**Geçen Senenin Soruları Ve Çalışma Notları**

**\*\*ScriptManager vs ScriptManagerProxy kontrolleri, ne zaman? fark?**

ScriptManager ve ScriptManagerProxy -> Server ve Browser arasında bağlantı kurar. Ajax nesnelerinin çalışması için gereklidir. ScriptManager sadece bir sayfa için kullanılırken ScriptManagerProxy MasterPage de kullanılarak, masterpage'e bağlı olan sayfalar için kullanılır.

//Bununla ilgili çıkabilecek soruları buraya yazıyorum...

**UpdatePanel** -> sayfanın neresinde ajax işlemini yapılacağını belirler. UpdatePanel'in içindeki kısım server tarafına gider işlenir ve cevabı geri dönerek yenileme işlemi yapılır.

**b) Kısmi sayfa güncellemesinin işlemekte oldu kullanıcıya nasıl bildirilir?**

**UpdateProcess** ->Ajax işlemi uzun sürdüğü durumlarda UpdateProgress sayesinde kullanıcıya bilgi verilebilir. UpdatePanel ile ilişkilendirmek için AssociatedUpdatePanelId propertiesi kullanılır. örnek kod;

<asp:UpdateProgress Id="UpdatePro1" runat="server" AccocitedUpdatePanelId="UpdatePanel1">

<ProgressTemple>

<div class="PleaseWait">PleaseWait</div>

</ProgressTemplate>

</asp:UpdateProgress>

Timer -> Belirlenen zaman aralıklarında server tarafında bir işlemin yapılmasını sağlar. Mesela UpdatePanel'lin her 5 saniyede yenilenmesi gibi... (update panelin içinde)

<asp:Timer ID="Timer1" runat="server Internal=”1000” Event=”timer1”/>

**\*\* a) phoneHome ve phoneBusiness en az biri girilmiş mi? Custom validator, servervalidate event'ı. Kod?**

CustomValidator kullanının server ve client raraflı validation tanımlamasına olanak sağlar. Client taraflı bir event yada server taraflı olabilir.

protected void CustomValidator1\_ServerValidate(object source, ServerValidateEventArgs args){

if(!string.IsNullOrEmpty(phoneHome.Text) || !string.IsNullOrEmtpy(phoneBussiness.Text) { args.IsValid=true;}

else {args.IsValid=false;}

}

<script type="text/javascript">

function ValidatePhoneNumbers(source, args){

var phoneHome = document.getElementById(<%=phoneHome.ClientId%>);

var phoneBussines = document.getElementById(<%=phoneBusiness.clientId%>);

if(phoneHome!='' || phoneBusines.value != '') args.IsValid=true;

else args.IsValid = false;

}

</script>

Validationlarla ilgili çıkabilecek konular ve Pdf’in bütün özeti..

Validation Conttrols

Kullanıcının girdiği veriyi doğrulamak için server tarafında ve client tarafında verinin kontrol edilmesini sağlayan controllerdir. Server-side validation asıl işlemin yaptıldığı yerdir.

Çeşitleri....

\*RangeValidator - Kullanıcının belirlenenen değer aralıkları arasında veri girişi yapmasını kontrol eder.

<asp:RangeValidator ID="..." runat="." ControlToValidate="Range" ErrorMessage="Enter a number between 1 and 10" MaximumValiue="10" MinumumValue="1" Type="Integer" />

\*RegularExpressionValidator - Bir reqular expression belirtilerek ona uygun veri giriplip girilmediğini kontrol eder, mesela bir email adresinin kontrolü gibi.

<asp:ReqularExpressionValidator Id.. runat.. ControlToValidate="Email" ErrorMessage="Enter a valid e-mail address" ValidationExpression="\w+..."/>

\*CompareValidator -> iki controlünün değerinin aynı olup olmadığını kontrol eder. Örnek olarak girilen bir şifrenin, şifre tekrarı kısmınında aynı olup olmasını kontrol etmek gibi.

\*CustomValidator -> kullanının client taraflı veya server taraflı kontrol yazmasına olanak sağlayan kontroldür...

