MVC 5 (Model-View-Controller)

Klijentsko programiranje

Kreiranje client-side skripte

- Pisanje inline client-side skripte na stranici:

```
<script language="javascript" type="text/javascript">
// Your script goes here
</script>
```

- Dodavanje skripte kao odvojenog resursa:

```
<head>
    <title></title>
    <script src="Site.js" type="text/javascript">
    </script>
</head>
```

Uvod u jQuery biblioteku

- jQuery je JavaScript biblioteka koja je podržana unutar Visual Studio radne okoline
- jQuery Selector sintaksa
  - Rad sa DOM elementima

```
$("#display").css("background-color", "yellow");
```

- jQuery događaja (events)
  - Lako okidanje DOM događaja (events)

```
<input type="button" id="btn1" value="Greetings" />
<script type="text/JavaScript">
    $(document).ready(function () {
        $("#btn1").click(function () {alert("hello world");
        });
    });
</script>
```

Asinkrono dohvaćanje podataka

- Služi za dohvaćanje podataka bez ponovnog učitavanja stranice
- Tehnologija koja stoji iza ovakvog načina rada zove se – Ajax
- SPA (Single Page Application) – cijela aplikacija je dostupna na samo jednoj stranici, bez ponovnih učitavanja, uz pomoć Ajax tehnologije
- Dobivamo na brzini i boljem korisničkom iskustvu

Svrha Ajax-a

- Ajax omogućuje pozivanje asinkronih poziva, što rezultira u bržem odgovoru sa servera.
- Ajax zahtjev (request) je pokrenut u pozadini, što ne zahtjeva ponovo učitavanje stranice.
- Korištenjem Ajax-a, možemo ažurirati dio stranice (parcijalno renderiranje).
- Ajax nam pomaže da unesemo dah desktop aplikacija na web.

Za i protiv korištenja Ajax-a

Za	Protiv
Ajax zahtjev se okida u pozadini, čime se izbjegava potpuni refresh stranice.	Ajax je JavaScript-dependent, što znači da korisnik koji ima onemogućen JavaScript neće moći koristiti Ajax.
Ajax efektivno prenosi malu količinu podataka do servera uz nesmetanu interakciju sa korisnikom.	Kod korištenja Back buttona moguća su neočekivana ponašanja preglednika.

Prednosti korištenja Ajax biblioteke na MVC stranicama

- Performanse – osigurava odlične performanse kod aplikacija koje su podatkovno orjentirane
- Interoperabilnost – tehnologija je neovisna o serverskoj platformi, kao i jQuery
- Proširivost – lako je proširiti biblioteku sa dodatnim funkcionalnostima

Kako koristiti Microsoft Ajax na MVC stranicama

- Potrebno je referencirati Ajax biblioteke
- Dostupna je MVC Ajax helper funkcija - Ajax.ActionLink
- Dostupna je MVC Ajax helper funkcija - Ajax.BeginForm

Implementacija Ajax-a unutar Asp.Net MVC stranica korištenjem jQuery-ja

- jQuery.load metoda
- jQuery.get metoda
- jQuery.post metoda
- jQuery.ajax metoda

jQuery.load metoda

- Sintaksa:

```
element.load(url, [data],
    [callback(response, status, XMLHttpRequest)]
```

- Poziva server i dohvaća HTML na stranicu.

```
$("#returnValue").load(
    "localhost/loadtest.htm #container");
```

jQuery.get metoda

- jQuery.get metoda omogućuje GET zahtjev na server.
- jQuery.get ima parametre, URL, objekt s podacima koji se prosljeđuje sa zahtjevom, i tzv. callback funkciju koja se izvršava nakon što se vrati rezultat:

```
$.get("localhost/employeedetails.html",
            { employeeId: 123, name: "Jeffrey" },
            function(data){
                alert("Employee details retrieved: " + data);
            });
```

jQuery.post metoda

- POST metoda se obično koristi kada zahtjev mijenja podatke na serveru.
- POST zahtjevi se nikada ne skladište, kao što je slučaj sa GET zahtjevima

```
$.post("localhost/detaljizaposlenika.html",
    { ime: 'Marko', prezime: 'Marić'},
        function(data) {
            alert("Dohvaćeni podaci o zaposleniku: " + data);
});
```

jQuery.ajax metoda

- Low level metoda koju koriste sve druge Ajax metode jQuery-ja.
- Prima cijeli skup parametara, od kojih su samo neki; url, type, data, success.
- Postoji i error parametar koji definira funkciju koja se izvršava u slučaju greške.

