



ASP.NET Cookies

Un **cookie** este o mica parte de text care insoteste cererea si paginile intre Servers si Browser. Cookie-ul contine informatia pe care aplicatia Web o poate citi de fiecare data cand un utilizator viziteaza aplicatia respectiva. Astfel, cookie-urile ofera un mijloc de stocare a informatiilor.

Cookie-urile sunt asociate cu intreg site-ul Web si nu cu paginile acestuia sau cu o anumita pagina din cadrul site-ului, astfel browserul si serverul vor schimba informatiile indiferent de pagina pe care utilizatorul o solicita. Daca utilizatorul viziteaza mai multe site-uri, fiecare dintre acestea trimite un cookie catre browserul utilizatorului, browserul stocand fiecare cookie separat.

Cand se pot utiliza Cookie-uri?

Cookie-urile ofera un mijloc de a stoca informatie specifica utilizatorilor, intr-o aplicatie Web.

De exemplu:

- In momentul in care un utilizator viziteaza un site sau o aplicatie, se pot utiliza cookie-uri pentru a stoca preferintele acelui utilizator sau alte informatii asemanatoare. In momentul in care utilizatorul viziteaza din nou aplicatia aceasta poate recupera informatia care a fost stocata

- Cookie-urile sunt utile pentru a mentine date despre utilizatori

- Pentru a mentine continuitatea intr-o aplicatie Web (ex: site de cumparaturi pentru a mentine evidenta cumparaturilor, pentru o aplicatie in care utilizatorii pot vota – un utilizator sa nu poata vota de mai multe ori din acelasi browser, vizualizarile de pe YouTube se contorizeaza in acelasi mod)

Scenariul

- Utilizatorul solicita o pagina de pe un site sau dintr-o aplicatie
- Aplicatia trimite pagina solicitata si un cookie care contine data si ora cand utilizatorul a accesat pagina
- Cookie-ul va fi stocat pe hard disk-ul utilizatorului
- Atunci cand utilizatorul acceseaza din nou pagina si introduce adresa URL in browser, se cauta cookie-ul asociat cu URL-ul pe hard disk
- Daca acest cookie exista, browserul il trimite in aplicatie
- In acest moment aplicatia poate determina data si ora cand utilizatorul s-a conectat ultima oara
- Se poate utiliza aceasta informatie si pentru a verifica daca o anumita data a expirat si astfel se poate trimite mesaj utilizatorului

Limitarile pe care le au Cookie-urile

Majoritatea browserelor accepta cookie-uri pana la 4096 bytes. Din cauza acestei limitari, ele trebuie sa stocheze cantitati mici de date sau cel mai bine ar putea stoca un identificator, cum ar fi id-ul utilizatorului. Id-ul unui utilizator poate fi astfel folosit pentru a identifica utilizatorul si pentru a citi date despre utilizator dintr-o baza de date sau dintr-o alta sursa de date.

De asemenea, browserele impun limitari referitoare la numarul de cookie-uri pe care site-ul le poate stoca. Majoritatea browserelor permit un numar de doar 20 de cookie-uri pe site. Daca se incearca stocarea mai multor cookie-uri, vechile cookie-uri vor fi eliminate. Alte browsere permit pana la 300 de cookie-uri.

O limitare ar fi atunci cand utilizatorul seteaza browserul sa refuze cookie-urile. Desi cookie-urile sunt foarte utile in cadrul unei aplicatii, aceasta nu trebuie sa depinda de posibilitatea de a stoca cookie-uri.

Scrierea Cookie-urilor

Browserul este responsabil pentru gestionarea cookie-urilor. Cookie-urile sunt trimise catre browser prin intermediul obiectului **HttpResponse**, care expune o colectie numita **Cookies**. Obiectul **HttpResponse** se poate accesa ca proprietate de raspuns (**Response**) a clasei **Page**.

Proprietatea **Page.Response** – obtine obiectul **HttpResponse** asociat clasei **Page**. Acest obiect permite trimiterea datelor de raspuns HTTP unui client si contine informatii despre raspunsul respectiv.

