

Web-aplikacije

ASP.NET Core MVC

2018/19.10-b

Padajuće liste za odabir povezanih vrijednosti

Dinamičko brisanje podatka

Slanje datoteke na server. Prikaz slike artikla.

Primjer unosa objekta s ograničenim skupom vrijednosti za neko svojstvo

2

- Mjesto ima strani ključ na tablicu Država
- Umjesto unosa šifre države omogućiti korisniku da odabere državu iz popisa država.
- Popis država nije dio modela, već se prenosi koristeći *ViewBag* ili *ViewData*

Države

Mjesta

Artikli

Partneri

Dokumenti

Unos novog mjesta

Poštanski broj mjesta

Naziv mjesta

Poštanski naziv mjesta

Država

Odaberite državu

- Bahamas
- Bahrain
- Bangladesh
- Barbados
- Belarus
- Belgium
- Belize
- Benin
- Bermuda
- Bhutan
- Bolivia
- Bosnia and Herzegovina**
- Botswana

Priprema podataka za padajuću listu

3

➤ Na osnovu liste država stvara se objekt tipa *SelectList*

➤ Navodi se izvor podataka, svojstvo koje predstavlja vrijednost odabranog elementa i svojstvo koje se prikazuje kao tekst u padajućoj listi

➤ Stvoreni objekt se pogledu prenosi koristeći ViewBag (ili ViewData)

➤ Može se upotrijebiti proizvoljno ime za novu vrijednost u ViewBagu

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ MjestoController.cs


```
[HttpGet]
public IActionResult Create() {
    PrepareDropDownLists();
    return View();
}

private void PrepareDropDownLists() {
    var drzave = ctx.Drzava.AsNoTracking().OrderBy(d => d.NazDrzave)
        .Select(d => new { d.NazDrzave, d.OznDrzave })
        .ToList();

    ViewBag.Drzave = new SelectList(drzave,
        nameof(Drzava.OznDrzave), nameof(Drzava.NazDrzave)) ;
}
```

Prikaz padajuće liste u pogledu

4

- Padajuća lista se koristi unutar HTML oznake *select*, pri čemu se izvor navodi atributom *asp-items*
 - Potrebno i navesti *asp-for* za određivanje svojstva kojem će se odabir pridružiti
- Moguće umetnuti i dodatne elemente u padajuću listu (osim onih koji su već pripremljeni).
 - Dodaju se na početak
 - Npr. poruka da se odabere neki element iz liste koja je inicijalno odabrana
 - Nakon što se jednom promijeni vrijednost (zbog atributa *disabled*) više se neće moći ponovo odabrati element s porukom
- Primjer:  Web \ Firma.Mvc \ Views \ Mjesto \ Create.cshtml

```
<select class="form-control" asp-for="OznDrzave"
        asp-items="ViewBag.Drzave">
    <option disabled selected value="">
        Odaberite državu
    </option>
</select>
```

Ponovna priprema podataka za padajuću listu

5


- U slučaju neispravnih ili nepotpunih podataka potrebno je ponovno prikazati pogled za unos, ali i pripremiti podatke za padajuću listu

➤ Primjer:  Firma.Mvc \ Controllers \ MjestoController.cs

```
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Create(Mjesto mjesto) {
    if (ModelState.IsValid) {
        try {
            ...
        }
        catch (Exception exc) {
            ...
            PrepareDropDownLists();
            return View(mjesto);
        }
    }
    else {
        PrepareDropDownLists();
        return View(mjesto);
    }
}
```

Model za pregled svih mjesta

6

- Umjesto entiteta Mjesto za pojedinačno mjesto koristi se novi prezentacijski pogled
 - Umjesto oznake države sadrži naziv države
 - Sadrži informacije o trenutnoj stranici i načinu sortiranja
- Primjer:  Web \ Firma.Mvc \ ViewModels \ MjestoViewModel.cs


```
public class MjestoViewModel {  
    public IEnumerable<MjestoViewModel> Mjesta { get; set; }  
    public PagingInfo PagingInfo { get; set; }  
}
```

- Primjer:  Web \ Firma.Mvc \ ViewModels \ MjestoViewModel.cs

```
public class MjestoViewModel {  
    public int IdMjesta { get; set; }  
    public int PostBrojMjesta { get; set; }  
    public string NazivMjesta { get; set; }  
    public string PostNazivMjesta { get; set; }  
    public string NazivDrzave { get; set; }  
}
```

