

ASP-MVC3

שוו הוא מתודולוגית פיתוח עבור אפליקציית web.

מושגים בסיסיים:

דף סטטי - הציגת דף html בסיסי כמוות זהה.

דף דינامي - אפשרות להציג דף html הנוצר בזמן בבקשת המשתמש ולפי בקשתו.

web servise - שירות שאפשר לקרוא לו בצורה חיצונית, לדוג' מכתובת url – וידוע להחזיר נתונים, או לחייב דפי html.

אפליקציה web application - תכיל בתוכה service web- להם נקרא ע"מ ליצור דף html לפי בבקשת המשתמש (ז"א דף דינامي), דף זה נציג למשתמש.

MVC וחלוקה

C – class ששמו מסתיים בـ "controller" והוא יורש ממחלקה הבסיסית controller – מוגדר כ- class שככל הפונקציות בתחום הון web service – ויש לנו גישה אליהם ממיקומות חיצוניים, הפונקציות יקראו "action"ים ויכלן להחזיר דף html או כל נתון אחר.

V – זה דף ה- html שהוא יחזיר לנו ה- action הנמצא ב- controller מסוים, בד"כ משתמש בדף מסווג cshtml – ע"מ שנוכל לשלב קוד server על הדף.

M – הוא class פשוט שמכיל את ה- data. אותו נעביר מה-action ל- view כדי שנוכל להציג את הנתונים בדף ה- html.

צעדים ראשוניים:

פתיחת פרויקט:

empty ← mvc3 ← web ← C# ← new project
פתיחה controller – לחיצה ימנית על התיקיה .add controller

שים לב: יש לסייע את השם עם המילה controller – אבל בשימוש גטייחס רק לשם ללא **מילה זו.**

הוספה action – בד"כ עם פתיחת controller חדש נוצר לנו action בשם index ואפשר להוסיף עוד פונקציות (actions).

כל פונקציה הנוצרת בתוך controller היא בעצם web service ונכנה אותה-action.

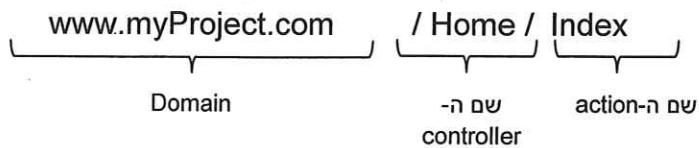
action אמרו להציג view (דף html) את שם ה- view יש לרשום באותו ה- ()
של ה- action.

ב- default ה- action יציג את ה- view שבו הוא שם ה- action ונמצא תחת תיקיה
שםה הוא שם ה- controller הנמצא מתחת תיקיה ה- views.

יצירת view – ע"מ ליצור view בהרכיה הנכונה אפשר ורצוי ללחוץ לחיצה ימנית על שם
ה- action ולבחר view – אוטומטית יוצר תחת התקיה בשם action.html
דף html בשם ה- action.

הקריאה ל- webservice בטכנולוגיית MVC:

יש לקרוא ל- domain (שם השירות עליו יושב הפרויקט או כרגע ... :localhost) אשר יכיל את controller ולח"כ מכון שם ה- action לדוג'.



ע"מ להגדיר איזה action יקרה כברירת מחדל מיד בעת הקריאה בשם הדומיין, יש לציין בקובץ ה- Global ב- routes את ה- Action Controller וה- routes הרצויים.

שליחת פרמטרים ל- action:

שרשור משתנים בכתבota ה- URL ע"י "?name=Dan" וכו'.
המשתנים מגיעים ל- action ושמורם ב- Request.QueryString ב-

ב-טכנולוגיות MVC אם מגדירים בתחום הסוגרים של ה- action משתנים ששם הוא בדיק
שם המשתנים הנמצאים ב- Request אוטומטית הערכים נכנסים למשתנים אלו.

כך יצא שם שרשרנו "name=Dan" וקיים ב- action(string name) :action . Request.name יתמלא בערך Dan שmagiu ל- Request .

