

# Master Pages

---

## CGT 456

Advanced Web Programming, Development, & Database Integration

Lecture 10



# What is a master page?

---

- ❑ Master pages allow you to create a consistent layout for the pages in your application.
- ❑ Used to control the overall design of your site.
- ❑ A single master page defines the look and feel and standard behavior that you want for all of the pages (or a group of pages) in your application.



# What is a master page?

---

- You can then create individual content pages that contain the content you want to display.
- When users request the content pages, they merge with the master page to produce output that combines the layout of the master page with the content from the content page.



# Master pages consist of...

---

- Two pieces:
  - The master page itself and
  - One or more content pages



# How master pages work

---

- A master page is an ASP.NET file with the extension .master (for example, Lab06.master) with a predefined layout that can include:
  - static text
  - HTML elements
  - server controls.



# How master pages work

---

- The master page is identified by a special @ Master directive that replaces the @ Page directive that is used for ordinary .aspx pages.
  
- The directive looks like:

`<% @ Master Language="C#" %>`



# Advantages

---

- Master pages provide functionality that developers have traditionally created by copying existing code, text, and control elements repeatedly; using framesets; using include files for common elements; using ASP.NET user controls; and so on.



# Advantages

---

## □ Advantages of master pages include:

- They allow you to centralize the common functionality of your pages so that you can make updates in just one place.
- They make it easy to create one set of controls and code and apply the results to a set of pages. For example, you can use controls on the master page to create a menu that applies to all pages.
- They give you fine-grained control over the layout of the final page by allowing you to control how the placeholder controls are rendered.
- They provide an object model that allows you to customize the master page from individual content pages.





# Run-time: order of operations

---

- At run time, master pages are handled in the following sequence:
  1. Users request a page by typing the URL of the content page.
  2. When the page is fetched, the @ Page directive is read. If the directive references a master page, the master page is read as well. If this is the first time the pages have been requested, both pages are compiled.
  3. The master page with the updated content is merged into the control tree of the content page.
  4. The content of individual Content controls is merged into the corresponding ContentPlaceHolder control in the master page.
  5. The resulting merged page is rendered to the browser.



# Navigation

---

- The master page is a good place to have your navigation since the master page template is merged with the rest of your content.



# Creating a Master Page

---

- Just like adding any other item
  - Right click solution > add new item
  - Choose Master Page
  
- Notice: At the top of the page is an @ Master declaration instead of the @ Page declaration normally found at the top of ASP.NET pages

# Creating a Master Page

The screenshot displays the Microsoft Visual Studio IDE with the following components:

