Ajax : تحولی بزرگ در عرصه وب (بخش هشتم)

آنچه تاکنون گفته شده است :

بخش اول	تاثیر متقابل وب و نرم افزار بر یکدیگر
بخش دوم	Ajax و فناوری های مرتبط با آن
بخش سوم	بررسی نمونه برنامه های مبتنی بر Ajax
بخش چهارم	برنامه نویسی غیرهمزمان در برنامه های وب
بخش پنجم	بررسی معماری ASP. NET Ajax
بخش ششم	بررسی مدل پیاده سازی با محوریت سرویس گیرنده و مدل پیاده سازی با محوریت سرویس دهنده .
بخش هفتم	ایجاد صفحات وب مبتنی بر Ajax با تاکید بر روی پتانسیل های سمت سرویس دهنده فریمورک Ajax ASP.NET

در بخش هفتم با نحوه ایجاد یک صفحه مبتنی بر Ajax با تمرکز بر روی پتانسیل های سمت سرویس دهنده فریمورک ASP. NET Ajax آشنا شدیم . بدین منظور یک نمونه مثال ساده را بررسی کردیم که در آن از کنترل های UpdatePanel و UpdateProgress به منظور نشان دادن قابلیت Ajax در صفحات ASP. NET استفاده شده بود .

در این بخش با نحوه ایجاد یک صفحه مبتنی بر Ajax با تمرکز بر روی پتانسیل های سمت سرویس گیرنده فریمورک ASP. NET Ajax ، آشنا خواهیم شد .

مقدمه

رویکرد مبتنی بر سرویس دهنده در عین سادگی و شفافیت دارای چالش های مختص به خود با توجه به اصول اولیه تعریف شده در خصوص برنامه های مبتنی بر Ajax است . پیاده سازی نرم افزارهای مبتنی بر Ajax زمانی که فرصت و یا بهتر بگوئیم شرایط اجرای اکثر ماژول های برنامه در مرورگر (در مقابل سرویس دهنده) فراهم گردد ، نتایج بمراتب موثرتری را به دنبال خواهند داشت .

فراموش نکنیم یکی از مهمترین اصول برنامه های Ajax ، عرضه هوشمندانه تر برنامه ها توسط مرورگر می باشد و به همین دلیل است که نقش سرویس دهنده به ارسال داده مورد نیاز جهت بهنگام سازی بخش رابط کاربر محدود شده است . بدیهی است تفکر فوق و تحقق عملی آن در زمان طراحی و پیاده سازی اینگونه برنامه ها ، کاهش محسوس مبادله داده بین سرویس دهنده و مرورگر را به دنبال خواهد داشت .

مدل پیاده سازی با محوریت سرویس گیرنده

همانگونه که در بخش ششم اشاره گردید ، در این مدل ، لایه Presentation متاثر از اسکریپت های سمت سرویس گیرنده با بکارگیری DHTML و جاوا اسکریپت می گردد. این بدان معنی است که یک برنامه با هوشمندی و تعامل بیشتر ، از طریق سرویس دهنده برای سرویس گیرنده ارسال می گردد (در زمان استقرار صفحه در حافظه برای مرتبه اول) . پس از آن ، تعامل بین برنامه مرورگر و سرویس دهنده محدود به بازیابی داده مورد نیاز جهت بهنگام

سازی صفحه است . در این مدل کاربران با برنامه تعامل زیادی خواهند داشت (برنامه ای که در سمت سرویس گیرنده و در مرورگر کاربر اجراء شده است) .
برای آشنائی عملی با مدل فوق ، در ادامه یک نمونه مثال ساده و در عین حال کاربردی را بررسی خواهیم کرد. در این مثال با فراخوانی یک سرویس وب از طریق کد سمت سرویس گیرنده ، داده مورد نیاز جهت به‌نگام سازی بخش رابط کاربر از سرویس دهنده دریافت خواهد شد .
قبل از تشریح مثال فوق ، بد نیست در ابتدا با برخی از مفاهیم کلیدی و مهم در رابطه با بکارگیری سرویس های وب از طریق کد سمت سرویس گیرنده در برنامه های وب مبتنی بر Ajax آشنا شویم .