Urmatorul exemplu prezinta crearea unui cookie si adaugarea lui in output-ul HTTP al paginii utilizand obiectul **HttpResponse**.

```
HttpCookie MyCookie = new HttpCookie("LastVisit");
DateTime now = DateTime.Now;

MyCookie.Value = now.ToString();
MyCookie.Expires = now.AddHours(1);

Response.Cookies.Add(MyCookie);
```

- Utilizatorii pot sterge cookie-urile de pe computer in orice moment. De aceea, chiar daca cookie-urile sunt stocate intr-o aplicatie pe o perioada

mai mare de timp, un utilizator ar putea decide sa stearga toate cookie-urile, stergand astfel toate setarile care au fost stocate prin intermediul acestora.

Daca nu setati expirarea cookie-ului, cookie-ul este creat, dar nu este stocat pe hard disk-ul utilizatorului. In schimb, cookie-ul este mentinut ca parte a informatiilor de sesiune ale utilizatorului. Cand utilizatorul inchide browserul, cookie-ul este eliminat. Un astfel de cookie **non-persistent** este util pentru informatii care trebuie stocate doar pentru o perioada scurta de timp sau, din motive de securitate, nu ar trebui sa fie scrise pe disk pe computer. De exemplu, cookie-urile non-persistente sunt utile daca utilizatorul lucreaza pe un computer public, unde nu se doreste scrierea cookie-urilor pe disk.

Se pot observa mai jos doua metode de a scrie cookie-urile:

```
Response.Cookies["userName"].Value = "user";  
Response.Cookies["userName"].Expires = DateTime.Now.AddDays(1);
```

```
HttpCookie aCookie = new HttpCookie("lastVisit");  
aCookie.Value = DateTime.Now.ToString();  
aCookie.Expires = DateTime.Now.AddDays(1);  
Response.Cookies.Add(aCookie);
```

Citirea cookie-urilor

In momentul in care un client (browser) face un request la server acesta trimite in mod implicit cookie-urile care au fost setate anterior.

Intr-o aplicatie ASP.NET acestea pot fi citite folosind obiectul **HttpRequest**, care este disponibil prin intermediul proprietatii **Request** a clasei **Page**.

```
if(Request.Cookies["userName"] != null)  
{
```

```
        HttpCookie aCookie = Request.Cookies["userName"];
        Response.Write(aCookie.Value);
    }
```

Intotdeauna trebuie verificat daca un cookie exista, comparand cu **null** deoarece in momentul incercarii de a accesa un cookie neexistent se va arunca o exceptie de tipul **NullReferenceException**.

Modificarea si stergerea cookie-urilor

Modificarea unui cookie consta in crearea unui nou cookie care sa contina noile valori si trimiterea cookie-ului catre browser pentru a rescrie ultima versiune existenta in client.

Urmatorul exemplu arata modul in care se poate **modifica** un cookie care stocheaza numarul de vizite ale utilizatorilor pe site.

```
int counter;

if (Request.Cookies["counter"] == null)
    counter = 0;

else
{
    counter = int.Parse(Request.Cookies["counter"].Value);
}

counter++;

Response.Cookies["counter"].Value = counter.ToString();
Response.Cookies["counter"].Expires = DateTime.Now.AddDays(1);
```

Nu se poate **sterge** direct un cookie deoarece acesta este pe computerul utilizatorului

Stergerea cookie-urilor se face prin crearea unui nou cookie care sa aiba acelasi nume cu cel pe care dorim sa il stergem folosind o data de expirare in trecut, iar in momentul in care browserul verifica data de expirare, acesta o sa stearga in mod automat cookie-ul respectiv.

```
HttpCookie aCookie = Request.Cookies["userName"];  
aCookie.Expires = DateTime.Now.AddDays(-1);  
Response.Cookies.Add(aCookie);
```



ASP.NET Session

O sesiune ofera posibilitatea de a salva si citi valori asociate utilizatorilor, in momentul in care acestia navigheaza printre paginile unei aplicatii. Sesiunile asociaza utilizatorului un identificator unic pentru o perioada determinata de timp si ofera o modalitate prin care valorile variabilelor persista pe intreaga durata in care sesiunea este activa.