- Target Schema for Validation Drop Down List:** A dropdown menu is open, showing options: Internet Explorer 6.0, Internet Explorer 6.0, HTML 4.01, XHTML 1.0 Transitional, XHTML 1.0 Frameset, and XHTML 1.1. This menu is circled in red.
- Code Editor:** Contains the following ASP.NET code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1" >  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head runat="server">  
<title></title>  
<asp:ContentPlaceHolder id="head" runat="server">  
</asp:ContentPlaceHolder>  
</head>  
<body>  
<form id="form1" runat="server">  
<div>  
<asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">  
</asp:ContentPlaceHolder>  
</div>  
</form>  
</body>  
</html>
```
- Solution Explorer:** Shows the project structure: Solution 'MasterPagesV1' (1 project) > C:\...MasterPagesV1 > MasterPagesV1.master, web.config.
- Properties Window:** Shows the properties for the selected master page:

<MASTER>	
AutoEventWir	True
ClassName	
ClientIDMode	Inherit
CodeBehind	
CodeFile	MasterPagesV1.ma
CodeFileBaset	
CompilationM	
CompilerOpti	
Debug	
EnableThemir	
EnableViewSt	
Explicit	
Inherits	MasterPagesV1
Language	C#
LinePragmas	
MasterPageFil	
Src	
Strict	
ViewStateMoc	
WarningLevel	
- Error List:** Shows 0 Errors, 0 Warnings, and 0 Messages.



# Creating a Master Page

---

- In design view:
  - You can click the page, then use the properties panel to set properties of the page: for example:
    - bgcolor
    - margins
    - text color
    - etc

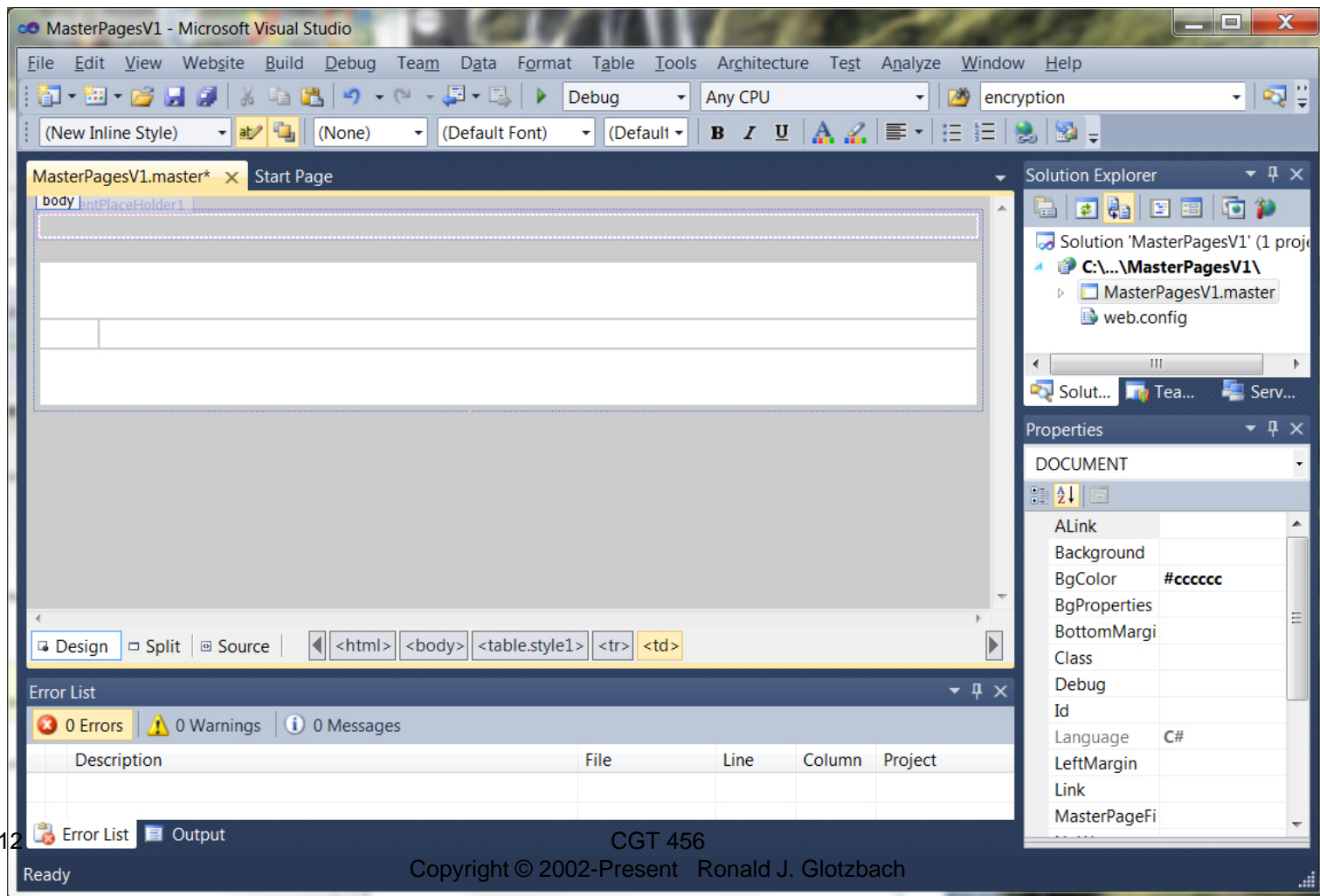


