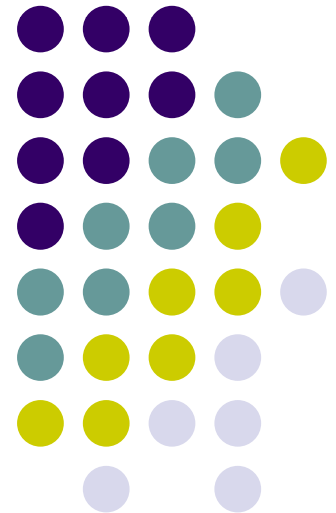
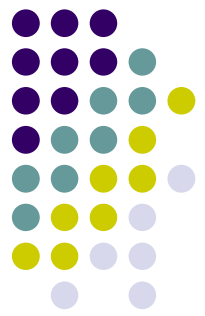


CSI2441: Applications Development

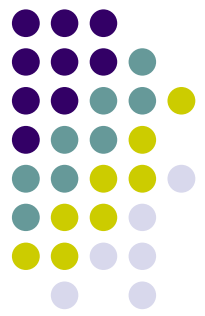
Lecture 6 *Visual Programming Environments*





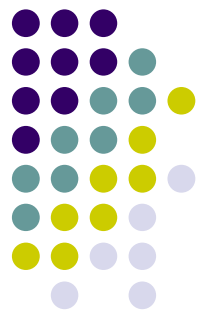
Objectives

- Examine visual programming environments
- Look at the Rapid Application Development (RAD) features of such environments
- Look at pre-built components
- Look ‘under the bonnet’ at coding around visual components
- Interface and navigation automation
- Benefits and drawbacks of visual programming environments



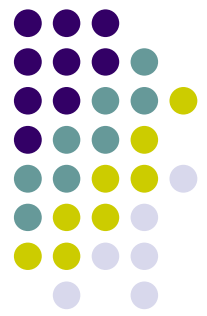
Introduction to Visual Programming Environments

- Integrated Development Environments (IDE's) can come in the form of text-based or visual programming systems (or both)
- Visual programming environments are designed to speed up traditional programming processes
- Visual environments contain pre-built components which encapsulate programming tasks
 - Database connections
 - Data retrieval and display
 - Interface and navigation
 - Login and authentication
- The whole idea of visual systems is to allow drag-and-drop Rapid Application Development (RAD)

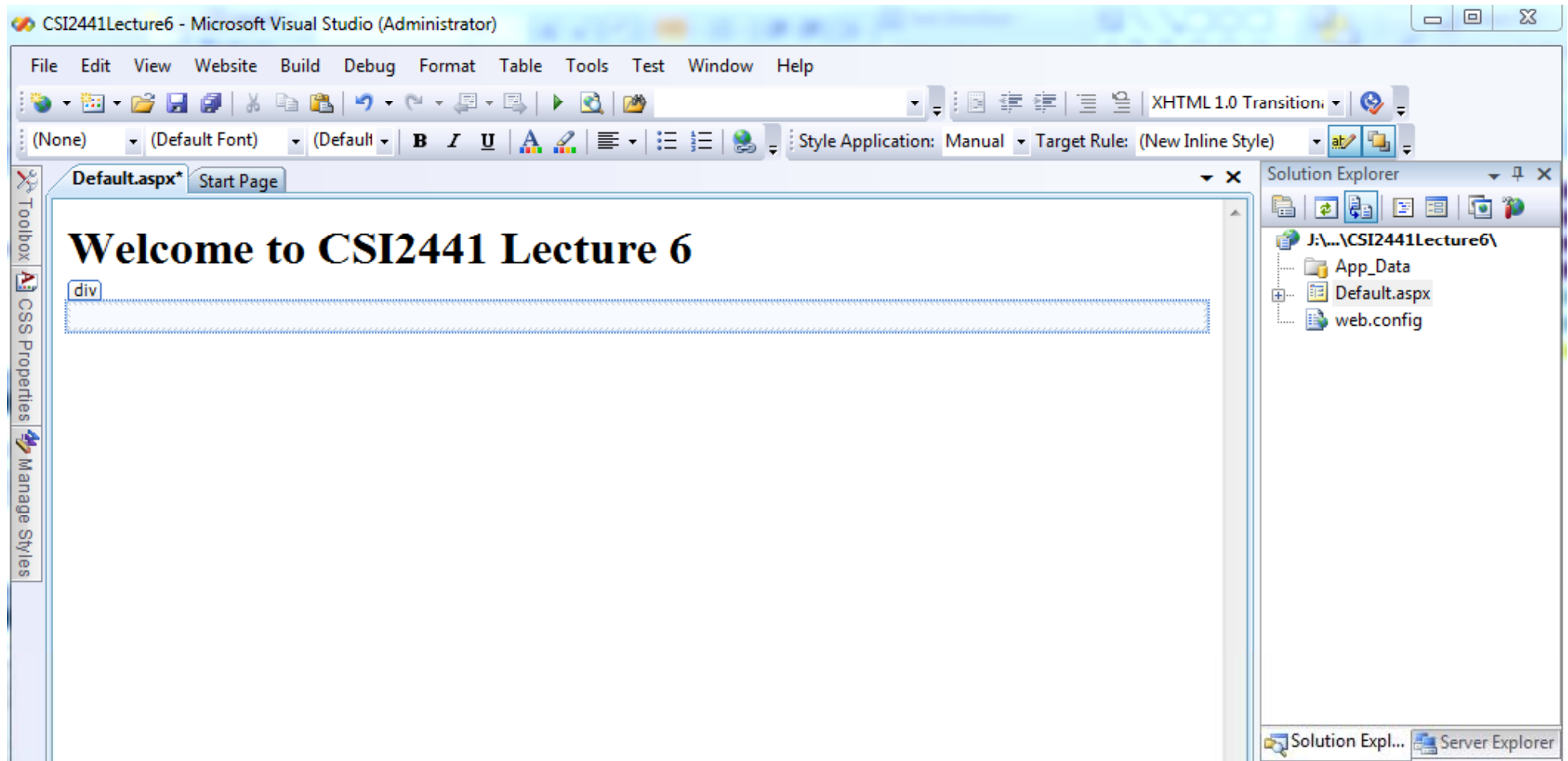


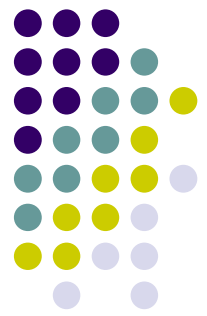
Visual Programming Environments

- There are a number of visual programming environments on the market, ranging from freeware to expensive enterprise level
- Some of the current include
 - Microsoft's Visual Studio
 - Lily
 - BlueJ (Java)
 - Adobe Dreamweaver CS4, Authorware and ColdFusion
 - Oracle Forms
- There are dozens if not hundreds of other environments which are designed for specialised tasks (such as games, audio or multimedia programming)
- The examples to follow are largely taken from Visual Studio 2008 in web programming mode