העברת נתונים מ- action ל- view:

1. ע"ו : `@viewData["key"]` :view ו-הגישה לcker ב- `viewData["key"]=Value`
או **לחילופין**: `@viewBag.key` :view ו-הגישה לcker ב- `viewBag.key=Value`

2. שליחת model: שליחת אובייקט שלם ל-view: יצרתו או קיבלתו מה- DB והעבرتו ל-view בהתאם להסוגים של ה- view. (אפשר לשולח אובייקט מכל טיפוס שהוא: לדוג:

public Action Index()

1

```
מילוי רשימת התלמידות בדרך כלל היא://  
List<Pupil> model=FromDB.GetAllPupils();  
return view (model);
```

{

הגישה לצר ב- view:

יש להגדיר במחילה עמודה `model` עם איזה טיפוס של `model` הוא משתמש ע"י שורת

ההגדה:

```
@model webApplication.Model.Pupil .PupilModel
```

ב-namespace זו הוא ומצא שטח המודול

לערכים בתוך המודול יש לקרוא כך: `Id`.
@Model Name

רבסריאב לעריכים שברמודול - `view.cshtml` תמיד ל- `@Model` עם אות גדולה.

שימוש ב- Razor

- כורת שילוב של כתיבת קוד server לצד client
- אפשר לכתוב בדף ה-view קוד C# פשוט כ-foreach, For, If וכו'.
- וכן הצגת נתונים מהמודול ע"י פתיחת סימן ה-@: Razor
- איך זה עובד?
- המשתמש קורא ל- action מסוים.
- ה-action מחולל את כל הקוד שבתוכו.
- מוצאת את ה-view אותו הוא צריך להציג.
- עובר על כל קוד ה-"server" שבתוכו, אותו הוא מזהה ע"פ ה-@.
- הופך אותו לתגיות html ומליל פשוט.
- את דף ה- html הטהור הוא מעביר למשתמש.

שכפולים – שימוש ב- Razor

HTML Helpers

המחלקה הסטטית `html` מכילה בתחום מגוון של פונקציות סטטיות המחזירות כ- `string` תג `html` ומסויימות לנו בכתיבת דף ה-`view`.

פונקציות אלו שיכות ל- `server` ולכן נקרא להם בفتיחה של - `@` `html` הפונקציה: (`@Html.Text.Box()` מוחזירה לנו-tag html:

```
<input type="text">
```

יש אפשרות לשלוח בתוך הסוגרים את ה- `id` של התג ואת ה- `value` שלו, כך:

```
@Html.TextBox("Txt1", "שלום", new { id = "txt1", value = "שלום" })
```

כמו כן נראה את:

Helper

- `Html.CheckBox`
- `Html.DropDownList`
- `Html.Hidden`
- `Html.Label`
- `Html.ListBox`
- `Html.Password`
- `Html.Radio`
- `Html.TextArea`
- `Html.TextBox`

HTML Element

- `<input type="checkbox" />`
- `<select></select>`
- `<input type="hidden" />`
- `<label for="" />`
- `<select></select> or <select multiple></select>`
- `<input type="password" />`
- `<input type="radio" />`
- `<textarea></textarea>`
- `<input type="text" />`

יש לחלק בין הפונקציות שעובדות על משתנה בודד כמו `TextBox` וכו'.

לבין אלו שימושות ברשימה אובייקטים כמו `Box`,

אלו אמורים לקבל בסוגרים `po` כמו בפונקציות האחרות, אבל במקום `value` לקבל `value` עליהם לקבל רשימה של : `SelectListItem`.

המחלקה `SelectListItem` מכילה: `text` – מה להציג. `value` – מה הערך ישמר מאחוריו הטקסט ו- `selected` – האם יהיה בחור מ滥כתילה.

יש ליצור רשימה זו או ב- `controller` או ב- `view` ע"פ הנ吐נים מה-`model` ואו שהיא לשולח לפונקציה, לדוגמה:

```
@{
    var items = new List<SelectListItem>{
        new SelectListItem {Value = "1", Text = "Blue"},
        new SelectListItem {Value = "2", Text = "Red"},
        new SelectListItem {Value = "3", Text = "Green", Selected = true},
        new SelectListItem {Value = "4", Text = "Yellow", Selected = true},
        new SelectListItem {Value = "5", Text = "Black"}
    };
}

@Html.ListBox("myListbox", items, null, 6, true)
```

Html Helpers for

ע"מ להקל על המפתחים, פותחו פונקציות מקבילות לפונקציות הנ"ל המקבלות כ-*פוקו* מאפיין המודול וידועות ע"פ שם המאפיין והערך העכשווי שלו ליצור את tag-*.html*.

פונקציות אלו מקבילות לפונקציות הקודמות – אך בתוספת של המילה *for* לדוג'.