# Creating a Master Page

---

- In design view
  - Place a layout table
    - Do not place it in ContentPlaceHolder

# Creating a Master Page





# Creating a Master Page

---

- Add content to the page
  - Header info
  - Menu
  - Footer info
  - etc





# Creating a Master Page

---

## □ Toolbox

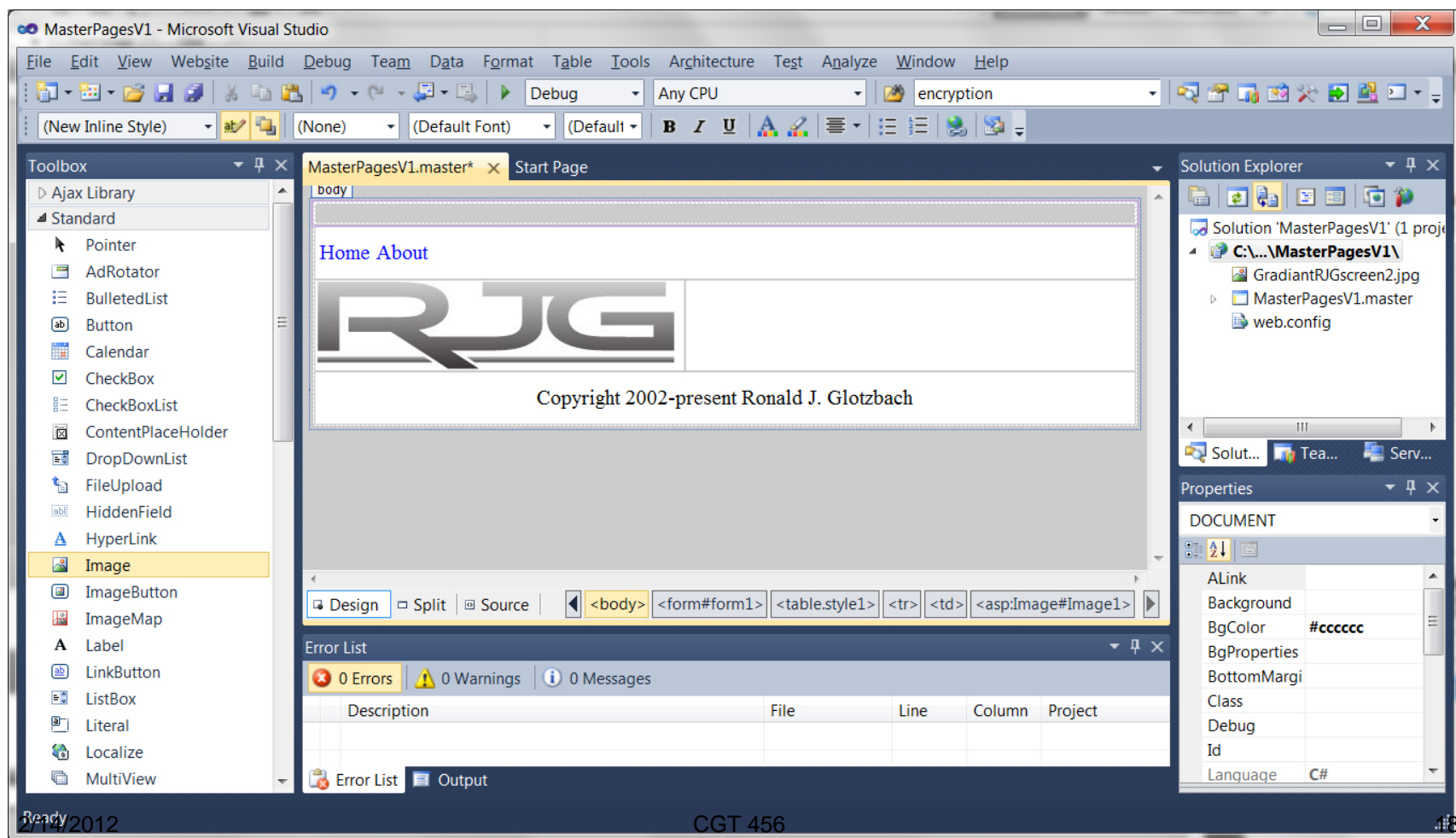
### ■ Navigation

- Menu > drag and drop
  - Change orientation in properties
  - Add root items using smart tag menu
  - Change their text

### ■ Standard

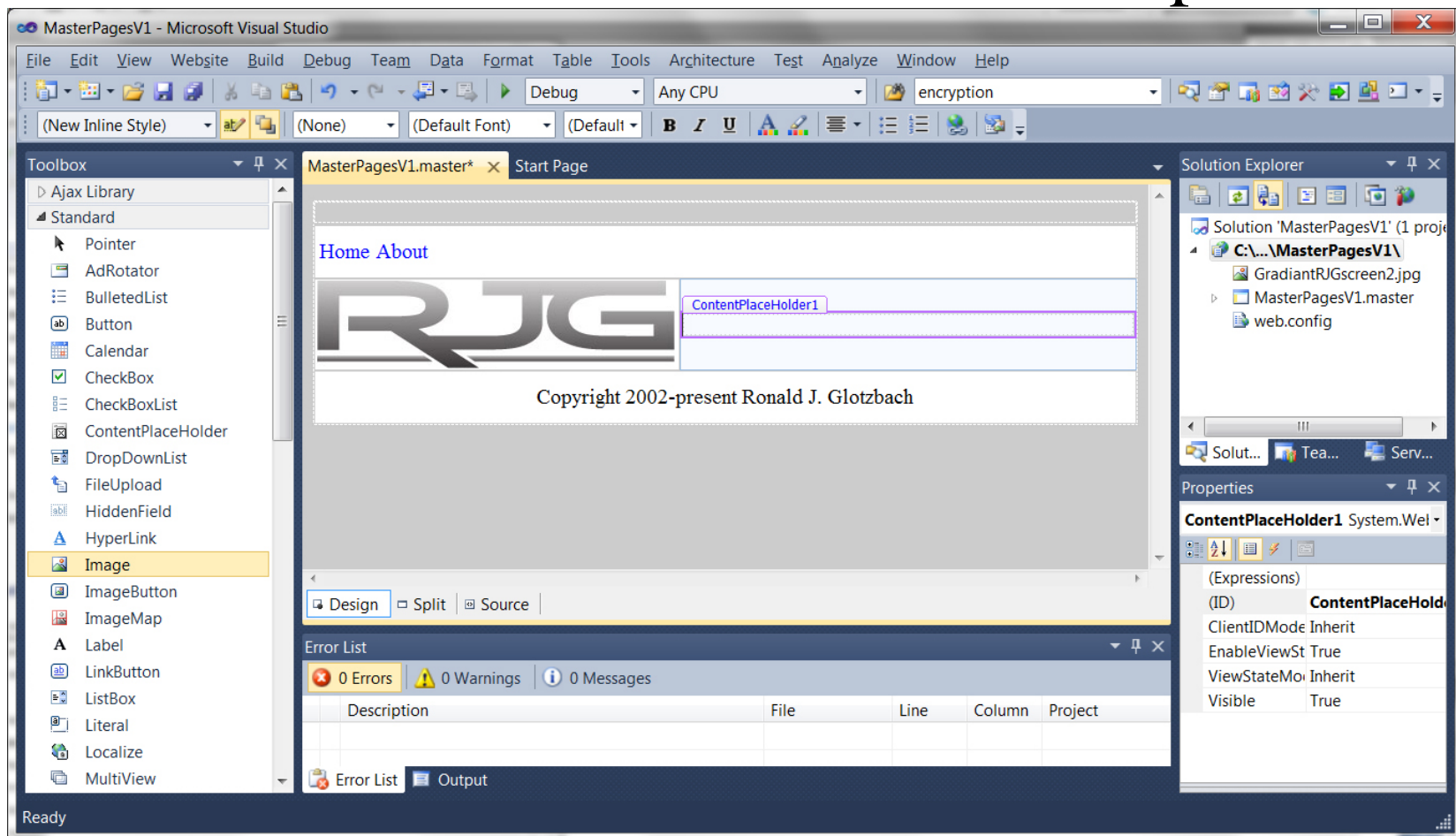
- Image > drag and drop
  - Set imageUrl to point to image
  - Note: you'll have to add existing item in order to add the image to your solution

# Creating a Master Page



# Creating a Master Page

- Move the ContentPlaceHolder into position