دستیابی به سرویس های وب از طریق کد سمت سرویس گیرنده

ASP. NET Ajax روش های متعددی را به منظور فراخوانی سرویس های وب ارائه می نماید . جاوا اسکریپت ، XML Script و AutoCompleteExtender نمونه هائی در این زمینه می باشند . با استفاده از امکانات ارائه شده در فریمورک فوق ، پیاده کنندگان می توانند سرویس های وب را (فایل هائی با انشعاب asmx .) از طریق مرورگر و به کمک کدهای سمت سرویس گیرنده فراخوانده و از پتانسیل های آنها در برنامه های وب با هدف بهبود بخش رابط کاربر استفاده نمایند .
در چنین مواردی ، یک صفحه می تواند متدهای سمت سرویس دهنده را بدون انجام postback و نیاز به بازخوانی تمامی صفحه فراخوانده و از آنها استفاده نماید ، چراکه صرفا" داده بین مرورگر و سرویس دهنده وب مبادله خواهد شد .
بکارگیری کلاس های پراکسی جاوا اسکریپت یکی از روش های فریمورک ASP.NET Ajax برای فراخوانی سرویس های وب است .بدین ترتیب می توان یک متد از سرویس وب موجود در سمت سرویس دهنده را با فراخوانی متد کلاس پراکسی جاوا اسکریپت مرتبط با آن فراخواند .
ASP. NET Ajax ، همچنین امکانات لازم برای جاوا اسکریپت به منظور فراخوانی سرویس هائی نظیر پروفایل و membership را ارائه می نماید .
در مثالی که در ادامه بررسی خواهیم کرد ، از طریق جاوا اسکریپت در سمت سرویس گیرنده ، متد مورد نیاز سرویس وب در سمت سرویس دهنده صدا زده می شود .

آشنائی با پراکسی های ASP.NET AJAX Web Service

کد پراکسی دارای یک نقش مهم و حیاتی در ارسال و دریافت پیام از سرویس های وب است . فراخوانی یک سرویس وب با استفاده از پراکسی سمت سرویس گیرنده یکی از ویژگی های مهم فریمورک ASP. NET Ajax در سمت سرویس گیرنده است . در دات نت ، با استفاده از wsdl.exe و یا بکارگیری گزینه Add Web Reference در ویژوال استودیو ، امکان ایجاد پراکسی فراهم می گردد . پراکسی های ASP. NET Ajax با روش های فوق تولید نخواهند شد و برای ایجاد آنها می بایست از خصلت Service کنترل ScriptManager استفاده کرد .

فعال کردن قابلیت فراخوانی سرویس های وب به کمک جاوا اسکریپت

برای فراخوانی سرویس های وب از طریق اسکریپت می بایست پیکربندی لازم را به کمک فایل web.config انجام داد . بدین منظور لازم است ScriptHandlerFactory HTTP handler را از طریق فایل web.config ، ریجستر کرد تا قادر به پردازش درخواست هائی باشد که از سمت سرویس گیرنده و به کمک اسکریپت ها قصد استفاده از

سرویس های وب را دارند. در مواردی که فراخوانی یک سرویس وب توسط ASP.NET Ajax صورت نمی پذیرد ، درخواست مربوطه به هندلر پیش فرض ارجاع داده می شود.
کد زیر ، یک نمونه فایل web.config به همراه handler مربوطه را نشان می دهد .

```
<system.web>
  <httpHandlers>
      <remove verb="*" path="*.asmx"/>
      <add verb="*" path="*.asmx" type="System.Web.Script.Services.ScriptHandlerFactory" validate="false"/>
 </httpHandlers>
<system.web>
```