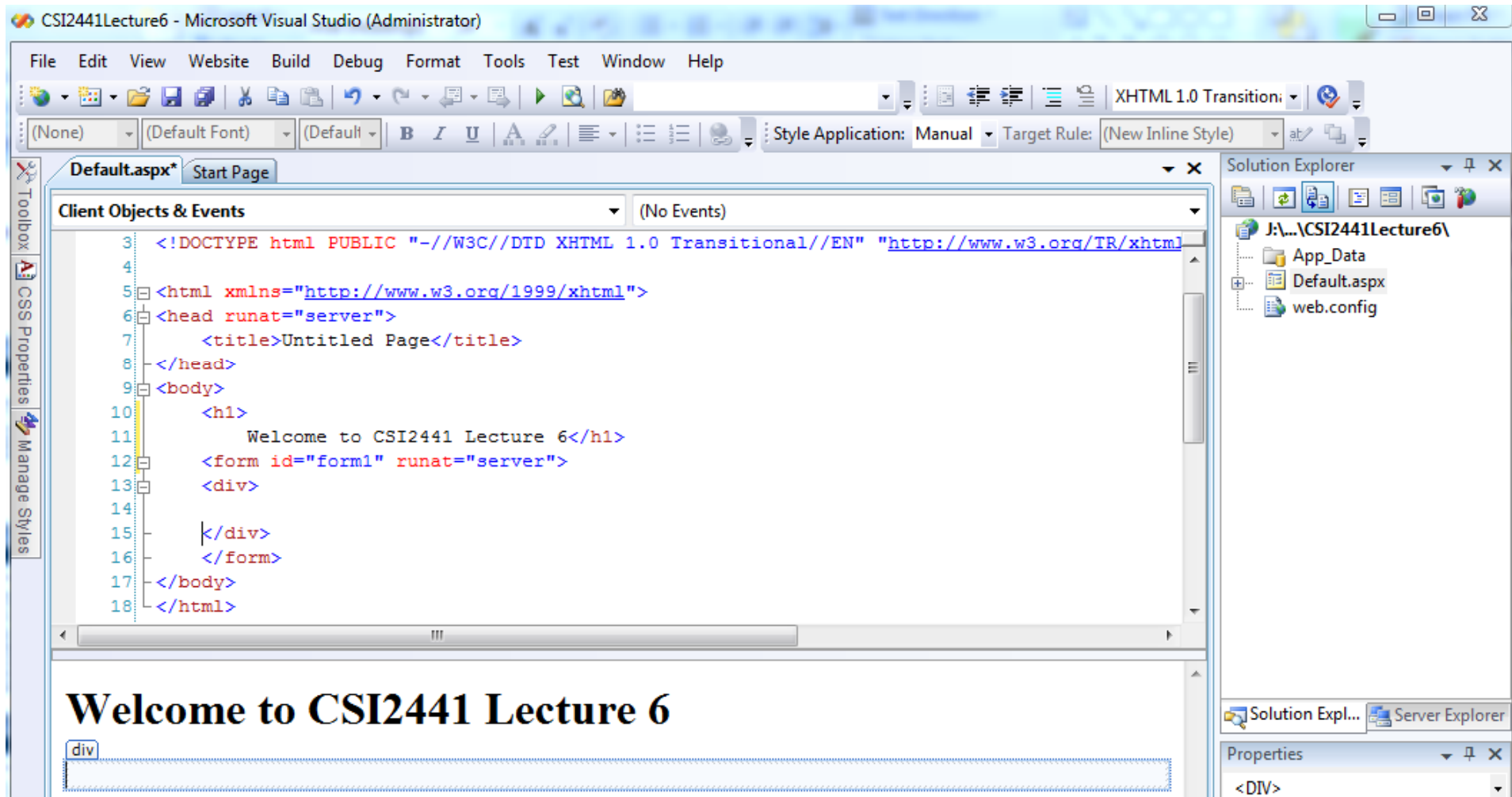


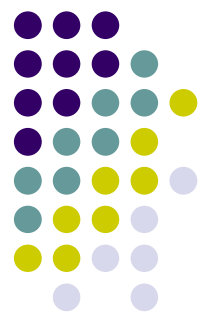
The Visual Interface





Split Visual Interface





The screenshot shows the Microsoft Visual Studio (Administrator) interface. The main window displays the file **Default.aspx** in the **Start Page** view. The **Client Objects & Events** pane shows **(No Events)**. The **Style Application** is set to **Manual** and the **Target Rule** is **(New Inline Style)**. The **Solution Explorer** on the right shows the project structure:

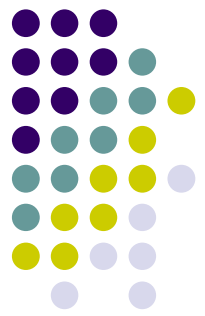
- J:\...\\CSI2441Lecture6\
 - App_Data
 - Default.aspx
 - web.config

The code in **Default.aspx** is as follows:

```

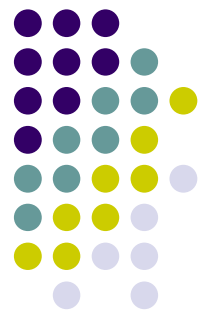
1 <%@ Page Language="VB" AutoEventWireup="false" CodeFile="Default.aspx.vb" Inherits="_Default" %>
2
3 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1" %>
4
5 <html xmlns="http://www.w3.org/1999/xhtml">
6 <head runat="server">
7   <title>Untitled Page</title>
8 </head>
9 <body>
10   <h1>
11     Welcome to CSI2441 Lecture 6</h1>
12   <form id="form1" runat="server">
13     <div>
14
15     </div>
16   </form>
17 </body>
18 </html>