# Creating a Master Page

---

- ❑ The master page provides the template for your content.
- ❑ You define content for the master page by creating an ASP.NET page that is associated with the master page.
- ❑ The content page is a specialized form of an ASP.NET page that contains only the content to be merged with the master page.
- ❑ In the content page, you add the text and controls that you want to display when users request that page.

# Creating a Master Page

---

## □ Add Content Page

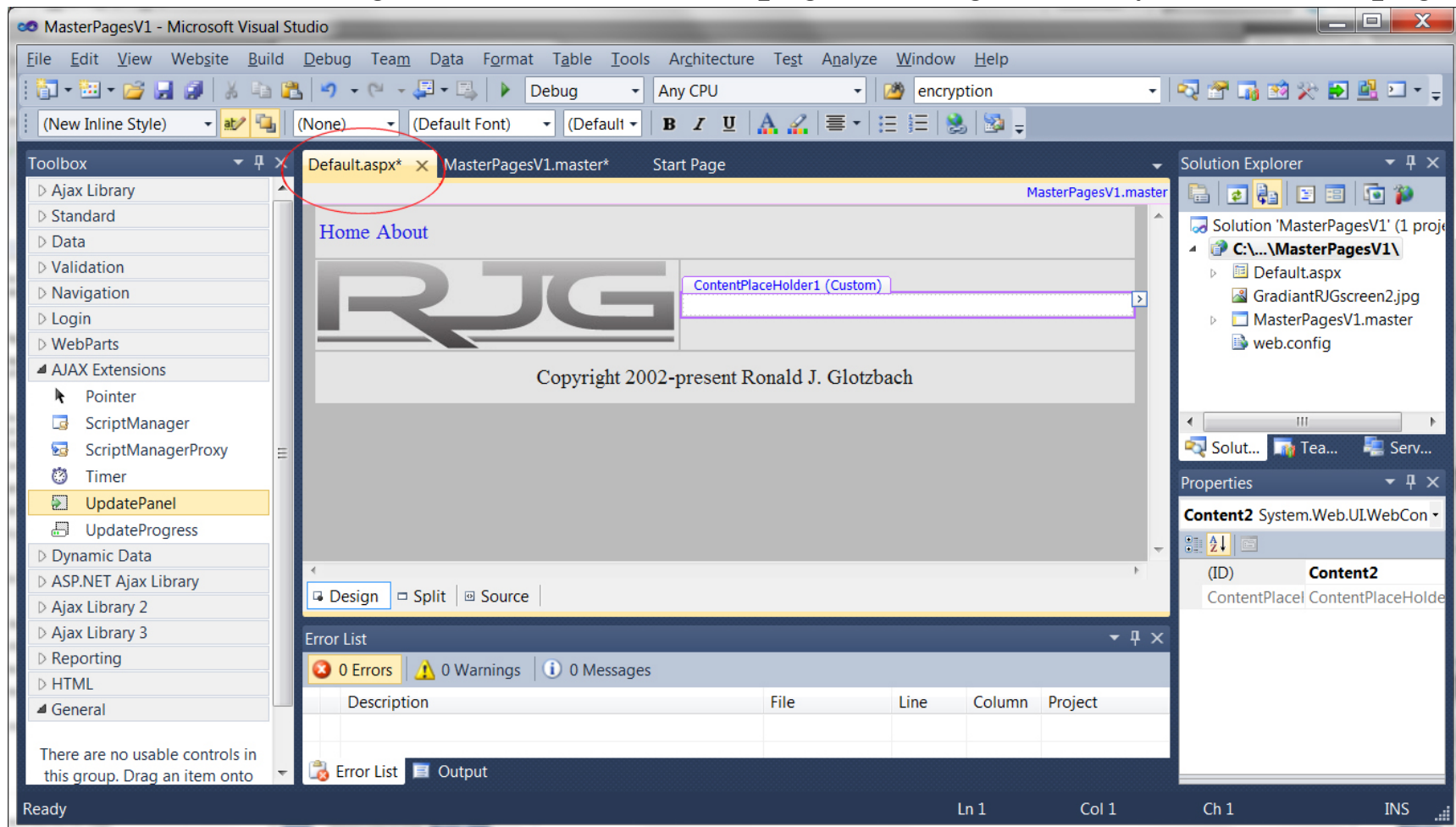
- Add new item > web form >
  - Check “select a master page”
  - Choose your master page
- The content page contains an @ Page directive that attaches the current page to the selected master page with the MasterPageFile attribute.