Rad sa jQuery i Ajax događajima (events)

- jQuery.ajaxSend Event Handler
- jQuery.ajaxComplete Event Handler
- jQuery.ajaxError Event Handler

Sintaksa jQuery Ajaxa

```
$.ajax({
        url: '...',
        type: '...',
        data: { var1: test,1 var2: test2, ...},
        beforeSend: function () {
            //kod
        },
        success: function (result) {
          //kod
        },
        error: function ( jqXHR, textStatus, errorThrown) {
          //kod
        },
        complete: function () {
          //kod
        }
    });
```

Primjer ajax poziva

```
$.ajax({
        url: '/Predavaci/Lista',
        type: 'GET',
        traditional: true,
        data: { arr: test, someBool: true, someInt: 1, someString: '', id: $(this).attr('id') },
        beforeSend: function () {
            alert("Prije slanja!");
        },
        success: function (result) {
            $("#listaPredavaca").html('');
            result.forEach(function (entry) {
                $("#listaPredavaca").append("<p>" + entry.Ime + " " + entry.Prezime + "</p>");
            });
        },
        error: function ( jqXHR, textStatus, errorThrown) {
            alert(jqXHR.responseText);
        },
        complete: function () {
            alert(„Gotovo!");
        }
    });
```

Like primjer (I)

- Ako želimo unutar naše aplikacije Enciklopedija, implementirati mogućnost asinkronog povećanja broja glasova određene pjesme, to ćemo učiniti na slijedeći način ...

Like primjer (II)

- Slijedeći kod treba staviti na View Details.cshtml koji je namijenjen za prikaz detalja jedne pjesme:
  - Na vrhu stranice ispod naslova:
    @Html.HiddenFor(model => model.ID)
  - Pored linkova na dnu stranice:

    Like

    ...

Like primjer (III)

- Unutar sekcije scripts, na dnu stranice:

```
@section scripts{
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script type="text/javascript">
   $(document).ready(function () {
      $("#like-button").click(function () {
         $.ajax({
            url: '/Glazba/Pjesmas/UvecajGlas',
            type: 'POST',
            traditional: true,
            data: { id: $("#ID").val() },
            beforeSend: function () {
               alert("Prije slanja!");
            },
            success: function (result) {
               alert("Uspješno izvršeno!");
            },
            error: function (jqXHR, textStatus, errorThrown) {
               alert(jqXHR.responseText);
            },
            complete: function () {
               alert("Gotovo!");
            }
         });
      });
   });
</script>
}
```

Like primjer (IV)

- Kod unutar Pjesmas Controllera:

```
[HttpPost]
public ActionResult UvecajGlas(int? id)
{
    var pjesma = db.Pjesmas.Where(p => p.ID == id).FirstOrDefault();
    pjesma.Glasovi = pjesma.Glasovi + 1;
    db.SaveChanges();

    return Json("OK");
}
```