`@Html.TextBoxFor (model->model.name)`

ונניח שה- *name* כרגע הוא *Dan* – כך שהfonקציה תיצור את tag הבא:

`<input Type ="text" Id="Name value="Dan">`

כשה- פא תמיד ילקח שם המאפיין וה- *value* מהערך המוכנס בו.

יש להבדיל בין פונקציה זו ל-

`@html.LabelFor (model-> model.Name)`

שיציג ב- *value* את שם המאפיין ולא את הערך שבתוכו, כר' :

`<label for "name">name</label>`

יש לזכור שלפונקציות:

`@Html.DropDownListFor()`

`@Html.ListBoxFor()`

יש לחת בסוגרים ערך של מודל שהוא מטיפוס: <SelectedListItem>

لينك

הfonkzia:

@Html.ActionLink()

מחליפה את התגית הפשוטה

לפונקציה זו יש לחת בסוגרים: 1. השם שיוצג עבור link זה.

2. שם ה- action אליו link מפנה.

3. שם ה- controller אליו link מפנה (אופציוני). אם

המקום הנוכחי נמצא באותו controller אליו link מפנה, אפשר לוותר על הכתיבה של
פרמטר זה והוא יפנה לשם כבירה מיוחד.

4. נתונים לשדרור בשורת ה- URL (אופציוני):

יש לנכון זאת בתחריר זה: { Parameter1=value1, Param2=value2 } new לדוג:

@html.ActionLink(new { Id=Model.id }, "showDetails", "pupils", "ראה פרט תלמיד")

שם ה- action

שם ה-

controller

ונניח ש- Id.model שווה ל- 1, כך שהfonkzia תיצור את התג הבא:

ראה פרט תלמיד

יצירת form

ע"מ ליצור טופס (תגית <form>) יש להשתמש בפונקציה:

@using (Html.BeginForm()) {

יש לסגור את הצומדים בסוף הטופס כך שכל מה שהוכנו לתוך צומדים אלו יהיה עטוף
בתגיות ה- <form></form>.

מה שיעזר לנו אח"כ בשליחת הנתונים חזרה ל- server ע"י שימוש ב- submit.

HttpGet \HttpPost

יש 2 צורות עיקריות לגשת ל-`webServices`:

HttpGet

זאת הצורה הבסיסית כאשר יוצרים שלילה דרך:

1. שורת ה- `url`.

2. דרך `link`.

`HttpGet` משמש בעיקר לקבלת נתונים מה-`client` ל-`server`.

יש לציין על ה-`webservice` (ב-`mvc` זהו ה-`action`) את ה-`[HttpGet]:attributens` (action) שרשור המשתנים והערכים באופן זה יש אפשרות להעביר פרמטרים ל-`webservice` ע"י שרשור המשתנים והערכים שלהם בסוף שורת ה- `url`.

דבר זה יבוצע כך

`Http://web.domain/MyController/MyAction?paramName=paramValue`

? יבוא תמיד אחרי שורת ה-`url` לפני שרשור המשתנה.

במקרה ונרצה להעביר מספר פרמטרים נשרשר ביניהם:

לדוגמא: `Http://web.website/PupilController/Edit?ID=1&Name=rachel`

בשורת ה- `url` שרשור המשתנה מתבצע ללא מרכאות גם כאשר מעבירים ערך `string` כמו

רחל – שם פרטי, שם משפחה

איך קיבל את משתנים אלו?

יש לציין בתוך הסוגרים של ה- `webservice` את שמות הפרמטרים אליהם אנחנו מצפים.

לדוגמא כך יראה ה-`action` ששמו `edit`:

```
public ActionResult Edit(int ID, string Name)
{
    return View();
}
```

בעת השילחה ל- `edit` ערכים אלו יתملאו מידית.

אין משמעות לסדר בו נשלחו הערכים אלא רק לשם עלי. כן יש לציין את השם המדויק (רגיש לאותיות קטנות/agdolot). אין חובה לקבל או לשלוח את הערכים בהתאם, אפשר בהחלה לשלוח או לקבל יותר ערכים, מה שמתאים יכנס.

הפרמטרים שנשלחו נוכנים גם לאובייקט: Request.QueryString
המכיל רשימה של-key (שמות הפרמטרים) ו-value (ערך שהם מכילים)
כך שבמקום לקבל אותם בסוגרים אפשר גם לשלוף אותם כך:

```
Request.QueryString["name"];  
Request.QueryString["id"];
```

בשפת ASP.net זאת הצורה היחידה להגעה לפרמטרים שנשלחו.