Variabilele de sesiune

Variabilele de sesiune sunt stocate intr-un obiect de tipul **SessionStateItemCollection** (reprezinta o colectie de obiecte stocate in sesiune) accesibil prin intermediul proprietatii **HttpContext.Session** (preia datele din sesiunea curenta). Variabilele sesiunii curente pot fi accesate prin intermediul proprietatii **Session** a obiectului Page.

Urmatorul exemplu arata cum se pot crea variabile de sesiune pentru numele si prenumele unui utilizator, informatii preluate din controale TextBox.

```
Session["FirstName"] = FirstNameTextBox.Text;  
Session["LastName"] = LastNameTextBox.Text;
```

In cadrul variabilelor de sesiune se pot stoca toate tipurile de date acceptate de framework-ul .NET. In exemplul urmator se stocheaza in variabila **items** o valoare de tipul ArrayList.

```
ArrayList items = (ArrayList)Session["items"];  
Session["items"] = items;
```

Sesiunile sunt identificate prin intermediul unui identificator unic care poate fi citit utilizand proprietatea **SessionID**. In cazul in care aceasta proprietate nu exista se alocă in mod automat o valoare noua si se creeaza o noua sesiune, asociata cu identificatorul creat.

Identificatorul de sesiune se salveaza in mod implicit intr-un cookie. Aceasta optiune poate fi modificata folosindu-se URL-ul in locul cookie-ului, de exemplu:

[http://www.example.com/\(S\(lit3py55t21z5v55v1m25s55\)\)/default.aspx](http://www.example.com/(S(lit3py55t21z5v55v1m25s55))/default.aspx)

Cookie-uri vs Sesiuni

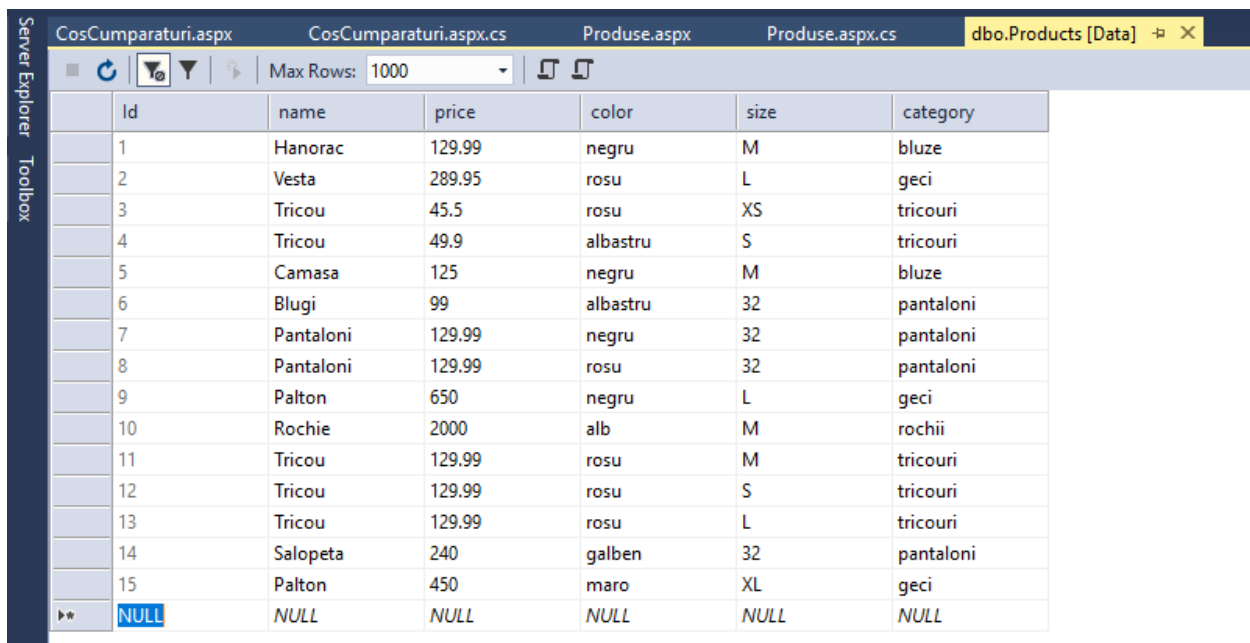
Cookie-urile si sesiunile sunt similare in ceea ce priveste modul de utilizare.