توجه داشته باشید در زمان ایجاد یک وب سایت مبتنی بر Ajax در ویژوال استودیو نسخه های 2005 و یا 2008 ، تنظیمات فوق بطور اتوماتیک در فایل web.config اعمال خواهد شد .
پس از اعمال تغییرات فوق ، برای فعال کردن قابلیت فراخوانی یک سرویس وب از طریق اسکریپت در یک صفحه ASP.NET ، مراحل زیر را می بایست انجام داد :

- **مرحله اول** : اضافه کردن کنترل ScriptManager بر روی صفحه
- **مرحله دوم** : اضافه کردن یک مرجع به سرویس وب توسط عنصر asp:ServiceReference و تنظیم خصلت path آن به گونه ای که به سرویس وب اشاره نماید . شی ServiceReference به ASP.NET Ajax اعلام می نماید که یک کلاس پراکسی جاوا اسکریپت را برای فراخوانی سرویس وب مورد نظر توسط اسکریپت ، تولید نماید .

کد زیر ، نحوه فراخوانی یک سرویس وب با نام Test.asmx توسط اسکریپت را نشان می دهد .

```
<asp:ScriptManager runat="server" ID="scriptManager">
  <Services>
    <asp:ServiceReference path="~/WebServices/Test.asmx" />
 </Services>
</asp:ScriptManager>
```

در زمان تفسیر صفحه ای که شامل عنصر <asp:ScriptManager> می باشد ، یک کلاس پراکسی جاوا اسکریپت برای سرویس وب Test.asmx ایجاد می گردد . کلاس پراکسی ، دارای متدهای مرتبط با هر یک از متدهای موجود در سرویس وب Test.asmx می باشد . صفحه همچنین شامل کلاس های پراکسی جاوا اسکریپت مرتبط با نوع های داده سرویس دهنده است که به عنوان پارامتر ورودی و یا مقادیر برگردانده شده توسط متدهای سرویس وب مورد استفاده قرار می گیرد . بدین ترتیب ، اسکریپت قادر به مقداردهی اولیه پارامترها و برگرداندن مقادیر خواهد بود .
پس از این مقدمه نسبتا" طولانی و شاید هم خسته کننده ! ولی ضروری ، اجازه دهید در ادامه به منظور آشنائی با قابلیت های فریمورک ASP.NET Ajax در سمت سرویس گیرنده ، یک نمونه مثال کاربردی را با یکدیگر دنبال نمائیم .

مثال : ایجاد یک صفحه مبتنی بر Ajax با تمرکز بر روی پتانسیل های سمت سرویس گیرنده

در این مثال می خواهیم تعداد مقالاتی را که بر روی سایت سخا روش و در هر یک از گروه های مختلف منتشر شده است ، به اطلاع ملاقات کنندگان سایت برسانیم .برای سادگی کار ، فرض می شود که تعداد مقالات منشتر شده بر روی سایت از طریق بانک اطلاعاتی بازیابی نخواهد شد و در مقابل از یک کلاس با نام Maghalat برای بازیابی تعداد مقالات منشتر شده در هر گروه استفاده خواهیم کرد . کلاس فوق دارای صرفا" یک متد با GetNumberOfMaghalat است که وظیفه آن برگرداندن تعداد مقالات منتشر شده در هر گروه است . برای دستیابی به خدمات این متد از یک سرویس وب با نام MaghalatService.asmx استفاده خواهیم کرد که از طریق کد سمت سرویس گیرنده فعال خواهد شد .

پس از آشنائی اولیه با صورت مسئله ، مراحل زیر را برای پیاده سازی یک صفحه وب مبتنی بر Ajax با محوریت سرویس گیرنده ، دنبال می نمائیم .

مراحل اول (ایجاد سایت) و **دوم** (طراحی و پیاده سازی کلاس Maghalat) همانند مثال ارائه شده در بخش هفتم است .