HttpPost

get P 1511 11/10/10 180 268 422

.server-7 נס סטטוס וריאנטים זיהויים

בשיטת זו הפראמטרים אינם משורשרים לשורת ה- Hub אלא מעוברים בצורה פנימית. המעלות:

1. אפשר לשלו כמות גדולה של נתונים, `HttpGet` מאפשר העברת נתונים עד לגודל מוגבל בלבד.
 2. הנתונים לא נראה למשתמש כך שהם מאובטחים.

צורת השימוש:

נעדר בתג `<input type="submit">` ובתג `<form></form>` ע"מ להעביר את הנתונים מהמשתמש (ה-client) ל-server.

כאשר אנו פותחים טופס (תגית `<form>`) המכיל תגית של `submit` מסוג `submit` ותגיות `button` אחרות כר:

```
<form>  
<input type="text" name="txtName" />  
<input type="password" name="password" />  
<input type="submit" value="שמור" />  
</form>
```

בעת לחיצה על ה- `submit` יועברו ל- `server` אוטומטית כל הערכים של שדות ה- `input` הנמצאים בתחום הטופס (בין תגיות ה- `form`) כשם הערך הוא ע"פ ה- `name` של ה- `input`. הערך שלו הוא ע"פ ה- `value`.

יועברו ל- `?server` ?לייזה `?webService` ?לייזה `?action`

```
<form action="/ControllerName/ActionName" > />  
ובשימוש ב- ((@using (Html.BeginForm("ActionName", "ControllerName")) {
```

מכיוון שליטה ע"י () HtmlBeginForm יוצרת כברית מחדל שליחת נתונים כ- httpPost יש להגדיר את ה- action כ- httpPost.

שליחת הנתונים מהמשתמש

כל ה- action עד עכשו היו webServices שפועלים בשיטת ה- httpGet זהה בירית המחדל של כל action הנמצא ב- controller . אפשר לציין זאת מפורשת ע"י הגדרת [HttpGet] מעל שם ה- action.

עת- ע"מ לקבל נתונים מהמשתמש נגיד action שיפעל כ- httpPost ע"י הגדרת ה- [HttpPost] מעל שם ה- action אליו יועברו הנתונים. מתקבל להשתמש ב- Get כאשר רוצים לשולף נתונים למסך וב- Post כאשר רוצים לקבל נתונים מהמשתמש ולעדק אותם בשרת.

הנתונים נשלחים ל- Request:

ב-טכנולוגיות MVC אם מגדרים בתוך הסוגרים של ה- action משתנים ששם הוא בדיק כשם המשתנים הנמצאים ב- Request אוטומטית הערכים נכנסים למשתנים אלו.

במוקום להגדר בתוך הסוגרים רשיימה ארוכה של כל הנתונים שמקבלים מהמשתמש, אפשר ליצור מחלוקת המכילה מאפיינים עם שמות זיהים ולהשתמש בה.

לדוגמא במקום:

```
[HttpPost]
public ActionResult Create(string Fname, string Lname, int Age)
{
    return View();
}
```

אפשר ליצור את המחלוקת:

```
public class PupilModel
{
    public string Fname { get; set; }
    public string Lname { get; set; }
    public int Age { get; set; }
}
```

ולהשתמש בה כך:

```
[HttpPost]
public ActionResult Create(PupilModel model)
{
    return View();
}
```

ה- model (מקובל לקרוא לזה בשם model אף כמובן זה לא חובה) יתמלא אוטומטית מה- request בבדיקה כפי שהערכים מולאו קודם לכן.

התהיליך שלם:

- העברנו מודל ל- view מטיפוס מסויים לדוג' pupilModel.
- ב- view עצמו השתמשנו ב- Html Helpers For שיצירות לנו תגיות submit כשה- id שלהם הם קפי שם המאפיין במחלקה.
- כך שבעת Submit שמויות הפרמטרים שעוברים ל- Server הם בדיקם כמו שמות המאפיינים ב- pupilModel.
- אם כך כמובן שנוכל להשתמש ב- Model עול מנת לקבל לתוכו נתונים מהמשתמש.
- כך נוצרת האפשרות שלחכנו את המודל וקיבלו אותו חזרה.**

הערה חשובה:

אם לא הגדרנו בתוך תגיית ה- form בדיק עלאייה action לפנות, הוא יפנה ל- action בשם זה הקיים בcontroller בשם הקיים בה נמצאת ה- view, וכך שמדובר לבנות עבור כל view שני action באותו שם:

1. httpGet – תהייה ריקה ותחזיר view.
 2. httpPost – שתקבל את המודל.

