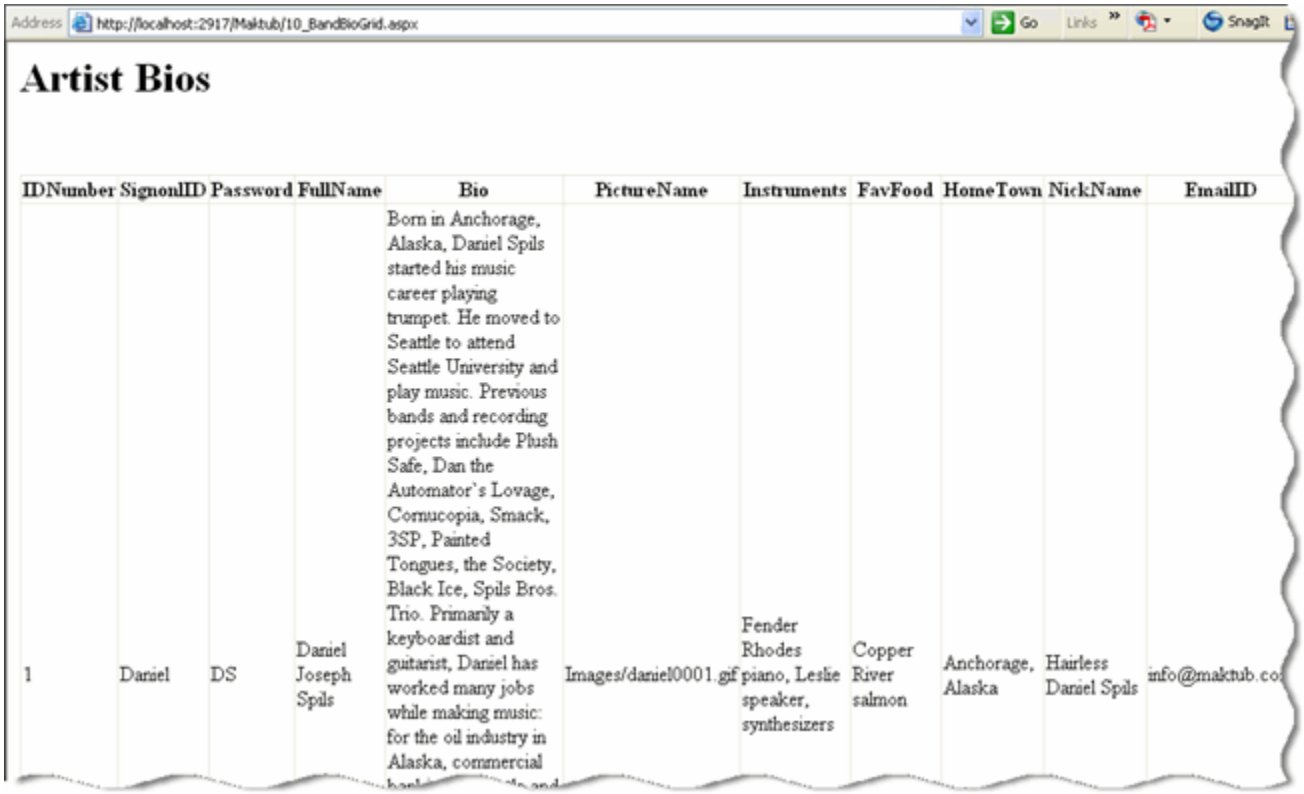
Teacher's Tip

You may point out to the students that they can also drag a table onto the form, thus automatically creating the GridView and AccessDataSource controls.

Notice that the IDNumber field of each record is filled with numbers (representing an integer/key field) and the remaining fields of each record are filled with "abc" (representing text).

Viewing the Band Bio Page

Believe it or not, the hard work is done. You're ready to view your Band Bio page. Test your web application. The web server starts and then your web page is displayed in the browser:



Teacher's Tip

The bios are rather lengthy and may cause the students to think their controls are not working. They just need to scroll down.

Wow! Your ASPX page displays a table containing all the data in the Artist table of the BandDatabase database. Scroll up and down and you see rows corresponding to all the records in the table -- one for each band member. Scroll left and right and you see columns of data corresponding to all the fields in the table. Notice that the records are ordered by IDNumber.

Notice also that the AccessDataSource control is not shown. Only the GridView control and our page title are displayed.

This is really great! Let's learn about some of the properties of the AccessDataSource and GridView controls that make up this ASPX page.

Your Task

Purpose

Demonstrate the use of a GridView control.

Instructions
Use the 11_Tours.aspx template page. Give the page a header of Tours.

Drag a GridView control onto the page. Connect the data source to the Tours table in the BandDatabase.

The initial result, if you save and test the page, looks something like this:

Tours

TourNumber	Date	City	State	ZipCode	VenueName	TicketLink	Cost	Time	Notes
401	3/26/2004 12:00:00 AM	Seattle	WA	98101	Nuemoe's				
402	5/6/2004 12:00:00 AM	San Francisco	CA	94101	The Independent	www.ticketweb.com	$12	11:00pm	
402	5/7/2004 12:00:00 AM	Santa Monica	CA	90401	Temple Bar	www.ticketweb.com	$12	11:00pm	
402	5/8/2004 12:00:00 AM	San Diego	CA	92101	Belly Up	www.ticketweb.com	$12	10:00pm	
402	5/9/2004 12:00:00 AM	Joshua Tree	CA	92252	Joshua Tree Musical Festival	www.ticketweb.com	$23-$75	7:00pm	All ages

That's a start. You can make it look a lot nicer.

Choose "Enable Paging". Make appropriate choices for the number of items per page. Make the page numbers numeric. Set the PagerStyle font size to medium to make the numbers easier to read.

For columns, choose only ShowDate, City, State, ZipCode, and TourNumber, in that order. The other fields will be used in a future Task. Set appropriate column titles.

Format the date as {0:MM/dd/yy}.

A white border two pixels wide would look nice.

Adjust the column widths to more appropriate sizes. The percent option is usually the easiest to use.

This page will be improved in a future Task by adding selection (filtering).

Challenges

Center the column data.

Use AutoFormat to change the appearance of the table, provided your choice is an improvement over the default.

Add the time column to the display.