```



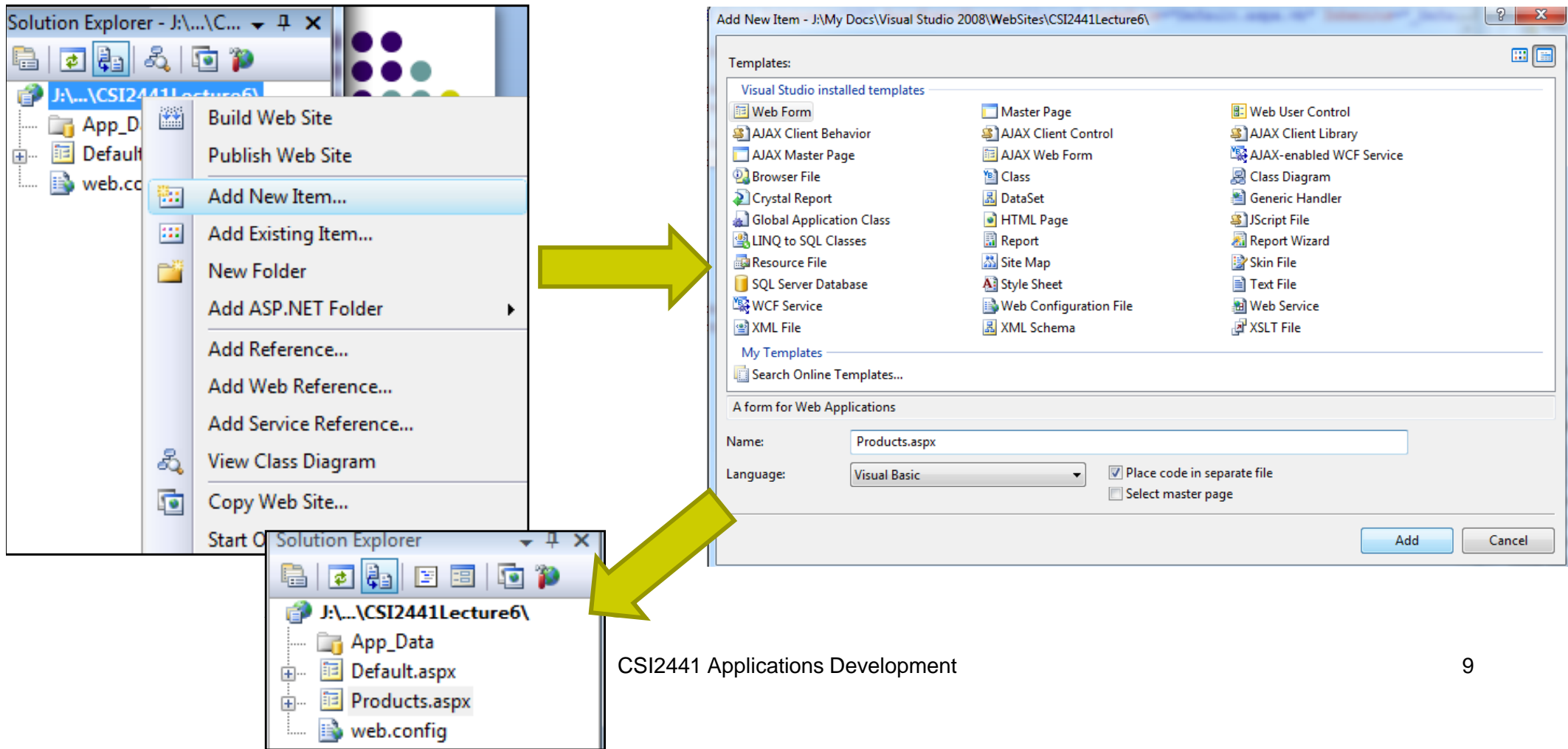
Visual versus Text Based Coding

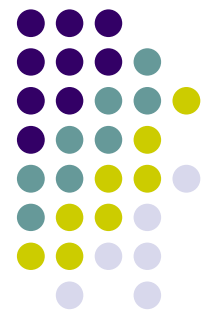
- As the previous slides show, the developer can choose what mode they wish to use to develop
- In reality, if you are using a visual IDE like VS 2008 then you are probably doing so to make use of the visual drag and drop tools
 - Using a tool like this in code-only mode is like swatting a fly with a battleship (ie overkill)
 - There are more efficient text only tools to do such a job
- The benefits of the visual environment include the pre-built controls, the auto-complete features and the limited on the fly error checking (such as undefined variables being highlighted)
- In almost any visual IDE, you will inevitably need to switch into source code mode to 'tweak' features that cannot be done visually



Visual Site Management

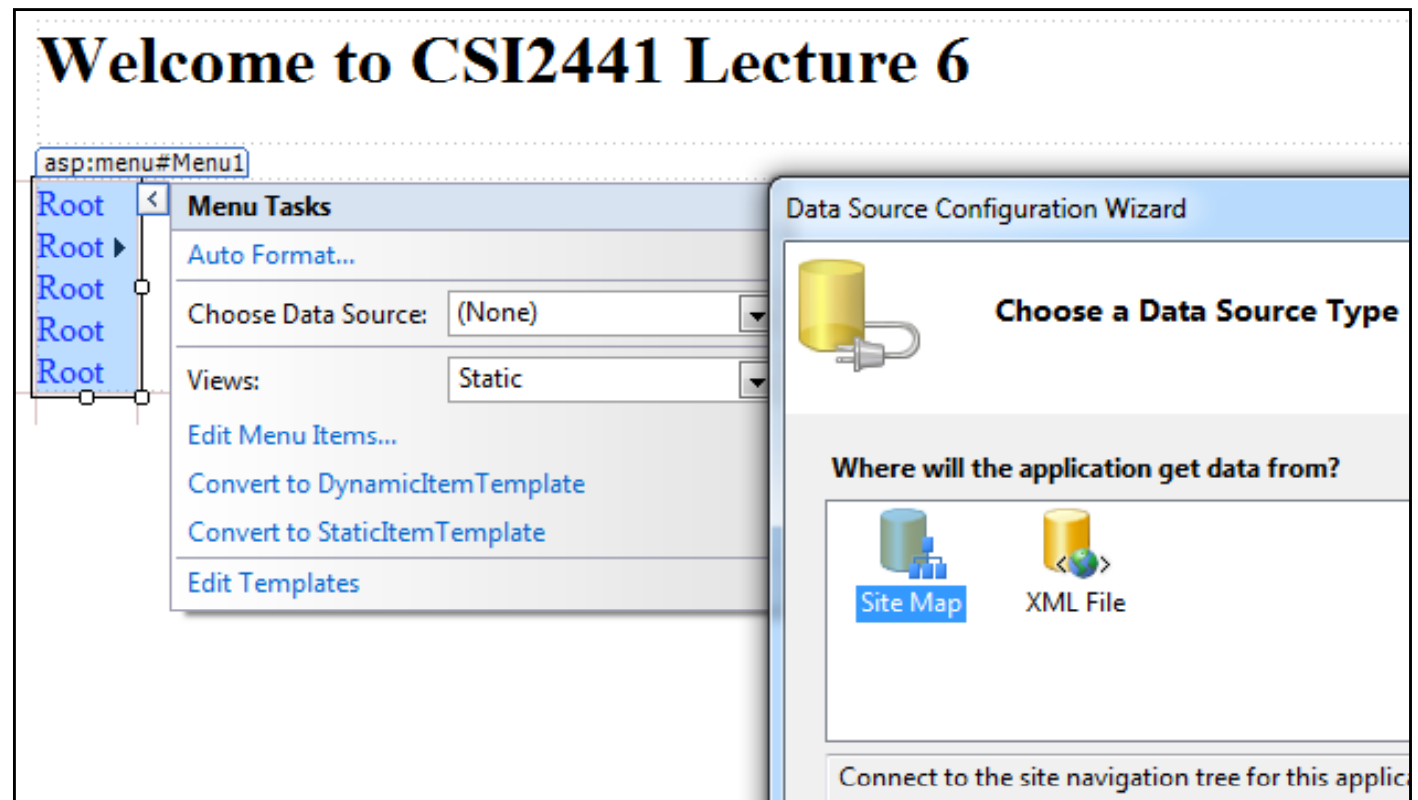
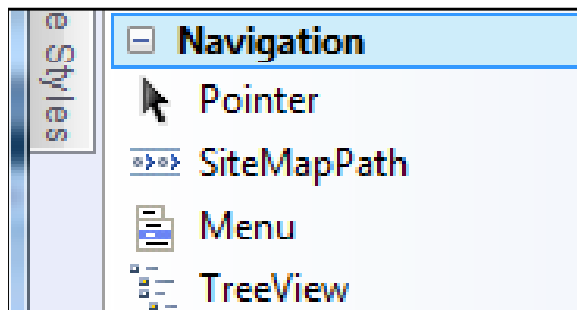
- When working in a visual IDE, adding new pages to a site and managing site navigation is typically a point and click affair

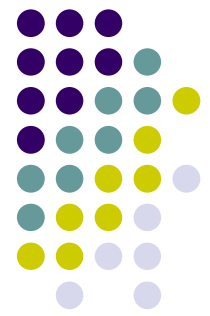




Navigation

- Rather than hand coding navigation for each page (and links to other pages) navigation controls can be used to automatically manage navigation





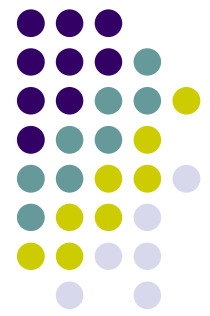
Visual Site Management cont...