\*\*\*\*ValidationSummary -> Sayfada bulunun validation kontrollerinin hatalarının liste şeklinde görünmesini sağlayan kontroldür.

Gösterme şeklini belirlemek için ShowMessageBox ve ShowSummary kısımları vardır. Aynı zamanda HeaderText kısmınıda vardır.

Server tarafında Page.IsValid özelliği ile validation kontrolü yapılabilir...

Çıkabilmesi ihtimal konuların devamı..

Ajax nedir? server tarafına bir postback işlemi olmadan, sayfanın tamamını yada bir kısmındaki verinin yenilenmesi sağlayan teknolojidir.

Web Service -> Web service bir konumda methodu çağırmak ve varsa geriye bir değer döndürmek için kullanılan teknolojidir. Temel fikri farklı platformlardaki haberleşmeyi sağlamak için geliştirilmiştir.

Asp.nett'teki bir methodu web servisi olarak kullanmak için WebMethod attribute'u eklenir.

[WebMethod]

public string HelloWord(){ return "Hello Word";}

web servislerinin uzantısı .asmx'dir.

Web servisin client-side'da yani javascript kodlarında kullanmak için ScriptManager'ı configure etmek gerekiyor

<asp:ScriptManager ID="ScriptManager1" runat="server">

<Services>

<asp:ServiceReference Path="~/WebServices/NameServices.asmx" />

</Services>

</asp:ScriptManager>

ve methodun üstüne eklenen attribuate'da değişmesi gerekir.

[System.Web.Script.Services.ScriptServive]

Örnek script kodu; (scriptmanager kullanmadan yapmakda mümkün..)

<input id="yourNameText" type="text"/>

<input id="sayHelloButton" type="button" value="Say Hello"/>

<script type="text/javascript">

function HelloWord(){

var yourName=$get('yourNameText').value;

NameService.HelloWord(yourName,HelloWordCallback);

}

functionHelloWordCallback(result){

alert(result);

}

$addHandler($get('sayHelloButton'),'click',HelloWord);

</script>

script manager'a service referans eklemeden kullanmak istiyorsak, scriptmanager'ın **EnablePageMethods=True** özelliğini eklememiz gerekiyor.

\*\* a) Querystrig ne? Kısımları, örnekle. b) Label'da QueryString örneğini gösterecek kod?

Query String veriyi bir sayfadan diğerine göndermeye yarar. Name/Value çiftidir. url'nin sonuna soruişarti eklenekerek name=value şeklinde kullanılır. name/value çiftlerini ayırmak için '&' kullanılır. örnek kod

http://mysite.com/login.aspx?name=Ercan&surname=Turing

protected void Page\_Load(object sender, EventArgs e){

Label1.Text = Request.QueryString["name"];

}

Extra Bilgi

Get: Sayfalar arasında veri transferi için kullanılan bir yöntem, ilgili mesajlar query string'e eklenerek gönderilir.

Post. Sayfalar arasında veri transferi için kullanılan bi başka yöntem, gönderilen veriler query stringe değilde arka planda request'e eklenerek gönderilir.

Tek Soru a ve b si var.

\*\* a) Başka sayfaya yönlendirme methodları? farkları? b) Source.aspx'den Target.aspx yönlendir? Kod?

Client Side Redirections… Client’ın sayfanın url’sinde yönlendirildiği görebildği durumlar…

**- Response.Redirect** .. Status Code=302

- **Response.RedirectPermanet –** HtmlStatus Code=301 Page has moved permenently

-**Response.RedirectToRoute**( new { productid = "1", category = "widgets" );

… bunlar gibi birkaç tane daha var, overload edilmeş halleride var.

Server-Side Redirection

**Server.Transfer**(“Target.aspx?name=Jane”); kodunu source.aspx de çalıştırıldıysa linkde gözüken sayfa Source.aspx’dir ama içerik Target.aspx’e aittir…

Protected void Page\_Load(object sender, EventArgs e){

Response.Redirect(“Target.aspx?name=Jane”);

}

­­­­­­­­­­­­­­­­­­­Sorularda yok ama bilmekte fayda olduğnu düşündüğüm birkaç konu

\*\*User Control\*\* Nedir, Kod Örneği...

