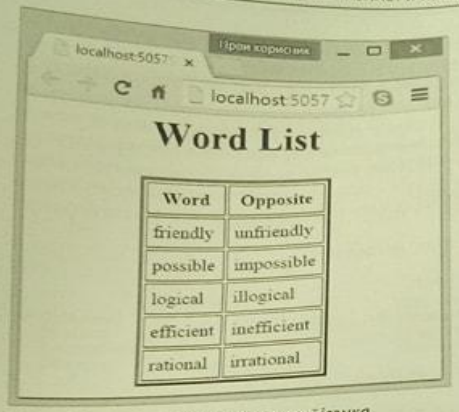
XML

- Kreiramo web sajt sa web stranicom na kojoj se prikazuje tabela sa spiskom reči i spiskom odgovarajućih reči sa suprotnim značenje.
- Reči čuvamo u XML fajlu i upotrebićemo kontrolu XML za njihov prikaz.



Веб-сајт са делом лекције из енглеског језика

XML је страндардни формат за размену података између различитих апликација. Као и документи који се креирају употребом описног језика *HTML*, XML документи се састоје од низа знакова који се најједноставније региструју на сваком рачунару. Ови документи такође користе тагове, али тагови нису предефинисани као што је то случај код описног језика *HTML*, већ могу да буду различити код сваког *XML* документа и основна сврха им је да се опишу подаци који се налазе у документу. Сами подаци могу да се налазе између тагова, или као вредност атрибута. Тако на пример, реч и њој супротну реч можемо да сачувамо обележене таговима на следећи начин:

```
<Word>
  <Value>friendly</Value>
  <Opposite>unfriendly</Opposite>
</Word>
```

Исте податке можемо и да чувамо и као вредности атрибута једног тага на следећи начин:

```
<Word Value="friendly" Opposite="unfriendly" />
```

У овом примеру смо се одлучили за други од два описана начина. Следи садржај *XML* фајла са називом *WordList.xml* који смо креирали у неком текстуалном едитору, на пример програму *Notepad*.

```
<?xml version="1.0" ?>
<WordList>
  <Word Value="friendly" Opposite="unfriendly" />
```

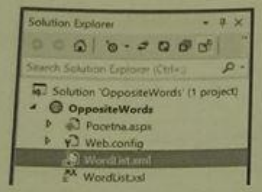
```
  <Word Value="possible" Opposite="impossible" />
  <Word Value="logical" Opposite="illogical" />
  <Word Value="efficient" Opposite="inefficient" />
  <Word Value="rational" Opposite="irrational" />
</WordList>
```

Податке које имамо у *XML* документу можемо да трансформишемо уз помоћ *XSL* фајла ради прегледног приказа на веб-страници. Следи *XSL* фајл *WordList.xsl* са описом како да се подаци из нашег *XML* фајла трансформишу у *HTML* документ са табелом коју ћемо приказати на веб-страници.

```
<?xml version="1.0" ?>
<xsl:stylesheet version="1.0"
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:msxsl="urn:schemas-microsoft-com:xslt"
xmlns:labs="http://labs.com/mynamespace">
  <xsl:template match="/">
    <html>
      <head>
        <title>Word List</title>
      </head>
      <body>
        <center>
          <h1>Word List</h1>
          <xsl:call-template name="KreirajZaglavlje"/>
        </center>
      </body>
    </html>
  </xsl:template>

<xsl:template name="KreirajZaglavlje">
  <table border="3" cellpadding="5">
  <tr>
    <th>Word</th>
    <th>Opposite</th>
  </tr>
  <xsl:call-template name="KreirajTabelu"/>
  </table>
</xsl:template>

<xsl:template name="KreirajTabelu">
  <xsl:for-each select="/WordList/Word">
  <tr>
    <td>
    <xsl:value-of select="@Value"/>
    </td>
    <td>
    <xsl:value-of select="@Opposite"/>
    </td>
  </tr>
```

```
    </xsl:for-each>
</xsl:template>

</xsl:stylesheet>
```

Када креирамо веб-сајт са једном веб-страницом, потребно је да сајту додамо креиране фајлове *WordList.xml* и *WordList.xsl*. Фајлове додајемо тако што их изаберемо у дијалогу који се појави када се кликне на ставку *Add/Existing Item* са менија који се добије када се кликне десни тастер миша над називом сајта у прозору *Solution Explorer*.



XML и XSL фајлови унутар сајта

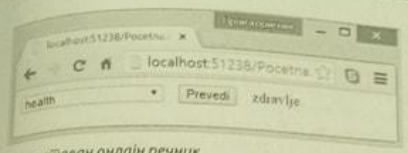
На веб-страницу *Pocetna.aspx* превучемо контролу *Xml* са списка *Toolbox*. Да би се приказали подаци из *XML* фајла у виду табеле на начин на који је то описано *XSL* фајлом за трансформацију, потребно је да контролу повежемо са креираним фајловима приликом учитавања веб-странице.

```
public partial class Pocetna : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Xml1.DocumentSource = "WordList.xml";
        Xml1.TransformSource = "WordList.xsl";
    }
}
```

Пример 6

Једна од најважнијих предности онлајн речника је брзина којом долазимо до превода речи која нас интересује. За разлику од књиге коју треба да пажљиво прелиставамо да бисмо стигли до жељеног превода, оналајн речници омогућавају једноставан унос речи, а њен превод добијамо за делић секунде. Као илустрацију у овом примеру ћемо креирати један врлоједноставан речник са преводом са енглеског на српски језик.