```
<% @ Page Title="" Language="C#" MasterPageFile="~/MasterPageV1.master" ... %>
```

- Set the page title to something appropriate

# Creating a Master Page

- Switch to design view > the master page is merged with your content page



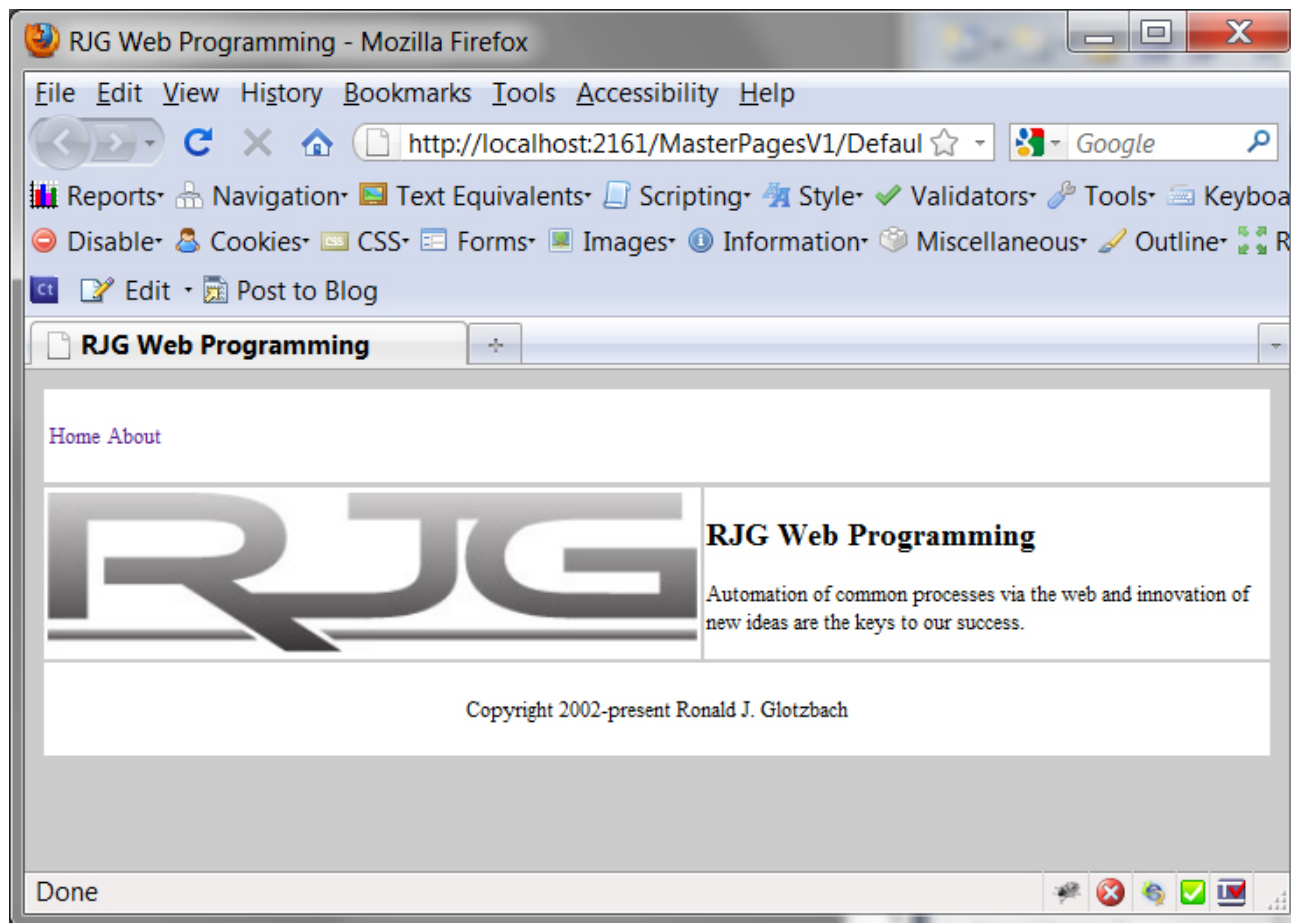
# Creating a Master Page

## □ Add content

The screenshot shows the Microsoft Visual Studio interface for a project named 'MasterPagesV1'. The main window displays the 'Start Page' of the master page, which contains a content placeholder labeled 'ContentPlaceHolder1 (Custom)'. The placeholder contains the text 'RJG Web Programming' and a paragraph: 'Automation of common processes via the web and innovation of new ideas are the keys to our success.' Below this is the copyright notice 'Copyright 2002-present Ronald J. Glotzbach'. The 'Paragraph' menu is open, showing various text formatting options. The 'Solution Explorer' on the right shows the project structure, including 'Default.aspx', 'GradientRJGscreen2.jpg', 'MasterPagesV1.master', and 'web.config'. The 'Properties' window on the right shows the properties for the selected paragraph element, such as 'AccessKey', 'Align', 'AtomicSelect', 'Class', 'ContentEditable', 'Dir', 'HideFocus', and 'Lang'. The 'Error List' at the bottom shows 0 errors, 0 warnings, and 0 messages. The status bar at the bottom indicates 'Ready', 'Ln 1', 'Col 1', 'Ch 1', and 'INS'.

# Creating a Master Page

- View in the browser







# Creating a Master Page

---

- ❑ Code in the content pages can reference members on the master page, including any public properties or methods and any controls on the master page.
- ❑ Create a property on the master page, and then use the value of the property in the content pages.
- ❑ The premise is that the company name for the Web site is stored as a property in the master page, and any reference to the company name in the content pages is based on the master page property.

# Creating a Master Page

---

## □ View the code for master.cs

- This code creates a property named `CompanyName` for the master page. The value is stored in view state so that it is persisted between postbacks