- In this example the sitemap is defined in a simple xml file
- In each page that requires navigation this menu control can be used to present the same navigation
- For complex site structures pre-built controls can be placed on each page which indicate to the user which page they are on within the site structure
- To do this in written code is both complex and time consuming
- Using pre-built controls there is no code required and almost no code associated with the control doing all the work

Welcome to CSI2441 Lecture 6

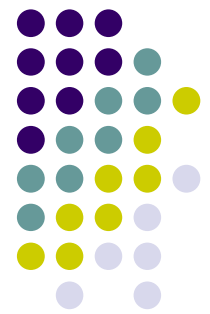
Main Menu ▶ Products
About Us in Menu
Contact Us

```
<asp:Menu ID="Menu1" runat="server" DataSourceID="SiteMapDataSource1">  
</asp:Menu>  
<asp:SiteMapDataSource ID="SiteMapDataSource1" runat="server" />
```



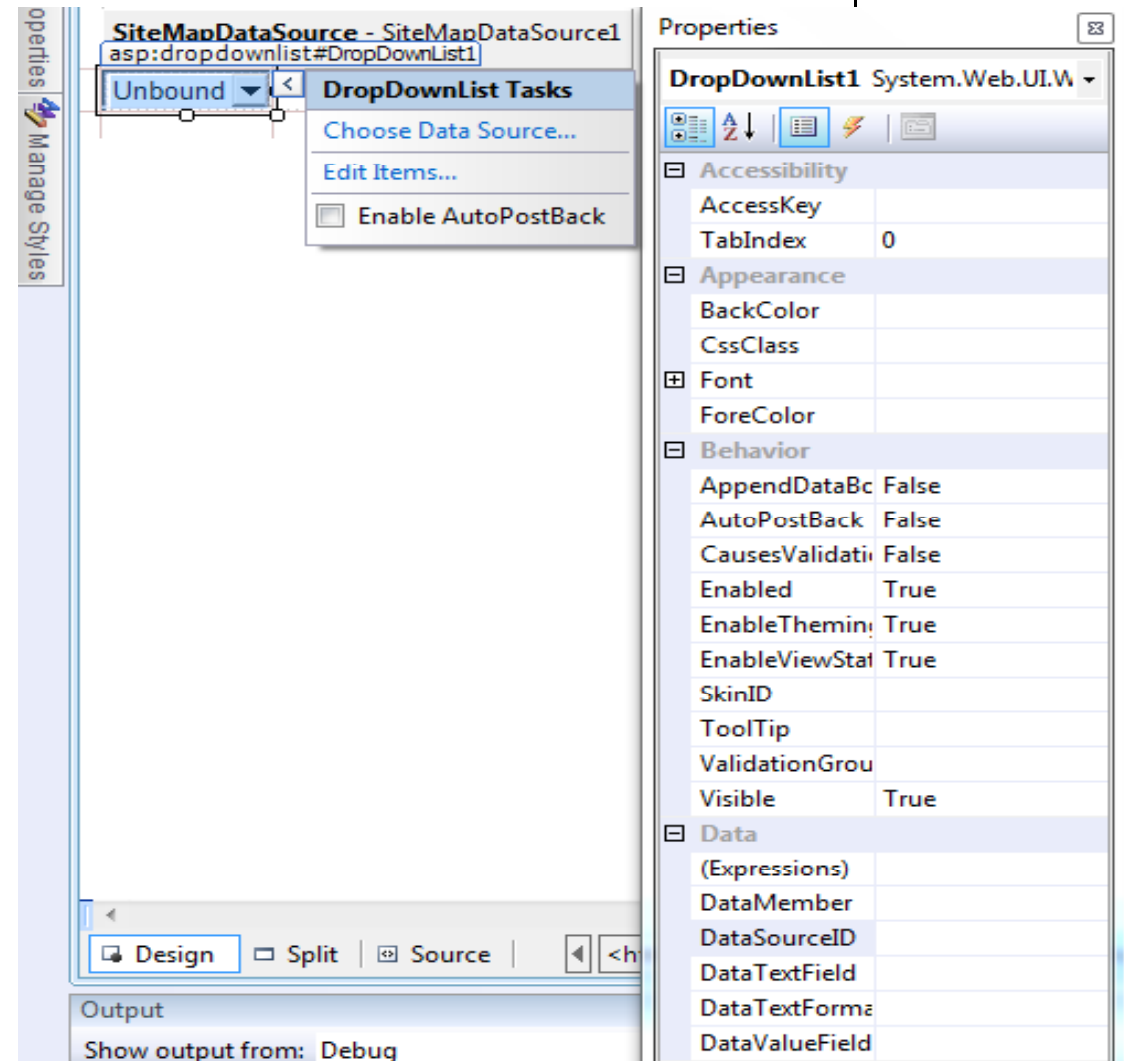
Controls and Components

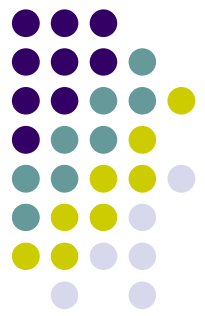
- Whilst the terminology varies across platforms, normally we refer to a control or a component
- Controls essentially are an encapsulation of a process into a pre-built representation of that process
- Controls can receive input and produce output, but the underlying code that makes them work is hidden (and in most cases copyrighted intellectual property)
- Controls can be customised to a certain degree by tweaking their settings (in terms of look, feel and functionality)
- In the end, most controls try to be both specific in nature (such as a database connection control) whilst also being a general as possible (so as to be able to be used in a large variety of applications)



Setting Control Properties

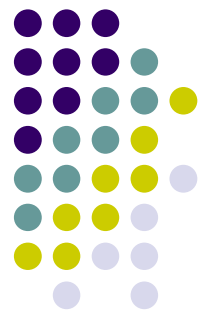
- The image to the right shows just some of the properties (or settings) for a drop-down list control
- These are categorised in terms of how it looks, how it is accessed, what it is called and how it interacts with data
- These properties basically represent the public programmatic interface between the control and the developer