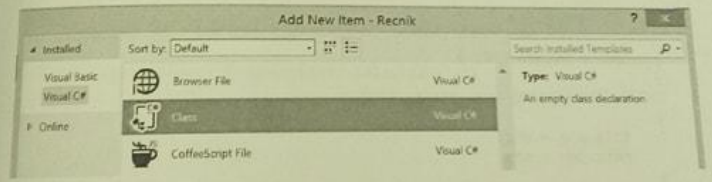
Веб-страница ће имати падајућу листу са речима на енглеском језику, а превод обележене речи са листе ћемо добити кликом на дугме.



Једноставан онлајн речник

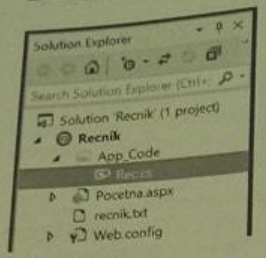
Контрола *DropDownList*, која служи за приказ падајуће листе, спада у групу контрола које се везују за податке. Постоје различити начини да се попуни падајућа листа, као и свака друга контрола која се везује за податке. У овом примеру ћемо падајућу листу попунити на основу садржаја текстуалног фајла *recnik.txt* који смо креирали у обичном текстуалном едитору. У једном реду фајла се налази реч на енглеском, а у следећем њен превод на српски. Садржај фајла ћемо једноставно учитати у листу објеката класе *Rec*, а затим ћемо падајућу листу везати за садржај те листе објеката.

Класа *Rec* ће имати два поља, реч на енглеском и одговарајућу реч на српском језику. Класу ћемо да креирамо унутар веб-сајта који смо направили за овај пример. Потребно је да изаберемо *Class* у дијалогу који се појави када изаберемо ставку *Add/Add New Item...* са менија који се добије када се кликне десни тастер миша над називом сајта у прозору *Solution Explorer*. У дијалогу је потребно још и да унесемо назив нове класе коју креирамо, у овом случају *Rec.cs*, уместо понуђеног назива *Class1.cs*.



Додавање нове класе веб-сајту

Постоје посебни фолдери које можемо да имамо унутар *ASP.NET* веб-сајта. Један од таквих фолдера је фолдер *App_Code*. Овај фолдер можемо и ми да додамо веб-сајту, а може да буде и аутоматски придодат када креирамо неки од фајлова са програмским кодом, као на пример, нову класу у овом примеру. Након затварања дијалога за додавање нове класе, добићемо дијалог са питањем да ли желимо да се наша класа чува у фолдеру *App_Code*. Фајлове са програмским кодом треба да чувамо унутар овог фолдера због заштите коју тај фолдер пружа. Фајлове унутар фолдера *App_Code* веб-сајт може да користи, али они нису доступни никоме споља. Списак свих фајлова нашег веб-сајта можете да видите у прозору *Solution Explorer* на следећој слици.



Фолдер App_Code

Класа *Rec* ће имати конструктор и два јавна својства, *Eng* и Srp, која омогућавају читање одговарајућих вредности приватних поља. Поред тога, класа ће имати и једну статичку методу *FormirajIzFajla* која ће служити за формирање динамичке листе објеката на основу садржаја текстуалног фајла. Да бисмо могли да радимо са текстуалним фајловима, потребно је да укључимо именски простор *System.IO*.

```
public class Rec
{
    private string eng, srp;

    public string Eng
    {
        get { return eng; }
    }

    public string Srp
    {
        get { return srp; }
    }
    public Rec(string eng, string srp)
    {
        this.eng = eng;
        this.srp = srp;
    }
    public static List<Rec> FormirajIzFajla(StreamReader f)
    {
        List<Rec> r = new List<Rec>();
        while (!f.EndOfStream)
        {
            string eng = f.ReadLine();
            string srp = f.ReadLine();
            Rec rec = new Rec(eng, srp);
            r.Add(rec);
        }
        return r;
    }
```

Приликом првог учитавања веб-станице, потребно је да падајућу листу повежемо са подацима. Као извор података наводимо назив динамичке листе објеката формиране на основу садржаја текстуалног фајла. За падајућу листу можемо да прецизирамо текст који ће се приказивати у њој везивањем својства *DataTextField* за јавно својство *Eng* класе *Rec*. Дакле, падајућа листа ће приказивати само списак енглеских речи. За сваку енглеску реч на списку имамо везану, у виду вредности, и њен превод на српски. То се постиже везивањем својства *DataValueField* за јавно својство *Srp* класе *Rec*. Након ових подешавања, потребно је да се позове метода *DataBind* којом се листа везује за податке, након чега ће падајућа листа бити попуњена речима. Сваки пут када се учита веб-страница, падајућа листа ће се попунити подацима из текстуалног фајла. Уколико се садржај фајла промени, дугачији ће бити и садржај падајуће листе.

```
public partial class Pocetna : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            StreamReader f=new StreamReader(Server.MapPath("recnik.txt"));
            List<Rec> recnik = Rec.FormirajIzFajla(f);
            f.Close();
            DropDownList1.DataSource = recnik;
            DropDownList1.DataTextField = "Eng";
            DropDownList1.DataValueField = "Srp";
            DropDownList1.DataBind();
        }
    }
    protected void Button1_Click(object sender, EventArgs e)
    {
        Label1.Text = DropDownList1.SelectedValue;
    }
}
```

Пример 7

Постоје контроле које служе за хијерархијско представљање података. Једна од тих контрола је и **контрола *Tree View***. Употребићемо је на веб-страници за приказ поделе угљеводоника.

Контрола *Tree View* има облик стабла и подаци се у њој налазе у чворовима (енгл. *Node*) који се појављују на сваком месту где има гранања, као и у листовима који се налазе на самом дну хијерархије. Контрола може да се попуни динамички подацима из неког текстуалног фајла, *XML* фајла или базе података, слично као што смо попунили падајућу листу у претходном примеру. У овом примеру ћемо,