Pregled svih mjesta


7

- Kao i u primjeru s državama zaglavlje tablice sadrži poveznice za sortiranje po određenom stupcu u prikazu
 - Izvedeno petljom i poljima koja sadrže nazive stupaca i parametre sortiranja
 - Primjer:  Web \ Firma.Mvc \ Views \ Mjesto \ Index.cshtml

```
<table class="table table-striped"> <thead>
  <tr>
    @{
      string[] nazivi = { "Poštanski broj", "Naziv mjesta",
                          "Poštanski naziv mjesta", "Država" };
      for (int i = 1; i <= nazivi.Length; i++) {
        <th>
          <a asp-route-sort="@i" asp-route-page="@Model.PagingInfo.CurrentPage"
            asp-route-ascending="@ (Model.PagingInfo.Sort == i ?
              Model.PagingInfo.Ascending? false : true : true)">
            @nazivi[i - 1]
          </a>
        </th>
      }
    }
```

Parcijalni pogled za prikaz mjesta (1)

8

- Prikaz pojedinog retka izveden parcijalnim pogledom
 - Parcijalni pogled je pogled koji ne koristi glavnu stranicu te se uključuje u neke druge poglede
 - Može imati svoj model (u prikazanom primjeru je to pojedinačno mjesto)
 - Prima kopiju podataka iz ViewData/ViewBag pozivatelja (kopija se može nadopuniti novim podacima)
- U pogledu *Index* poziva se parcijalni pogled *Row* te se njegov sadržaj (tj. rezultat izvršavanja) umetne u konačni rezultat
 - Poziv se obavlja korištenjem *tag-helpera partial*
 - Primjer:  Web \ Firma.Mvc \ Views \ Mjesto \ Index.cshtml

```
<table class="table table-striped">
...
@{ ViewDataDictionary vdd = new ViewDataDictionary(this.ViewData);
    vdd.Add("PagingInfo", Model.PagingInfo); }
@foreach (var mjesto in Model.Mjesta) {
    <partial name="Row" model="mjesto" view-data="vdd" />
}...
```


Parcijalni pogled za prikaz mjesta (2)

9


► Primjer:  Web \ Firma.Mvc \ Views \ Mjesto \ Row.cshtml

► Model je tipa MjestoViewModel (sadrži naziv umjesto oznake države)

```
@model MjestoViewModel
<tr>
  <td class="text-left">@Model.PostBrojMjesta</td>
  <td class="text-left">@Model.NazivMjesta</td>
  <td class="text-left">@Model.PostNazivMjesta</td>
  <td class="text-left">@Model.NazivDrzave</td>
  <td>
    <a asp-action="Edit"
      asp-route-page="@ViewBag.PagingInfo.CurrentPage"
      asp-route-sort="@(((PagingInfo) ViewData["PagingInfo"]).Sort)"
      asp-route-ascending="@ViewBag.PagingInfo.Ascending"
      asp-route-id="@Model.IdMjesta"
      class="btn btn-sm" title="Ažuriraj"><i class="fas fa-edit"></i></a>
  </td>
  <td>
    ...
  </td>
</tr>
```

„Poveznica” za brisanje podatka bez osvježavanja cijele stranice

10

- Za razliku od brisanje države, u primjeru s mjestima ne dolazi do osvježavanje cijele stranice
 - Umjesto slanja zahtjeva za novom stranicom, koristit će se jQuery Ajax poziv
 - Posljedično ne treba postojati skriveno polje s identifikatorom mjesta
 - Zbog AntiforgeryToken potrebno dodati HTML oznaku *form* u pogled iako se neće koristiti
 - Podatak u mjestu koje treba obrisati postavljen kao atribut na gumbu Obriši
 - Primjer:  Web \ Firma.Mvc \ Views \ Mjesto \ Row.cshtml

```
<tr>
    ...
    <td>
        @*Neće se koristiti submit, pa ne treba hidden polje (ako skripta ispravno
        radi), ali form treba zbog antiforgery tokena*@
        <form asp-action="Delete" method="post">
            <input type="hidden" name="id" value="@Model.IdMjesta" />
            <button data-idmjesta="@Model.IdMjesta"
                class="btn btn-sm btn-danger deleteajax" title="Obriši">
                <i class="fas fa-trash-alt"></i></button>
        </form>
```

Brisanje retka bez osvježavana stranice (2)

11


- Brisanje obavlja vlastiti programski kod pisan u jQueryju
 - Nakon učitavanja stranice svakoj kontroli koja ima postavljen stil *deleteajax* pridružuje se kod kojim se klikom na tu kontrolu izvršava brisanje mjesta
 - Kôd zapisan u funkciji *SetDeleteAjax*
 - Funkcija parametrizirana akcijom koju treba izvršiti i nazivom parametra po kojem se prepoznaje redak u tablici
 - U ovom primjeru to je *idmjesta* te je akcija *Delete*
 - Umjesto pisanje url-a, koristi se postupak *Url.Action* koji vrati poveznicu na akciju na nekom (ili istom upravljaču)