```
public String CompanyName
{
    get { return (String) ViewState["companyName"]; }
    set { ViewState["companyName"] = value; }
}

void Page_Init(Object sender, EventArgs e)
{
    this.CompanyName = "RJG Web Programming";
}
```



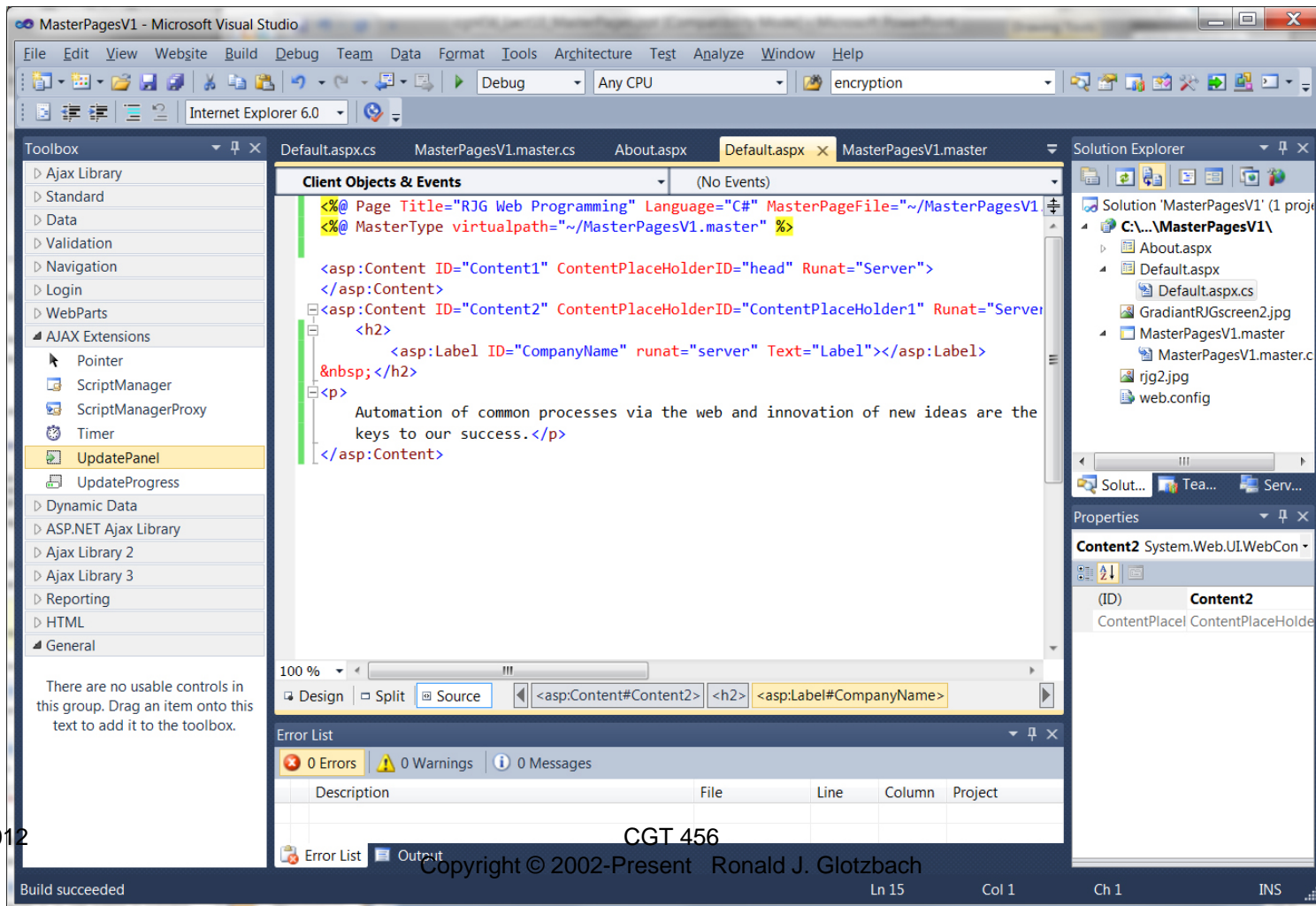
# Creating a Master Page

---

- Default.aspx.cs code view
  - At the top of the page, underneath the @ Page directive, add the following @ MasterType directive:  