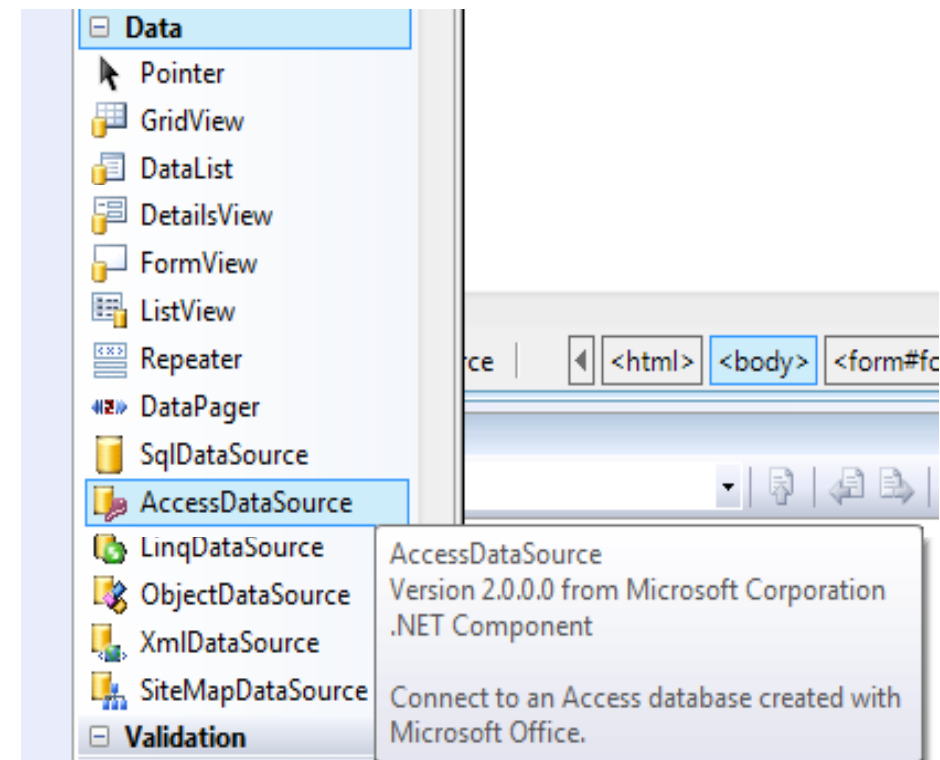
Data Driven Controls

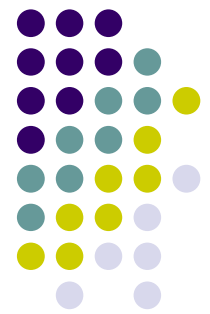
- In terms of both desktop and web based development, interaction between users and databases represent the core functionality of most applications
- Pulling data out of a database, presenting it to users, and then allowing users to interact with that data (such as adding, editing, updating and deleting) can be very time consuming
- The real power of visual development environments is the provision of pre-built controls to assist in these processes
- As the following slides demonstrate, drag and drop visual programming can be extremely quick and easy compared to hand coded solutions



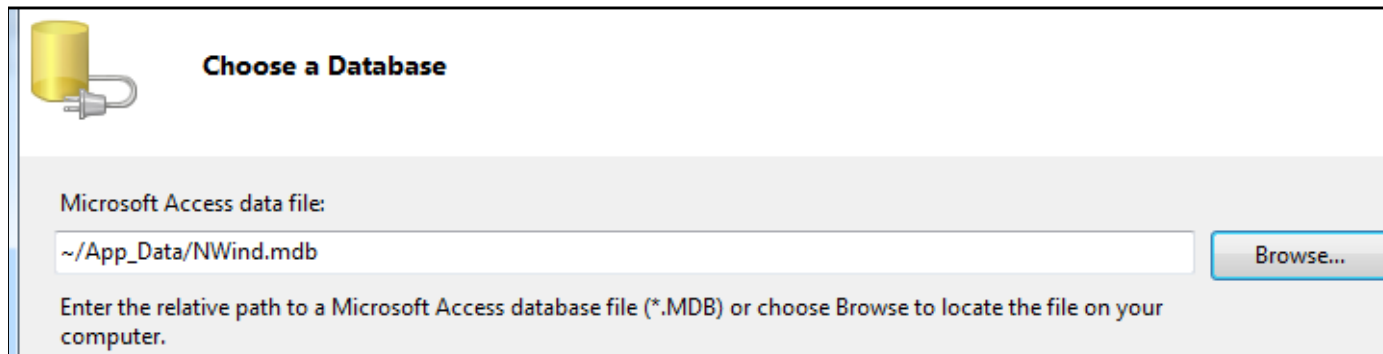
Data Controls

- Using VS 2008 as an example, we can see a number of Controls which perform common tasks, such as the Form View and the DataList
- In this example we will take some product data from the Nwind database used in Workshops 4 and 5
- Before performing any actions with the data, we must link to the database using the AccessDataSource Control





Data Controls cont....

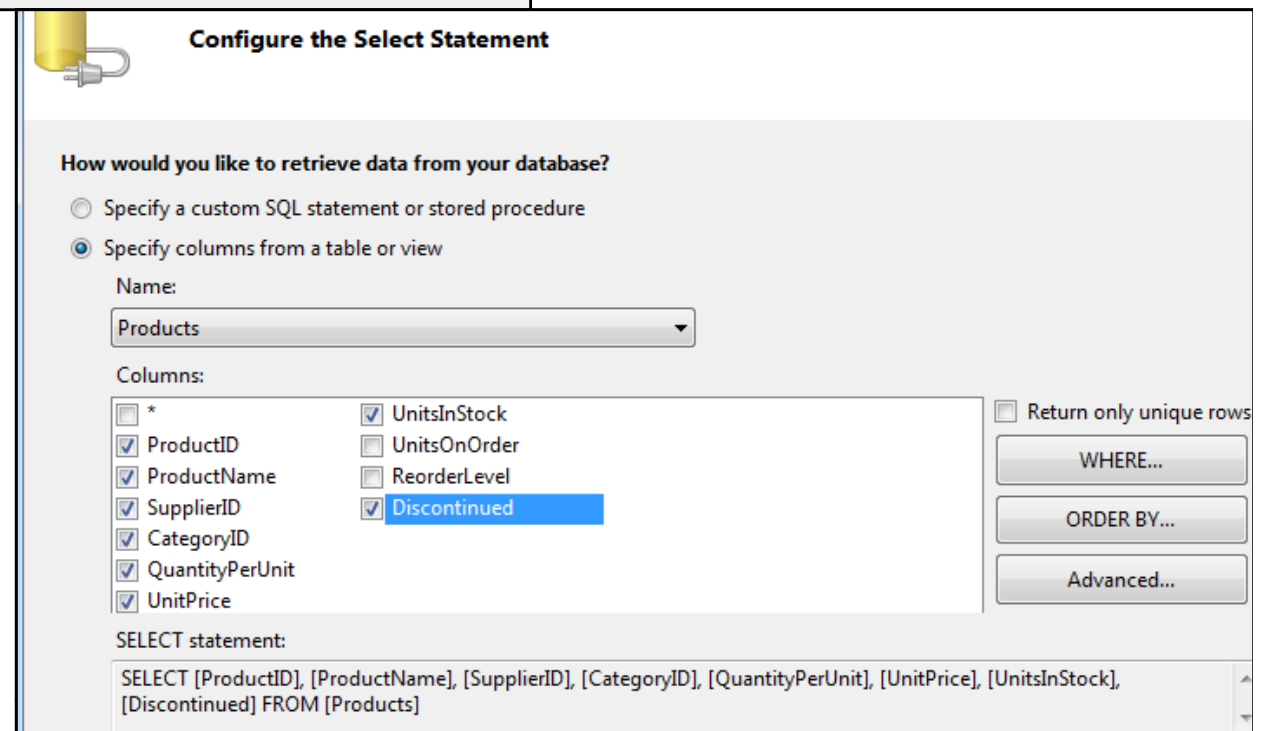


Choose a Database

Microsoft Access data file:

Enter the relative path to a Microsoft Access database file (*.MDB) or choose Browse to locate the file on your computer.

- Here the Control asks for the location of the DB file
- Then we can choose which parts of the DB we wish to read



Configure the Select Statement

How would you like to retrieve data from your database?