➤ Primjer:  Web \ Firma.Mvc \ Views \ Mjesto \ Index.cshtml

```
@section scripts{
  <script type="text/javascript">
    $(function () {
      ...
      SetDeleteAjax(".deleteajax", '@Url.Action("Delete")', 'idmjesta');
    });
  </script>
}
```

Brisanje retka bez osvježavana stranice (3)


12

- Klikom na kontrolu uklanja se prethodna statusna poruka i dohvaćaju podaci o podatku koji treba obrisati
 - *selector* predstavlja informaciju kako će jQuery doći do elementa
 - *paramname* označava u kojem data atributu je identifikator podatka
- Primjer:  Web \ Firma.Mvc \ wwwroot \ js \ site.js

```
function SetDeleteAjax(selector, url, paramname) {  
    $(document).on('click', selector, function (event) {  
        event.preventDefault(); //bitno ako je submit button  
        var paramval = $(this).data(paramname);  
        var tr = $(this).parents("tr");  
        ...  
        if (confirm('Obrisati zapis?')) {  
            var token = $('input[name="__RequestVerificationToken"]').first().val();  
            $("#tempmessage").siblings().remove();  
            $("#tempmessage").removeClass("alert-success");  
            $("#tempmessage").removeClass("alert-danger");  
            $("#tempmessage").html('');  
            ...  
        }  
    });  
}
```

Brisanje retka bez osvježavana stranice (4)

13

- Skripta poziva odgovarajući url s identifikatorom podatka i anti-forgery tokenom
 - Ispisuje se poruka o uspješnosti brisanja
 - successful i message su vlastite varijable u rezultatu (vidi sljedeći slajd)
 - U slučaju uspješnog brisanja uklanja se redak sa zapisom
 - U slučaju pogreške prilikom zahtjeva prikazuje se dijalog s opisom pogreška
 - Primjer:  Web \ Firma.Mvc \ wwwroot \ js \ site.js

```
function SetDeleteAjax(selector, url, paramname) {  
    ...  
    $.post(url, { id: paramval, __RequestVerificationToken: token },  
        function (data) {  
            if (data.successful)  
                $(tr).remove();  
            $("#tempmessage").addClass(data.successful ?  
                "alert-success" : "alert-danger");  
            $("#tempmessage").html(data.message);  
        }).fail(function (jqXHR) {  
            alert(jqXHR.status + " : " + jqXHR.responseText);  
        });  
}
```

Akcija brisanja mjesta

14


➡ Postupak vraća anonimni razred pretvoren u JSON

➡ Primjer:  Web \ Firma.Mvc \ Controllers \ MjestoController.cs

```
[HttpPost]
[ValidateAntiForgeryToken]
public IActionResult Delete(int id) {
    var mjesto = ctx.Mjesto.AsNoTracking()
        .Where(m => m.IdMjesta == id)
        .SingleOrDefault();
    if (mjesto != null) {
        try {
            string naziv = mjesto.NazMjesta;
            ctx.Remove(mjesto); ctx.SaveChanges();
            var result = new {
                message = $"Mjesto {naziv} sa šifrom {id} obrisano.",
                successful = true };
            return Json(result);
        }
        catch (Exception exc) {
            var result = new {
                message = "Pogreška prilikom brisanja mjesta: " ...
```

Forma za slanje datoteke


15

- Prilikom unosa novog artikla može se poslati datoteka sa slikom artikla
- Za unos se koristi HTML *input* kontrola tipa *file*
 - Naziv može biti proizvoljan, ali mora odgovarati argumentu u akciji upravljača
- Forma mora imati atribut *enctype* postavljen na *multipart/form-data*
- Primjer:  Web \ Firma.Mvc \ Views \ Artikl \ Create.cshtml

```
<form asp-action="Create" method="post" enctype="multipart/form-data">
...
<label asp-for="SlikaArtikla" class="col-sm-2 col-form-label"></label>
  <div class="col-sm-5">
    <input type="file" name="slika" />
  </div>
...
<button class="btn btn-primary" type="submit">Dodaj</button>
```

Prihvat datoteke na upravljaču


16

- Postupak prima artikl stvoren na osnovu podataka iz forme i argument tipa *IFormFile*
 - Naziv argumenta odgovara vrijednosti atributa *name* iz kontrole za unos u pogledu
 - Ako je korisnik odabrao datoteku, njen sadržaj se može kopirati u MemoryStream nakon čega se može dobiti polje bajtova i pospremiti u entitet, odnosno u bazu podataka
- Primjer:  Web \ Firma.Mvc \ Controllers \ ArtiklController.cs

```
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create(Artikl artikl, IFormFile slika) {
    ...
    if (ModelState.IsValid) {
        if (slika != null && slika.Length > 0) {
            using (MemoryStream stream = new MemoryStream()) {
                await slika.CopyToAsync(stream);
                artikl.SlikaArtikla = stream.ToArray();
            }
        }
        ctx.Add(artikl);
        ctx.SaveChanges();
    }
}
```