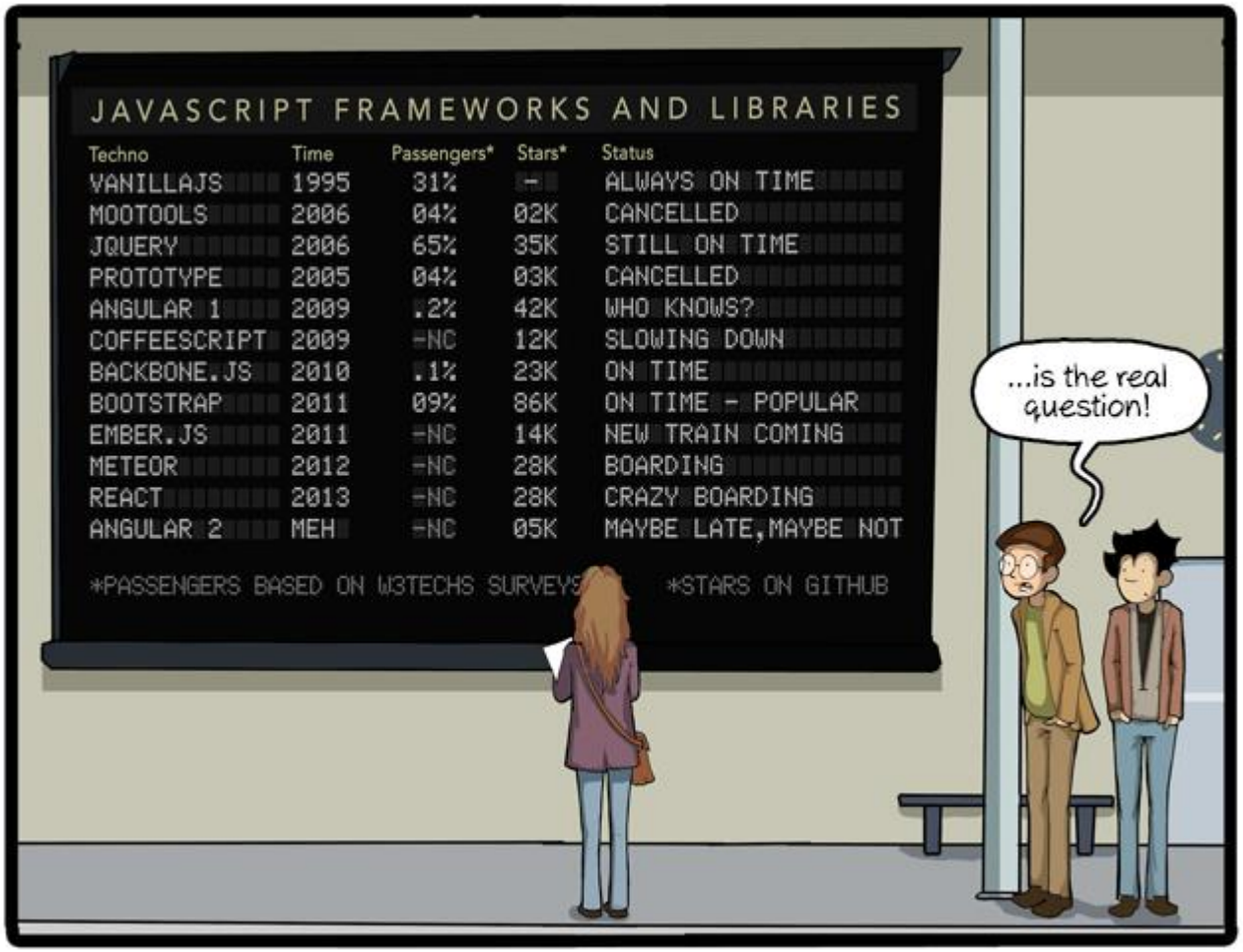
Like primjer (V)

- Pokrenite i provjerite ispravno funkcioniranje aplikacije
- Istu stvar možete implementirati za izvođače
- Ako vam smetaju poruke koje skaču tijekom izvršavanja, zakomentirajte sve alert-eve u klijentskom kodu

Dodatak!

- Postoje razne JavaScript biblioteke, koje se pojavljuju na tržištu vrlo velikom brzinom
- Neke od biblioteka su specializirane, dok druge nastoje osigurati kompaktnu razvojnu okolinu za razvoj cijelih web aplikacija
- Koje od njih koristiti je ponekad teško pitanje ...


The challenge when choosing a JS framework is to jump onto the right train at the right moment
If you jump too soon, you don't know where you'll end up...
But if you wait too long, you'll get left behind...
Hey, wait!
And how do you know which is the right train?
That, my friend...


JAVASCRIPT FRAMEWORKS AND LIBRARIES
Techno Time Passengers* Stars* Status
VANILLAJS 1995 31% - ALWAYS ON TIME
MOOTOOLS 2006 04% 02K CANCELLED
JQUERY 2006 65% 35K STILL ON TIME
PROTOTYPE 2005 04% 03K CANCELLED
ANGULAR 1 2009 .2% 42K WHO KNOWS?
COFFEESCRIPT 2009 =NC 12K SLOWING DOWN
BACKBONE.JS 2010 .1% 23K ON TIME
BOOTSTRAP 2011 09% 86K ON TIME - POPULAR
EMBER.JS 2011 =NC 14K NEW TRAIN COMING
METEOR 2012 =NC 28K BOARDING
REACT 2013 =NC 28K CRAZY BOARDING
ANGULAR 2 MEH =NC 05K MAYBE LATE,MAYBE NOT
*PASSENGERS BASED ON W3TECHS SURVEYS
*STARS ON GITHUB
...is the real question!