User Control > Html ve Asp.Net kodlarının birleştirerek kullanıcın kendi istediği şekilde yeni bir control tanımlamasıdır. (.)ascx uzantılıdır. Doğrudan çağrılamaz. Örnek;

Page Yerini Controle Bırakıyor.

<%@ Control Language="C#" AutoEventWireup="true" CodeFile="Banner.ascx.cs" Inherits="Controls\_Banner" %>

<asp:Panel ID="HorizontalPanel" runat="server">

<a href="http://bilmuh.ege.edu.tr" target="\_blank" runat="server" id="HorizontalLink">

<asp:Image ID="Image2" runat="server" AlternateText="This is a sample banner."

ImageUrl="~/Images/Banner\_468x60.jpg" />

</a>

</asp:Panel>

Bir sayfaya eklemenin yönemleri..

1.si

<%@ Register Src="ControlName.ascx" TagName="ControlName" TagPrefix="acu1" %>

<uc1:Banner Id="Banner1" runat="server" />

2. yöntem web.config'de tanımlamak, Tag'namei bütün web sayfasında gösterilebilir.

<pages thme="Monochrome">

<controls>

<add tagPrefix="Ege" tagName="Banner" src="~/Controls/Banner.ascx"/>

</control>

<pages>

**\*\* Authentication vs Authorization**

**Kelime Anlamları (ne anlama geldiğinde anlaşılmasına yardımcı olur diye düşündüm)  
Authentication** kimlik dogrulama ispatlama belgeleme  
**Authorization** yetki onay izin ruhsat anlamina gelir.

**Authentication :** sizin kim olduğunuzu ispatlar.

**Authorization** Sizin kim olduğunuza bağlı olarak, sistem size ilgili alanlara erişmek ve işlem yapmak için ayrıcalık tanır. Bunun için sistem sizin iki özelliğinizi bilmelidir.

1. Aktif kullanıcı için Permissions(izinler)
2. Kaynaklara erişmek için aktif kullanıcının Authorization rules(yetkilendirme kuralları)’nı

Kullanıcı ve sahip olduğu kurarlar sistemdekilerle eşleşirse granded, eşleşmezse denied olur.

**\*\*LogiStatus vs LoginView kontrolleri, ne zaman? fark?**

**LoginStatus** kullanıcının kimlik doğrulama durumunu algılar ve bilgisini tutar, kullanıcı login olmuş mu veya logout olmuş mu? Adı nedir gibisinden…

**LoginView** ise giriş yapmış(login) kullancılarla login olmamış kullanıcıların görüntüleyeceği şeyleri belirler. Hatta farklı rollere sahip olduklarında farklı şeyler gösterilebilir.

Örnek kod;

<asp:LoginView ID=”LoginView1” runat=”server”>

<AnonymousTemplate>

Hit here visitor. Üye olmak istermisiniz …

</AnonymousTemplate>

**<LoggedInTemplate>**

Sevgili üyemz hoşgeldiniz…

**</LoggedInTemplate>**

<RoleGroups>

<asp:RoleGroup Roles=”Managers”>

<ContentTemplate>

Sevgili Yöneticimiz Sitenizi Hoş Geldiniz…

</ContentTemplate>

</asp:RoleGroup>

</RoleGroups>

</asp:LoginView>

İşe yarar örnekler…

Asp.Net Application Services…

<asp:LoginName ID=”LoginName1” runat=”server” FormatString=”Kullanıcı Adınız {0}”/>

İf(User.IsInRole(“Manager”))… veya Roles.IsUserInRole(“Managers”) ..