מעבר בין "action"ים

ע"מ לעبور בין "action"ים שונים נקרא לפונקציה:

Html.RedirectToAction(ActionName, ControllerName, model)

כששוב- יש לציין את שם ה- controller רק אם ברצונינו לעبور ל- action הנמצא ב- controller אחר זהה בו אנו כעת נמצאים.
ה- model הוא אופציונלי - אם אנו מעוניינים להעביר אובייקט כלשהו (וכמובן שרק מקובל לקרוא לו בשם זה).

מתי זה שימושי?

אחרי קבלת הנתונים מהמשתמש- שמירתם בדיקתם וכו' , נרצה להחזיר למשתמש דף אחר,
זאת נעשה ע"י משפט ה- Redirect .

Partial View

דף **view** מורכב המכיל כמה חלקים שונים, כדאי להפריד כל חלק ולכתוב אותו כדף **html** נפרד (בדומה ל- **user control**), כך שה- **view** המורכב יוכל לקרוא ל- **PartialView**.

יצירת PartialView

לחיצה ימנית על התיקיה המתאימה מ太久 תקית ה- **views**.
לבחור: "add view" ו"ולסמן" ("creat as PartialView").
יש לתת ל- **Partial** שם, מקובל להתחיל את השם עם קוו תחתוי, לדוגמה: "**_pupilGrade**"

קריאה ל- PartialView

ישנים 2 סוגי קריאות:

@html.partial .1

ע"מ להציג PartialView פשוט יש להשתמש ב:

```
@html.partial("_partialName", Model);
```

יש לציין מהו שם ה- **partial** ויש אפשרות להעביר לו אובייקטים – אפשר להעביר את כל המודל שקיבלו בו- **view** ואפשר רק חלקocr:

```
@Html.Partial("_partialName", new {id=Model.Id, name=Model.Name})
```

חשוב לדעת:

ה- **compiler** מחפש את הדף לפי השם שצווין רק בתוך התיקיה בה ממוקם ה- **view** שקרא ל- **partial**.

אם הוא לא מוצא שם **partial** בשם זה, הוא מחפש בתוך תיקיית ה- **shared** שהיא תיקיה המוכרת בכל המערכת.

ע"כ יש למקם את ה- **partial** תחת התיקיה בה נמצא ה- **view**, כדי ש**partial** כללי יש למקם אותו בו- **shared**.

@Html.Action .2

כאשר מעוניינים לגשת ל- **action** מסוים שיכן לנו את ה- **partial** לפני הצגתו משתמש ב:

```
@Html.Action("actionName", "controllerName", new {id=Model.Id})
```

ה- **action** יוגדר כ- **Http Get** כמו כל **view** – ויחזיר **.returnPartialView()**

Ajax

הדף יכול לבקש בקשות מהשרת ב-2 דרכים:

1. בקשה Html – הדף מבקש להציג דף שונה => כתובות ה- url תשתנה.
2. בקשה Ajax – הבקשת נשלחת מקוד javascript והתוצאה חוזרת לקוד => לא רואים שינוי בדף.

בקשת Html

כל הבקשות שהכרנו עד היום נכללות בסוג בקשה זו. לדוגמה: – אומר לדף
להביא דף אחר ונראה את השינוי בכתובת ה- url.
גם קווין מסוג submit מבקש בעצם דף אחר מהשרת ולכן נדרש להחזיר לו view מסוים.