Cookie-urile pot fi persistente o lunga perioada de timp, ceea ce inseamna ca informatia din cookie-uri poate fi stocata pentru cateva luni sau chiar ani. Stocarea datelor se efectueaza in client, securitatea fiind mai scazuta deoarece exista riscul ca acestea sa fie modificate sau furate de un atacator.

Sesiunile sunt persistente pana in momentul in care utilizatorul inchide browserul. Acestea sunt stocate pe server, ducand la o securitate crescuta, comparativ cu cookie-urile, deoarece datele nu sunt vizibile utilizatorului si nu pot fi furate sau modificate. Datele din sesiune nu trebuie transmise la fiecare accesare de pagina deoarece ele pot fi cerute oricand prin intermediul id-ului de sesiune. In sesiune se pot stoca date fara sa existe o limitare din punctul de vedere al spatiului si al dimensiunii datelor.

Exemplu utilizand cookie-uri si sesiuni

Se considera urmatorul tabel care reprezinta produsele unui magazin online:



	Id	name	price	color	size	category
	1	Hanorac	129.99	negru	M	bluze
	2	Vesta	289.95	rosu	L	geci
	3	Tricou	45.5	rosu	XS	tricouri
	4	Tricou	49.9	albastru	S	tricouri
	5	Camasa	125	negru	M	bluze
	6	Blugi	99	albastru	32	pantaloni
	7	Pantaloni	129.99	negru	32	pantaloni
	8	Pantaloni	129.99	rosu	32	pantaloni
	9	Palton	650	negru	L	geci
	10	Rochie	2000	alb	M	rochii
	11	Tricou	129.99	rosu	M	tricouri
	12	Tricou	129.99	rosu	S	tricouri
	13	Tricou	129.99	rosu	L	tricouri
	14	Salopeta	240	galben	32	pantaloni
	15	Palton	450	maro	XL	geci
	NULL	NULL	NULL	NULL	NULL	NULL

Produsele magazinului sunt afisate impreuna cu detaliile, dupa cum urmeaza:

Produse

[Vezi cosul de cumparaturi](#)

Hanorac <ul style="list-style-type: none">• Pret: 129.99• Culoare: negru• Marime: M• Categorie: bluze Adauga in cos	Vesta <ul style="list-style-type: none">• Pret: 289.95• Culoare: rosu• Marime: L• Categorie: geci Adauga in cos	Tricou <ul style="list-style-type: none">• Pret: 45.5• Culoare: rosu• Marime: XS• Categorie: tricouri Adauga in cos	Tricou <ul style="list-style-type: none">• Pret: 49.9• Culoare: albastru• Marime: S• Categorie: tricouri Adauga in cos
Camasa <ul style="list-style-type: none">• Pret: 125• Culoare: negru• Marime: M• Categorie: bluze Adauga in cos	Blugi <ul style="list-style-type: none">• Pret: 99• Culoare: albastru• Marime: 32• Categorie: pantaloni Adauga in cos	Pantaloni <ul style="list-style-type: none">• Pret: 129.99• Culoare: negru• Marime: 32• Categorie: pantaloni Adauga in cos	Pantaloni <ul style="list-style-type: none">• Pret: 129.99• Culoare: rosu• Marime: 32• Categorie: pantaloni Adauga in cos

Afisarea cosului de cumparaturi:

[Inapoi la produse](#)

[Goleste cosul de cumparaturi](#)

Id	name	price	color	size	category
2	Vesta	289.95	rosu	L	geci
5	Camasa	125	negru	M	bluze

In ceea ce priveste functionalitatea, se considera urmatoarele cerinte:

- Pagina de produse trebuie sa afiseze lista produselor cu informatiile aferente si sa contina un buton prin care utilizatorul poate adauga fiecare produs in parte in cosul de cumparaturi. In momentul adaugarii unui produs in cosul de cumparaturi, sa se

afiseze un mesaj de confirmare in pagina.