RoleManager enable olmak zorunda

**\*\* b) Management sadece managers erişebiliyor, Can isimli kullanıcı ve Editors rolündeki bütün kullanıcılar erişsin. web.config kod?**

**Benzer kod örneği**

**<roleManager enable=”true”/> //web.config dosyasında farklı rollere sahip olması için**

**<location path="Management">**

**<system.web>**

**<authorization>**

**<allow roles="Managers"/>**

**<allow users="Ercan"/>**

**<deny users="\*"/>**

**</authorization>**

**</system.web>**

**</location>**

**</configuration>**

**\*\* a) Login olmuş, re-quthentication sağlayan cookie var, Timeout 1440dk ayarlamak için web config dosyası kodu?**

<authentication mode=”form”>

<forms timesout=”1440”/>

</authentication>

**\*\* Kullanıcı -> verileri gösterme, filitreleme, düzenleme için hangi kontrol iyi? Veritabanına nası bağlanılır?**

**GridView** kontrolü, çünkü otomatik pages desteği var. Ayrıca direk gridview üzerinden ekleme, çıkarma, sıralama ve güncelleme işlemleri de yapılabiliyor. Aynı zamanda hazırda var olan skin seçeneklerini de kullanılabilir.

Datasource kısmınandan bir source control seçilir. Hazırda yoksa new data source seçeneği seçilerek ilgili database bağlantısı sağlanılıp, ilgili tablosu seçilir…

Linq // bununla ilgili soru yok ama temel olarak özet geçiyorum

Language-Integrated Query .net programlama dilleri ile veri tabanı sorguları yapmamıza olanak sağlayan teknolojidir. .Net Framework 3.5 ile beraber kullanıma sunulmuştur. Select,From ve Where gibi keyword’leri vardır.

Temel olarak veri tabanı neslerini .net nesneleriymiş gibi alarak onlar üzerinde işlem yapmamızı sağlar.

**\*\*Mvc vs Web Forms?**

**Web Forms’un Avantajları**

- Hazır Controlleri Vardır, doğru kullanımda bize kolaylıklar sunabilir.

- ViewState desteği vardır, sayfayla ilgili bazı bilgiler bu teknoloji sayesinde saklanır.

- Controllerin event’ları vardır, hem client taraflı hemde server tarafında.

- Hazır kontroller olduğundan ve mantığı çok basit olduğundan kolay öğrenilir.

**Dezavantajları**

**-**Html ve codebehind kısımları ayrıldığından ve tasarım deseni öğrenme sürecinde kullanılmadığından dolayı spagetti kod yazılabiliyor.

-ViewState desteğinden dolayı server tarafında extra yük biniyor.

- Hazır kontrollerin doğru kullanılmaması, hatta doğru kullanılsa bile yavaşlamaya sebep olabiliyor. Tam olarak bir html hakimiyeti olmayışından dolayı.

- Url’de direk sayfa adresi gözüktüğünden dolayı arama motorları ve kullanıcı için çok iyi sayılmaz. Düzentilebilir ama uğraştırır.

**Mvc’in Avantajları**

-Öncelikli olarak bir tasarım deseni üzerinden gidildiğinden dolayı proje büyüse de çok fazla karmaşık hal alamayacaktır.

- Mvc viewstate desteklemez, bu yüzden daha performanslıdır. Helper’lar ile ihtiyaç karşılanabilir.

- Html üzerinde tam kontrole hakim, server controlleri desteklemez (hız açısından avantajlı). Html üzerinde full kontrol olduğundan id özelliğini direk olarak kullanır. Bu sayede javascript ve jquery gibi script ve kütüphanelerin kullanımı daha kolaydır.

- Model – View – Control yapısı olduğundan dolayı takım halinde bir projenin geliştirilmesi çok daha kolay olur. Back-end developerlar control kısmını geliştirirken front-end’lerde view kısmı ile ilgilenebilir.

- Güçlü ve kolay bir routing’e sahiptir. Arama motorları ve developer için kolaylık sağlar.

- Ve Mvc’nin yeni versiyonları ile gelen Razor engin gibi, Authentication’ı sağlayan thirt parti kütüphaneler sayesinde birçok kolaylık da sağlanmıştır. Bence en büyük avantajı bunlar. Ayrıca istenirse mvc projesine web form’da eklenebilir.

**Dezavantajları**

- Öğrenme süreci daha zor ve uzundur.

View’e gönderilen model yapısı kod tekrarına çok kolay izin verir ve bazen bu karmaşaya sebep olabilir.