Validacija za vrijednosti s decimalnim zareзом/decimalnom točkom

17

- ➡ jQuery validacija radi s decimalnom točkom, što nije u skladu s našim postavkama
- ➡ Potrebno promijeniti funkcije za provjeru raspona i provjeru je li neka vrijednost brojčana vrijednost
 - ➡ Primjer:  Web \ wwwroot \ js \ validation.js

```
$.validator.methods.range = function (value, element, param) {  
    var globalizedValue = value.replace(",", ".", ".");  
    return this.optional(element)  
        ||  
        (globalizedValue >= param[0] && globalizedValue <= param[1]);  
}  
  
$.validator.methods.number = function (value, element) {  
    return this.optional(element)  
        ||  
        /^-?(?:\d+|\d{1,3}(?:[\s\.,]\d{3})+)(?:[\s\.,]\d+)?$/ .test(value);  
}
```

Prikaz svih artikala (1)

18

- ➡ Prikaz pojedinačnog artikla izveden u parcijalnom pogledu Row.cshtml koji se uključuje u Index.cshtml za svaki artikl

➡ Primjer:  Web \ Firma.Mvc \ Views \ Artikl \ Index.cshtml

```
@foreach (var artikl in Model.Artikli) {  
    <partial name="Row" model="artikl" />  
}
```

- ➡ Model za pojedinačni artikl je razred ArtikelViewModel

➡ Primjer:  Web \ Firma.Mvc \ ViewModels \ ArtikelViewModel.cs

```
public class ArtikelViewModel {  
    public int SifraArtikla { get; set; }  
    public string NazivArtikla { get; set; }  
    public string JedinicaMjere { get; set; }  
    public decimal CijenaArtikla { get; set; }  
    public bool Usluga { get; set; }  
    public string TekstArtikla { get; set; }  
    public bool ImaSliku { get; set; }  
    public int? ImageHash { get; set; }  
}
```

Prikaz svih artikala (2)


19

- Koristi se posebni razred umjesto razreda Artikal iz EF modela kako se ne bi prilikom dohvata svih artikala istovremeno preuzimale i slike svih artikala
 - Umjesto sadržaja slike, evidentira se postoji li slika te veličina slike te će taj broj biti dodan na adresu
 - Uz pretpostavku da se veličina slike mijenja ako se promijeni slika artikla preglednik će osvježiti sliku.
 - U stvarnosti bi trebalo koristiti hashcode slike
- Primjer:  Web \ Firma.Mvc \ Controllers \ ArtikalController.cs

```
var artikli = ctx.Artikal.Select(a => new ArtikalViewModel {  
    SifraArtikla = a.SifArtikla,  
    NazivArtikla = a.NazArtikla,  
    JedinicaMjere = a.JedMjere,  
    CijenaArtikla = a.CijArtikla,  
    Usluga = a.ZastUsluga,  
    TekstArtikla = a.TekstArtikla,  
    ImaSliku = a.SlikaArtikla != null,  
    ImageHash = a.SlikaChecksum  
})  
...
```

Poveznica za prikaz slike

20

- Ako artikl koji se prikazuje u pojedinom retku ima sliku, stvara se HTML img kontrola
- Adresa slike je akcije *GetImage* na upravljaču *Artikl*
 - Adresi slike se dodaj parameter hash kako bi preglednik mogao prepoznati novu sliku artikla u odnosu na onu koju ima spremljenu u svojoj memoriji
- Primjer:  Web \ Firma.Mvc \ Views \ Artikl \ Row.cshtml

```
@if (Model.ImaSliku) {  
      
}
```

Akcija za dohvat slike

21

- Polje bajtova koje predstavlja sliku artikla dohvati se EF upitom
- Rezultat postupka je *FileContentResult* koji nastane pozivom naslijeđenog postupka *File* iz upravljača.

➤ Primjer:  Web \ Firma.Mvc \ Controllers \ ArtiklController.cs

```
public FileContentResult GetImage(int id) {  
    byte[] image = ctx.Artikl  
        .Where(a => a.SifArtikla == id)  
        .Select(a => a.SlikaArtikla)  
        .SingleOrDefault();  
  
    if (image != null)  
        return File(image, "image/jpeg");  
    else  
        return null;  
}
```

➤ Napomene

- Povratna vrijednost je mogla biti i *ActionResult*, pa je umjesto `return null` moglo pisati `return NotFound()`
- Metoda vraća cijelu sliku što može biti nepraktično/sporo ako se traži samo minijatura. U tom slučaju potrebno je smanjiti sliku nekim od adekvatnih alata (npr. *LibVips*)