- Pagina care reprezinta cosul de cumparaturi sa afiseze produsele pe care utilizatorii le-au adaugat. Tot in aceasta pagina sa existe posibilitatea de stergere a tuturor produselor din cosul de cumparaturi.

In pagina **Produse.aspx** se pot afisa produsele utilizand un control de tipul ListView in care se afiseaza datele acestora cat si un link de adaugare in cosul de cumparaturi.

```
<a href="/Produse.aspx?addToCart=<%# Eval("id") %>">Adauga in  
cos</a>
```

In pagina **Produse.aspx.cs** avem urmatoarea secventa de cod care reprezinta functionalitatea adaugarii unui produs in cosul de cumparaturi. Astfel, in cookie-uri salvam produsele si cantitatea.

De asemenea, mesajul de confirmare la adaugarea unui produs va fi stocat in sesiune.

```
// Verificam daca se doreste adaugarea unui produs in cosul de cumparaturi  
if(Request.Params["addToCart"] != null)  
{  
    string idProdus = Request.Params["addToCart"];  
  
    // Verificam daca exista cookie-ul care sa contina ID-urile produselor  
adaugate  
  
    if(Request.Cookies["cart"] == null)  
    {  
        // Cookie-ul pentru cosul de cumparaturi (cart) este gol -> adaugam prima  
valoare  
        Response.Cookies["cart"][idProdus] = "1";  
    }  
  
    else  
    {
```

```

        // Luam o colectie de cookie-uri existente
        NameValueCollection cartCookies;
        cartCookies = Request.Cookies["cart"].Values;

        // Iteram prin toate cheile cookie-ului "cart" (adica id-urile de
produse)
        foreach( string productId in cartCookies)
        {
            // Readaugam valoarea veche a cookie-ului in cookie-ul cart
            Response.Cookies["cart"][productId] = cartCookies[productId];
        }

        // Adaugam sau modificam noua valoare pentru idProdus

        if (Request.Cookies["cart"][idProdus] == null)
        {
            Response.Cookies["cart"][idProdus] = "1";
        }
        else
        {
            int count = int.Parse(Request.Cookies["cart"][idProdus]);
            Response.Cookies["cart"][idProdus] = (count + 1).ToString();
        }
    }

    Response.Cookies["cart"].Expires = DateTime.Now.AddDays(15);

    // Am adaugat cookie-ul, facem redirect la produse
    Session["message"] = "Produsul a fost adaugat in cosul de cumparaturi";

    Response.Redirect("/CosCumparaturi.aspx");
}

```

Pentru a afisa mesajul de succes stocat in sesiune (Session["message"] = "Produsul a fost adaugat in cosul de cumparaturi") se verifica daca exista mesajul salvat, se afiseaza, dupa care se sterge din sesiune deoarece nu se doreste afisarea permanenta a sa.

```

// Verificam daca exista un mesaj in sesiune
if (Session["message"] != null)
{
    // Afisam mesajul din sesiune
    Response.Write(Session["message"]);
}

```

```

        // Stergem mesajul din sesiune
        Session["message"] = null;
    }

```

Pentru pagina CosCumparaturi.aspx afisam produsele care au fost adaugate in cos.