```
<% @ MasterType virtualpath="~/MasterPagesV1.master" %>
```

# Creating a Master Page



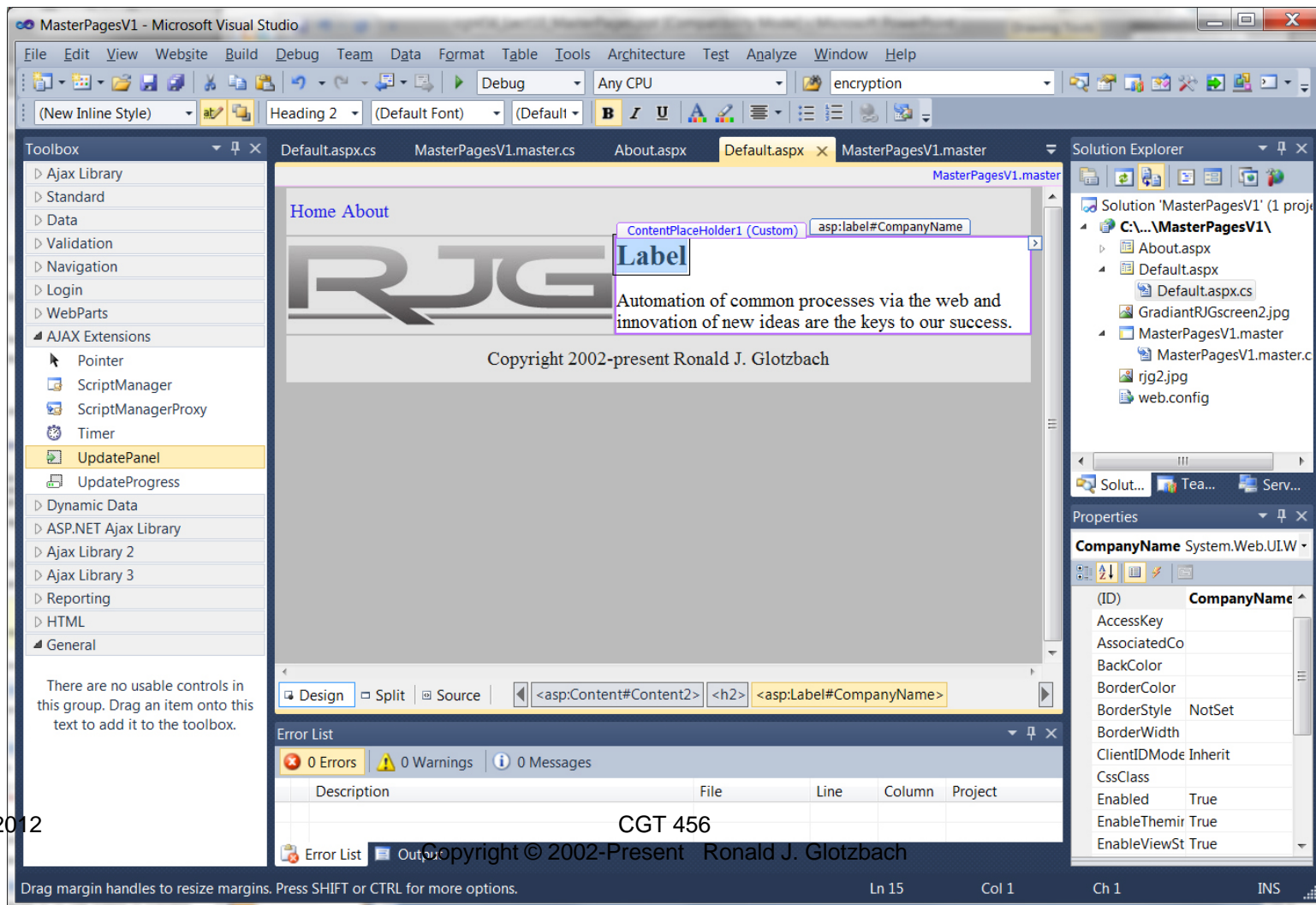
2/14/2012

CGT 456

Copyright © 2002-Present Ronald J. Glotzbach

28

# Creating a Master Page



2/14/2012

CGT 456

29



# Creating a Master Page

---

- In the Page\_Load of default.aspx.cs

```
protected void Page_Load(object sender, EventArgs e)
{
    CompanyName.Text = Master.CompanyName;
}
```