☐ Specify a custom SQL statement or stored procedure

☒ Specify columns from a table or view

Name:

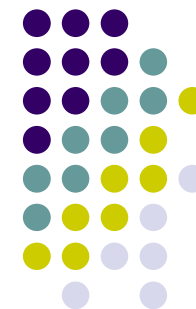
Columns:

<input type="checkbox"/> *	<input checked="" type="checkbox"/> UnitsInStock
<input checked="" type="checkbox"/> ProductID	<input type="checkbox"/> UnitsOnOrder
<input checked="" type="checkbox"/> ProductName	<input type="checkbox"/> ReorderLevel
<input checked="" type="checkbox"/> SupplierID	<input checked="" type="checkbox"/> Discontinued
<input checked="" type="checkbox"/> CategoryID	
<input checked="" type="checkbox"/> QuantityPerUnit	
<input checked="" type="checkbox"/> UnitPrice	

☐ Return only unique rows

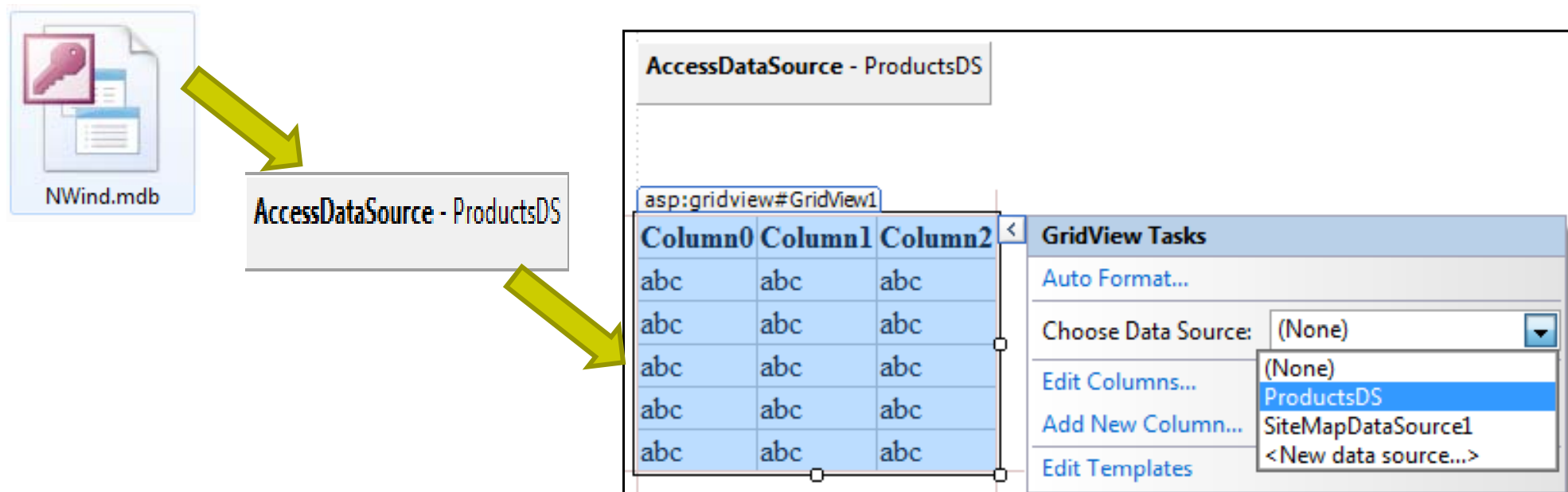
SELECT statement:

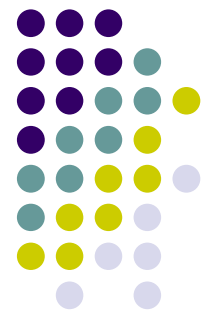
`SELECT [ProductID], [ProductName], [SupplierID], [CategoryID], [QuantityPerUnit], [UnitPrice], [UnitsInStock], [Discontinued] FROM [Products]`



Data Controls cont....

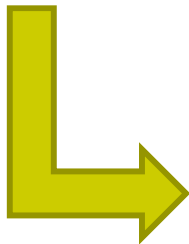
- Once the datasource is configured and named, we can then add another control to display and manipulate the contents of the datasource
- In this case we hook up a GridView data display control to our Products datasource





Data Controls cont...

ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInStock	Discontinued
0	abc	0	0	abc	0	0	<input type="checkbox"/>
1	abc	1	1	abc	0.1	1	<input checked="" type="checkbox"/>
2	abc	2	2	abc	0.2	2	<input type="checkbox"/>
3	abc	3	3	abc	0.3	3	<input checked="" type="checkbox"/>
4	abc	4	4	abc	0.4	4	<input type="checkbox"/>

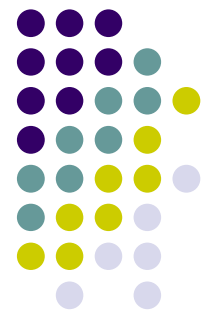


The pre-rendered
GridView now displays all
records in the products
table onto screen without
a line of code being written

Products

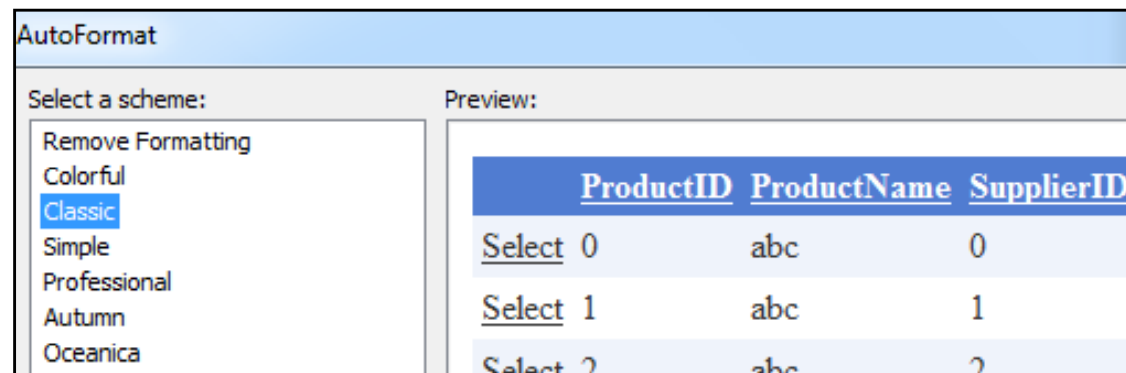
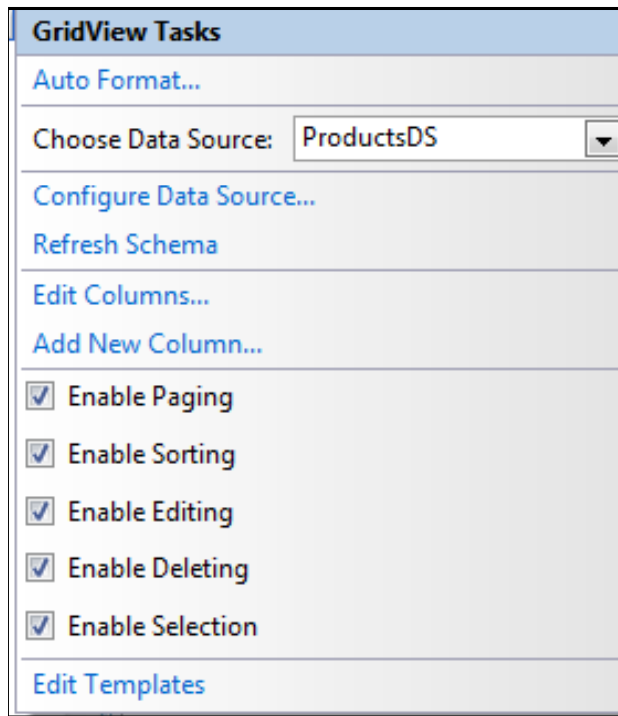
[Main Menu](#) ▶