In pagina CosCumparaturi.aspx.cs verificam daca exista elemente in cookie-ul cart, adica in cosul de cumparaturi. Daca exista, preluam cheile array-ului asociativ (Vezi imaginea de mai jos) si le unim prin virgula pentru a le folosi in clauza de select.

```

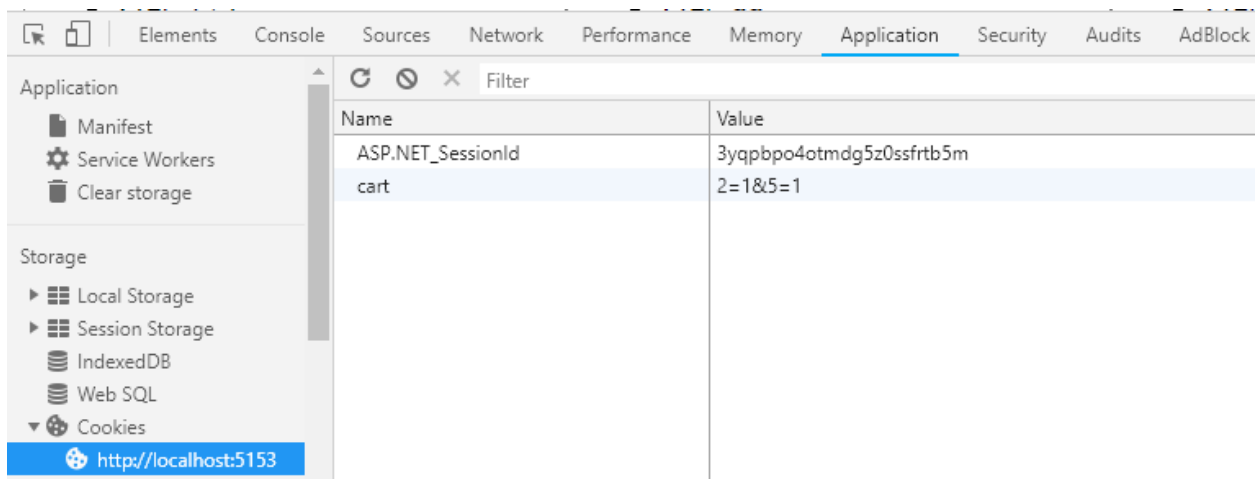
if (Request.Cookies["cart"] != null)
{
    NameValueCollection cartCookies;
    cartCookies = Request.Cookies["cart"].Values;

    string idProduce = string.Join(",", cartCookies.AllKeys);

    SqlDataSource1.SelectCommand = "SELECT * FROM products WHERE Id IN (" +
idProduce + ")";
}

else
{
    Response.Write("Cosul de cumparaturi este gol");
}

```



Stergerea tuturor produselor din cosul de cumparaturi se face prin apasarea unui buton sau accesarea unui link:

```
<a href="CosCumparaturi.aspx?remove=1">Goleste cosul de cumparaturi</a>
```

In partea de CS includem urmatoarea secventa de cod:

```
if (Request.Params["remove"] != null)
{
    if (Request.Params["remove"] == "1")
    {
        Response.Cookies["cart"].Expires = DateTime.Now.AddDays(-1);
        Session["message"] = "Cosul de cumparaturi a fost golit";
        Response.Redirect("/CosCumparaturi.aspx");
    }
}
```