مرحله سوم : تعریف یک سرویس وب

برای جستجو و یافتن تعداد مقالات منتشر شده در هر گروه از یک سرویس وب با نام MaghalatService.asmx استفاده خواهیم کرد که متد GetNumberOfMaghalat کلاس Maghlat را فرامی خواند . کد سرویس وب فوق در جدول زیر نشان داده شده است .

```
<%@ WebService Language="VB" Class="MaghalatService" %>
Imports System
Imports System.Web
Imports System.Web.Services
Imports System.Web.Services.Protocols
Imports System.Web.Script.Services

<ScriptService()> _
Public Class MaghalatService
    Inherits System.Web.Services.WebService
    <ScriptMethod()> _
    <WebMethod()> _
    Public Function GetNumberOfMaghalat(ByVal articleGroup As String) As Integer
        System.Threading.Thread.Sleep(1000)
        Return Maghalat.GetNumberOfMaghalat(articleGroup)
    End Function
End Class
```

توضیحات :

- namespace با نام System.Web.Script.Services بخشی از هسته فریمورک ASP.NET Ajax است که برخی از عملیات مبادله داده در شبکه و اسکریپت نویسی را کپسوله می نماید .
- از دو خصلت جدید ScriptService و ScriptMethod در زمان تعریف متد و کلاس سرویس وب استفاده شده است . با استفاده از خصلت های فوق ، فریمورک ASP.NET Ajax تشخیص می دهد که کدام بخش از سرویس ها توسط پراکسی های جاوا اسکریپت بکار گرفته شده است . وجود خصلت ScriptMethod ضروری نیست ولی با استفاده از آن می توان برخی تنظیمات متد را انجام داد .
- پرداختن به این موضوع که پراکسی چه چیزهائی را تولید می نماید ، خارج از حوصله این مقاله است. با نگاهی به انتهای پراکسی تعریف شده در سرویس وب فوق ،در انتها با متد GetNumberOfMaghalat مواجه می شویم . متد فوق به اسکریپت سمت سرویس گیرنده مکانیزمی را ارائه می نماید که بتواند متدهای وب موجود در سرویس وب را استفاده نماید . در زمان فراخوانی سرویس وب از مجموعه پارامترهای بمراتب بیشتری استفاده خواهد شد که پیاده کنندگان آنها را در سرویس وب تعریف نکرده اند .
- در صورت مشاده فایل ASMX در مرورگر به همراه سوئیچ /js ، پراکسی جاوا اسکریپت تولید شده توسط فریمورک برای سرویس فوق نشان داده می شود .

```
var MaghalatService=function() {
MaghalatService.initializeBase(this);
this._timeout = 0;
this._userContext = null;
this._succeeded = null;
this._failed = null;
}
MaghalatService.prototype={
_get_path:function() {
 var p = this.get_path();
 if (p) return p;
 else return MaghalatService._staticInstance.get_path();},
GetNumberOfMaghalat:function(articleGroup,succeededCallback, failedCallback, userContext) {
return this._invoke(this._get_path(),
'GetNumberOfMaghalat',false,{articleGroup:articleGroup},succeededCallback,failedCallback,userContext);
}}
MaghalatService.registerClass('MaghalatService',Sys.Net.WebServiceProxy);
MaghalatService._staticInstance = new MaghalatService();
MaghalatService.set_path = function(value) { MaghalatService._staticInstance.set_path(value); }
MaghalatService.get_path = function() { return MaghalatService._staticInstance.get_path(); }
MaghalatService.set_timeout = function(value) { MaghalatService._staticInstance.set_timeout(value); }
MaghalatService.get_timeout = function() { return MaghalatService._staticInstance.get_timeout(); }
MaghalatService.set_defaultUserContext = function(value) {
MaghalatService._staticInstance.set_defaultUserContext(value); }
MaghalatService.get_defaultUserContext = function() { return
MaghalatService._staticInstance.get_defaultUserContext(); }
MaghalatService.set_defaultSucceededCallback = function(value) {
MaghalatService._staticInstance.set_defaultSucceededCallback(value); }
MaghalatService.get_defaultSucceededCallback = function() { return
```

```
MaghalatService._staticInstance.get_defaultSucceededCallback(); }
MaghalatService.set_defaultFailedCallback = function(value) {
MaghalatService._staticInstance.set_defaultFailedCallback(value); }
MaghalatService.get_defaultFailedCallback = function() { return
MaghalatService._staticInstance.get_defaultFailedCallback(); }
MaghalatService.set_path("/Ajax1/MaghalatService.asmx");
MaghalatService.GetNumberOfMaghalat= function(articleGroup,onSuccess,onFailed,userContext)
{MaghalatService._staticInstance
```