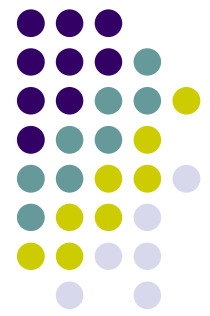
ProductID	ProductName	SupplierID	CategoryID	QuantityPerUnit	UnitPrice	UnitsInStock	Discontinued
1	Chai	1	1	10 boxes x 20 bags	18	39	<input type="checkbox"/>
2	Chang	1	1	24 - 12 oz bottles	19	17	<input type="checkbox"/>
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10	13	<input type="checkbox"/>
4	Chef Anton's Cajun Seasoning	2	2	48 - 6 oz jars	22	53	<input type="checkbox"/>
5	Chef Anton's Gumbo Mix	2	2	36 boxes	21.35	0	<input checked="" type="checkbox"/>
6	Grandma's Boysenberry Spread	3	2	12 - 8 oz jars	25	120	<input type="checkbox"/>
7	Uncle Bob's Organic Dried Pears	3	7	12 - 1 lb pkgs.	30	15	<input type="checkbox"/>
8	Northwoods Cranberry Sauce	3	2	12 - 12 oz jars	40	6	<input type="checkbox"/>
9	Mishi Kobe Niku	4	6	18 - 500 g pkgs.	97	29	<input checked="" type="checkbox"/>
10	Ikura	4	8	12 - 200 ml jars	31	31	<input type="checkbox"/>




Data Controls cont...

- With some simple clicks we can now heavily extend the functionality of the GridView Control
- The left image enables data sorting and management functions (adding, editing and deleting)
- The right image enables pre-defined presentation styles





Data Controls cont...

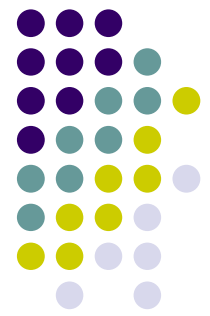
- And this is the result
- For each record as well as viewing it we can edit it or delete it
- By clicking Select we can also send the ProductID out of this Control into another one, such as a  DetailsView Control

Products

[Main Menu](#) ▶

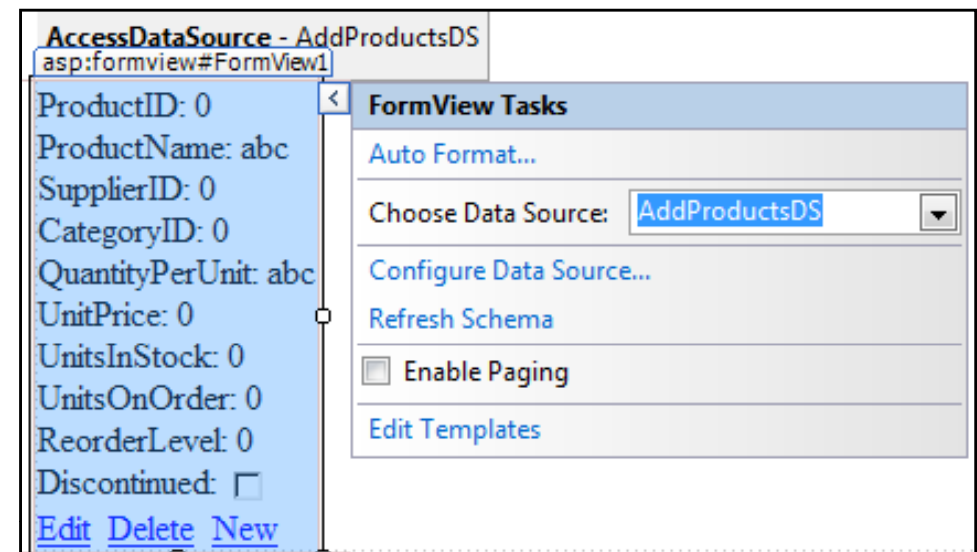
	<u>ProductID</u>	<u>ProductName</u>	<u>SupplierID</u>	<u>CategoryID</u>	<u>QuantityPerUnit</u>	<u>UnitPrice</u>	<u>UnitsInStock</u>	<u>Discontinued</u>
Edit Delete Select	1	Chai	1	1	10 boxes x 20 bags	18	39	<input type="checkbox"/>
Edit Delete Select	2	Chang	1	1	24 - 12 oz bottles	19	17	<input type="checkbox"/>

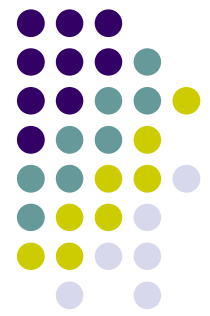
- As you may recall from Workshops 4 & 5, to do a more basic version of this functionality in ASP by hand required ~ 200 lines of code and approx 2 hours of work – as opposed to approx 2 minutes of work and no hand written code



Form Input Controls

- One of the benefits of data driven visual interfaces is that the structure of a database can be used to define the form that inputs data (as discussed in Lecture 4)
- In this case a FormView Control has been linked to an AccessDataSource control which is connected again to the Products table in the database
- As the image shows, the resulting form has created itself as a representation of the database table it is linked to
- By default it also provides options to edit and delete a record





Form Input Controls cont...

- By default this Control allows us to view, then edit or add a record
- Obviously, the default form is very crude, however the developer can go in and alter the layout, colours, spacing, labels, buttons, almost everything
- However, the most time consuming elements are done already

ProductID: 1
ProductName: Chai
SupplierID: 1
CategoryID: 1
QuantityPerUnit: 10 boxes x 20 bags

UnitPrice: 18
UnitsInStock: 39
UnitsOnOrder: 0
ReorderLevel: 10
Discontinued: ☐

[Edit](#) [Delete](#) [New](#)

ProductID: 1
ProductName: Chai
SupplierID: 1
CategoryID: 1
QuantityPerUnit: 10 boxes x 20 bags

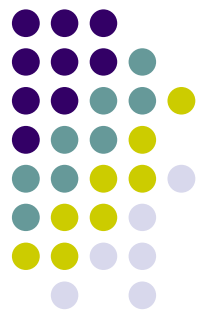
UnitPrice: 18
UnitsInStock: 39
UnitsOnOrder: 0
ReorderLevel: 10
Discontinued: ☐

[Update](#) [Cancel](#)

ProductName:
SupplierID:
CategoryID:
QuantityPerUnit:

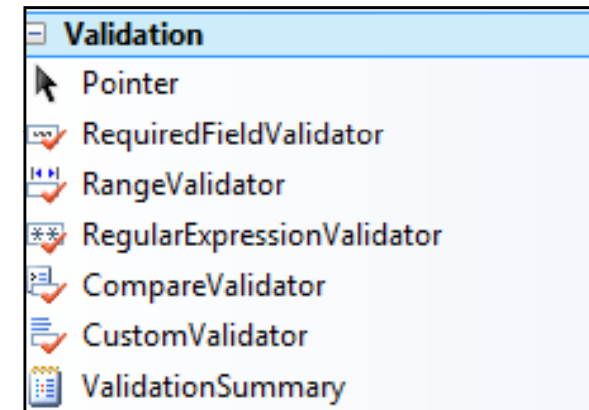
UnitPrice:
UnitsInStock:
UnitsOnOrder:
ReorderLevel:
Discontinued: ☐

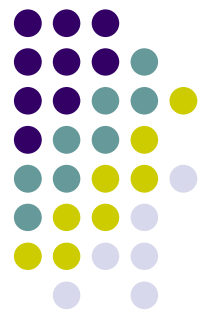
[Insert](#) [Cancel](#)



Validation Controls

- Input validation is another time consuming coding task that can be made easier
- These are the built-in validation Controls found in VS 2008
- These can be added to a form and then set up for the required validation rules
- For example, using the form on the previous slide as an example, we can ensure Product Name is not left blank





Validation Controls cont...

- We drag the RequiredFieldValidator next to the form field we wish to validate
- We set the properties for the Control, including the message, the field to validate and whether we want the field with the error to end up with the cursor in it (known as the focus)

AccessDataSource - AddProductsDS
asp:formview#FormView1

FormView1 - InsertItemTemplate

InsertItemTemplate

ProductName: **asp:requiredfield...#RequiredField...**
Product Name cannot be left blank

SupplierID:

CategoryID:

QuantityPerUnit:

UnitPrice:

UnitsInStock:

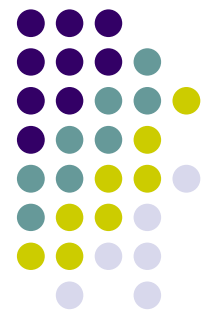
UnitsOnOrder:

ReorderLevel:

Discontinued: ☐ [DiscontinuedCheckBox]

[Insert](#) [Cancel](#)

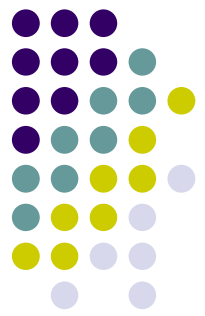
Display	Dynamic
ErrorMessage	Product Name cannot be left blank
Font	
ForeColor	Red
Text	
Behavior	
ControlToValidate	ProductNameTextBox
EnableClientScript	True
Enabled	True
EnableTheming	True
EnableViewState	True
InitialValue	
SetFocusOnError	True



Validation Controls cont...

- If we try and add a new product but leave the name field blank the error appears and the cursor goes straight to that field
- Multiple validation controls can be linked to one field
- Instead of field at a time, all errors can be reported as a group
- Obviously, this is basic physical validation
- Complex logical validation (ie checking data in the DB before inserting) is more complex and beyond the capacity of these built-in controls

The screenshot shows a web form for adding a new product. The fields are: ProductName, SupplierID, CategoryID, QuantityPerUnit, UnitPrice, UnitsInStock, UnitsOnOrder, ReorderLevel, and Discontinued (a checkbox). The ProductName field is empty, and a red error message "Product Name cannot be left blank" is displayed to its right. At the bottom, there are "Insert" and "Cancel" buttons.



User Login and Management

- As stated in a previous lecture, user management can be one of the most time consuming function points to include in an application
- IDE's like VS 2008 include some core Controls to deal with user login and authentication, sessions management, user creation and password recovery
- These can be hooked to existing databases of valid users, or the system can create an authentication model on the fly
- Again, a certain level of customisation is available, but for complex or unusual user management tasks, the default Controls may not work

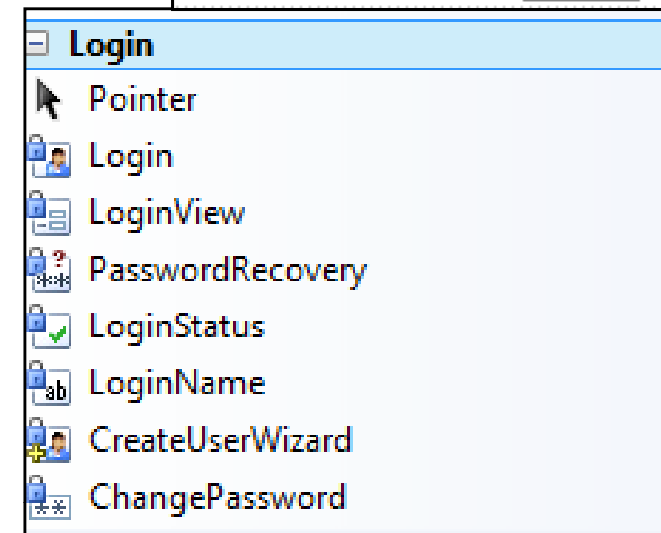
Log In

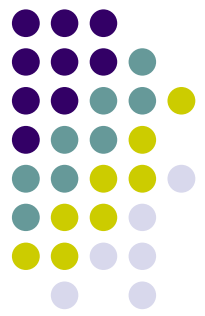
User Name: *

Password: *

☐ Remember me next time.

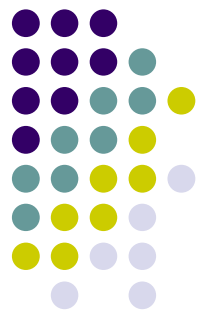
Log In





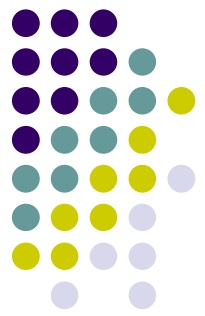
Issues with Visual IDE's

- These basic examples have shown the types of Controls and Components that modern visual IDE's can provide
- In these cases, the default Controls provide easy drag and drop development of tasks which would otherwise be very time consuming to code by hand
- However, there are drawbacks
 - Whilst being as generic as possible, such Controls or Components may not suit all scenarios
 - Though customisable, they do tend to have a 'generic' look
 - Experience would indicate that while you can do 90% of your developing using drag and drop, the last 10% can be very tricky (and time consuming)



Issues with IDE's cont...

- The environments themselves can have significant processing requirements (ie need expensive machines to run on)
- Are usually platform dependent
 - In the case of ASP.Net – server dependent as well (IIS 7)
- For novice developers, visual environments can obscure coding structures and make 'learning to code' more difficult
- In most cases, visual environments are not designed to replace coding knowledge but rather to complement such knowledge and speed up time consuming tasks
- As many of these examples have shown, good database structures are essential when using data driven Controls – but database design usually means non-functional applications



Conclusion

- Different visual IDE's offer different sets of Controls and functionality
- However, the aim of most such environments is to encapsulate complex and time consuming tasks into simple to use, drag and drop components
- Environments such as VS 2008 have a huge range of Commercial Off The Shelf (COTS) Control and Control packs that can be purchased to add to the default ones that came with the system
- Whilst these can be expensive, they can pay for themselves in saved development time and testing costs
- Visual development environments should be used where they will save time, effort and money on a development project
- In most development houses they are used along side traditional text coding environments