XML

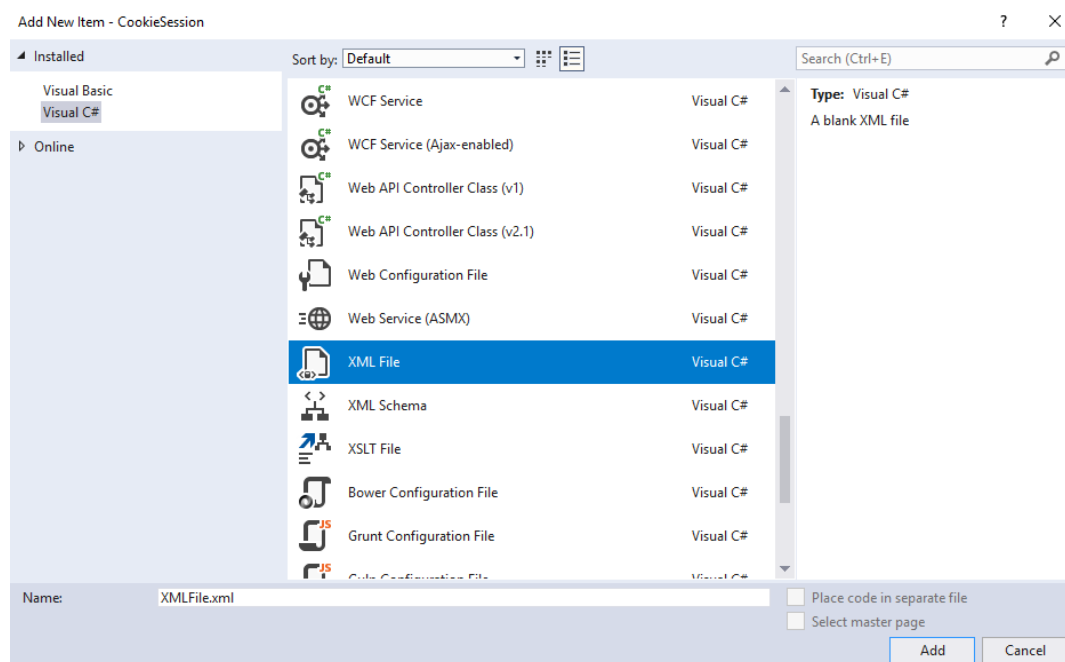


XML este un limbaj cross-platform, independent de hardware sau de software, care permite stocarea datelor intr-un mod structurat folosind tag-uri. Fisierele XML stocheaza date, similar cu o baza de date, datele fiind stocate in plain text.

In documentele XML se specifica structura datelor, folosind o schema XML. Aceasta schema trebuie sa fie valida, datele trebuie scrise sub forma arborescenta, iar toate tagurile trebuie inchise corespunzator.

Fisierele XML se pot folosi atat in ASP.NET ca sursa de date pentru elemente de tip ListView sau GridView, cat si in partea de CS unde pot fi citite si modificate.

Un fisier de tip XML se creeaza astfel:



Exemplu fisier XML:

```
<?xml version="1.0" encoding="utf-8"?>

<moviestore>

  <movie ID="1" genre="Fantasy, Action" year="2015" duration="120">
    <title>Star Wars: The Force Awakens</title>
    <director>J.J. Abrams</director>
    <actor>Harrison Ford</actor>
    <actor>Mark Hamill</actor>
    <actor>Carrie Fisher</actor>
  </movie>

  <movie ID="2" genre="Thriller" year="2015" duration="148">
    <title>Spectre</title>
    <director>Sam Mendes</director>
    <actor>Daniel Craig</actor>
  </movie>

  <movie ID="3" genre="Sci-Fi" year="2015" duration="144">
    <title>The Martian</title>
    <director>Ridley Scott</director>
    <actor>Matt Damon</actor>
  </movie>

  <movie ID="4" genre="Sci-Fi" year="2014" duration="169">
    <title>Interstellar</title>
    <director>Christopher Nolan</director>
    <actor>Matthew McConaughey</actor>
    <actor>Anne Hathaway</actor>
  </movie>

  <movie ID="5" genre="Comedy" year="2015" duration="91">
    <title>Minions</title>
    <director>Kyle Balda</director>
    <actor>Sandra Bullock</actor>
    <actor>Jon Hamm</actor>
    <actor>Michael Keaton</actor>
  </movie>

  <movie genre="Western" year="2015" duration="168" ID="6">
    <title>The Hateful Eight</title>
    <director>Quentin Tarantino </director>
    <actor>Jennifer Jason Leigh </actor>
    <actor>Channing Tatum</actor>
    <actor>Samuel L. Jackson</actor>
  </movie>

</moviestore>
```

Metode de procesare a unui document XML:

`XmlDocument doc = new XmlDocument();` - creare unui document nou

`XmlElement movie = doc.CreateElement("movie");` - crearea unui element

`movie.SetAttribute("genre", IDTextBox.Text);` - setare atributelor pe taguri

`XmlElement root = doc.DocumentElement;` - crearea unui element de tip root

`XmlNode lastMovie = root.LastChild;` - gasirea ultimului element din ierarhie

`movie.AppendChild(ChildTitle);` - adaugarea unui element (child)

`root.AppendChild(movie);` - adaugarea unui element subordonat radacinii

`doc.Save(Server.MapPath("~/") + "/Movies.xml");` - salvarea documentului XML