مرحله چهارم : ایجاد یک صفحه ASP.NET

در ادامه یک صفحه aspx . را به منظور استفاده از امکانات ارائه شده در سرویس فوق ایجاد می نمائیم . در اولین گام می بایست قابلیت استفاده از Ajax در صفحه فعال گردد . بدین منظور از کنترل ScriptManager استفاده خواهیم کرد . در ادامه ، یک مرجع به سرویس وب توسط عنصر asp:ServiceReference اضافه کرده و مقدار خصلت path آن را MaghalatService.asmx در نظر می گیریم .

```
<asp:ScriptManager ID="ScriptManager1" runat="server" >
 <Services>
    <asp:ServiceReference Path="MaghalatService.asmx" />
 </Services>
</asp:ScriptManager>
```

در بخش ویژوال صفحه ASP.NET از عناصر سرویس دهنده استفاده نشده است و صرفا" از عناصر HTML که امکان دستیابی به آنها از طریق کدهای سمت سرویس گیرنده وجود دارد ، استفاده شده است .

```
<div align="center">
  <span style="font-size: 10pt; font-family: Tahoma">
          <strong dir="rtl" style="text-align: center">صفحه وب مبتنی بر ایجاد یک  Ajax <br />
     ((پیاده سازی با محوریت پتانسیل های سمت سرویس گیرنده فریمورک ASP.NET Ajax)) </strong>
    <br />
  </span>
 <br />
  <select id="ArticleGroup" size="5" style="font-size: 12pt; width: 146px; color: navy; font-family: Tahoma">
       <option value="Software">نرم افزار </option>
         <option value="Hardware" >افزار سخت  </option>
         <option value="Security">  امنیت اطلاعات  </option>
         <option value="Network">شبکه </option>
         <option value="others">سایر </option>
  </select>
</div>
 <br/>
<div align="center" style ="font-family :Tahoma ; font-size :small ">
    <span id="maghalatResults"></span>
    <span id="loading" style="display:none;">
```

```
        <img src="images/ajax1.gif" alt="" />
            در حال بارگذاری...
    </span>
</div>
```

با توجه به این که برای ایجاد بخش رابط کاربر به امکانات موجود در سمت سرویس دهنده نیاز نمی باشد ، صرفا" از عناصر HTML معمولی در مقابل کنترل های سرویس دهنده استفاده شده است . در صفحه فوق از عناصری نظیر Select (جهت ارائه لیست گروه مقالات) و Span (جهت نمایش فیدبک دیداری به کاربر در زمان بازیابی داده از سرویس دهنده) استفاده شده است .

برای دمیدن روح حیات در کالبد صفحه فوق از کد جاوا اسکریپت زیر استفاده شده است .

```
<script type="text/javascript">
 <!--
  var articleGroup = null;
  Sys.Application.add_load(page_load);
  Sys.Application.add_unload(page_unload);

function page_load(sender, e){
    articleGroup = $get("ArticleGroup");
    $addHandler(articleGroup, "change", articleGroup_onchange);
   }

function page_unload(sender, e){
 $removeHandler(articleGroup, "change", articleGroup_onchange);
  }

function articleGroup_onchange(sender, e){
  $get("maghalatResults").innerHTML = "";
  $get("loading").style.display = "block";
  var selectedValue = articleGroup.value;
  MaghalatService.GetNumberOfMaghalat(selectedValue,onSuccess);
  }

function onSuccess(result){
 $get("loading").style.display = "none";
 $get("maghalatResults").innerHTML = " گروه مقالات در تعداد :  " + result;
 }
//-->
</script>
```

توضیحات :

- با استفاده از دستورات (Sys.Application.add_load(page_load و (Sys.Application.add_unload(page_unload ، توابع مربوط به رویدادهای Load و Unload در مرورگر ریجستر شده است .
- فریمورک سمت سرویس گیرنده یک چرخه حیات مشابه با چرخه حیات صفحات ASP.NET را ارائه می نماید . در چنین مواردی می توان از رویداد Load به عنوان فرصتی جهت ریجستر کردن یک handler به منظور کنترل هر گونه تغییرات در لیست مقالات استفاده کرد.

```
articleGroup_onchange ,"change" ,addHandler(articleGroup$)
```

- از متد unload برای سلب مسئولیت از handler ریجستر شده استفاده شده است

```
articleGroup_onchange ,"change" ,removeHandler(articleGroup$)
```

- در کد فوق به دستورات جدیدی برخورد می کنیم که با علامت $ شروع شده اند . دستورات فوق ، اسامی مستعار و یا کوتاه شده ای می باشند که در نهایت به کد جاوا اسکریپت ترجمه خواهند شد . به عنوان نمونه، دستور $get همانند بکارگیری document.getElementById می باشد . یکی از مزایای این روش ، استقلال کدها از تفاوت های موجود بین مرورگرهای مختلف است .
- در ادامه ، به هندلر ریجستر شده ای برخورد می کنیم که پس از هر مرتبه انتخاب یک گروه مقاله توسط کاربر ، صدا زده می شود . در چنین مواردی ، سرویس وب MaghalatService.asmx صدا زده شده تا تعداد مقالات منتشر شده در گروه مقالات انتخاب شده توسط کاربر را برگرداند . اولین پارامتر ، گروه مقاله انتخاب شده توسط کاربر است و دومین پارامتر نام تابع callback است که در صورت اجرای موفقیت آمیز متد مربوطه در سرویس وب ، فراخوانده می شود .

```
MaghalatService.GetNumberOfMaghalat(selectedValue,onSuccess)
```

- در نهایت ، به کمک نتایج برگردانده شده بخش رابط کاربر بطور پویا بهنگام خواهد شد .

خروجی برنامه فوق که مشابه خروجی ارائه شده در بخش هفتم می باشد در شکل 1 نشان داده شده است .

شکل 1 : نمایش تعداد مقالات منتشر شده در هر گروه

جمع بندی

به منظور آشنائی اولیه علاقه مندان با Ajax و تاثیر آن در دنیای برنامه نویسی وب ، هشت مقاله بر روی سایت منتشر گردید . در مجموعه مقالات فوق ، پس از بررسی تاثیر متقابل نرم افزار بر وب و بالعکس ، Ajax را معرفی کردیم و این که از کجا آمده است و قصد آن پوشش چه مسائلی در عرصه برنامه های وب است . در ادامه با کنترل XMLHttpRequest آشنا شدیم و به ضرورت استفاده از یک فریمورک برای بکارگیری قابلیت های Ajax در برنامه های وب اشاره کردیم . در ادامه ، فریمورک ASP.NET Ajax را معرفی و با معماری آن آشنا شدیم . در نهایت به منظور آشنائی عملی با فریمورک فوق دو نمونه مثال کاربردی را با هدف آشنائی با پتانسیل های سمت سرویس دهنده و سرویس گیرنده بررسی کردیم .

هدف از ارائه مقالات فوق ، آشنائی اولیه با فریمورک Ajax ASP.NET بود . هم اینک علاقه مندانی که مطالب منتشر شده را مطالعه کرده اند دارای یک شناخت مناسب از فریمورک فوق بوده و می توانند از آن به عنوان یک زیرساخت علمی مناسب در ادامه راه خود استفاده نمایند .

قطعا" تمامی داستان به این نقطه ختم نخواهد شد و امیدواریم در آینده بتوانیم با انتشار مقالاتی دیگر با مزایای بکارگیری فریمورک فوق در برنامه های وب بطور کاملا" کاربردی آشنا شویم .