בקשת Ajax

נזכיר לדוג' שירות חיצוני שמחזיר לנו את תחזית מזג האוויר ל- 3 הימים הקרובים, השירות זה
הועלה לאוויר כ- webService והכתובת שלו היא www.weather/3day. השירות מחזיר כ-
string את התחזית, לדוג' "18/01/13 15-18, 29/01/13 ...".
כasher נכתבת הכתובת בשורת ה- url נראה את המלול הנ"ל מופיע.
יש לנו אפשרות לקרוא לשירות זהה גם מתוך הקוד. והכוונה מתחום כל קוד שהוא:
C#, java, .NET, javaScript, ווכו' נוכל לקרוא לשירות החיצוני גם דרך קוד

קריאה Ajax מקוד js:

```

$( "#MeetingID" ).click(function () {
    $.ajax({//AJAX
        type: "GET", // POST-GET או ב-GET
        url: "/Meeting/GetMeeting", //פונט/
        dataType: "json", //DATA-L-JSON
        data: { "meetingID":$(this).val() }, //ACTION
        success: function (data) {
            //parse json to object
            var meeting = jQuery.parseJSON(data);

            $("#MeetingSubject").val(meeting.MeetingSubject);
            $("#Descripion").val(meeting.Descripion);
            $("#Date").val(meeting.Date);
            $("#Place").val(meeting.Place);
        },
        error: function (xhr, ajaxOptions, thrownError) {
            alert("fail")
        }
    });
})

```

בנייה של שירות ל- ajax

בדיקת כמו שnitן להתחבר לשירות מזג האוויר, ניתן להתחבר גם לשירות שיצרנו בעצמנו. וכיון שהיא- "action"ים הם בעצם webServices ניתן לקרוא ב- ajax ל- action מסוים. נרשום ב- url: @url.content("~/controller/action")

שימושים נפוצים ב- ajax:

1. כישיש דף שבו מוצגת טבלה של כל המוצרים בchnoot לדוגמה, ובעת לחיצה על שורה נוספת למטה טבלה ובזה מוצגים פרטי מוצר הנבחר.

בעת הלחיצה על השורה – לא נרצה שככל הדף יגיע מחדש אלא שרק הטבלה למטה תשתנה, לכן בקוד ה- razor נגידיר שבעת click על השורה ישלח ajax שיחזר PartialView עם הנתונים המתאים.

ה- Partial יחזיר לנו ל- data של ה- success כרגע של string שמכיל את כל הדף כולל תגיית ה- Html. קוד ה- razor יחולף ל- Html שמאחוריו יוחזר אף הוא. נדרש רק למקם את כל ה"מלל" הזה במקום הרצוי לנו בדף, והדף יוצג.

2. כאשר נתונים למשתמש לעדכן פרטיים, לדוג' את פרטי האישים – לא תמיד מעוניינים בעת save להוביל את המשתמש לדף אחר לפעמים מעדיפים רק להראות לו באיזה div עליון:

"פרטיך נשמרו בהצלחה" – כיצד נבצע זאת? את השיליחה של "save" נבצע ב- ajax וnochzir כ- string האם זה הצליח, ב- success נציג את המלל שוחרר ב- div העליון.

3. כאשר יש כמה comboBox שבעת הלחיצה על אחד מתמלא ע"פ הבחירה השני לדוג' בחירת ארץ ולפיה נמלא את שמות הערים. ע"מ למלא את הרשימה נהיה ח"יבים לפני לשרת, אך כמובן שלא נרצה להביא את כל הדף מחדש ונפנה ב- ajax נקלט ב- data הרשימה וככניס אותה ל – combo המתאים.

יש ל- ajax עוד שימושים רבים ובקיצור:

כל פעם שנרצה להגיע לשרת מוביל להביא את כל הדף מחדש בשיטת ajax.

החזורת אובייקט לבקשת ה-Ajax:

כל service מסוג Http (זהו הסוג בו אנו משתמשים ב – mvc) יכול לקבל רק string ולחזר רק string – return view/ PartialView !string מהזיר את הנתונים כ – string. ניתן לראות זאת כشمפלים את ה – Partial ב – data.

אם מעוניינים להעביר רשימה של אובייקטים או אובייקט מורכב – כמו: רשימה של ערבים – פרטיהם של לקוחות יש להפוך אותם ל – json שזה סוג של string.

Ajax Begin Form

כאשר אנו מעוניינים שכפתור ה- `submit` שלנו לדוג' `save` יופעל כ- `ajax`, ע"מ לחסוך לנו את כתיבת `-ajax` ב-`click` של הכפתור ושליחת הנתונים מהמשתמש ב-`-data`:
אפשר להשתמש ב- `@using.AjaxBeginForm` (בדוק בדרך בה השתמשנו ב-`@using`) – ובעת לחיצה נגיע ב- `-ajax` ל- `action` שמוגדר.

אם לא נזכיר לאיזה `action` לפנות, נגיע ל- `action` בשם שמו שם ה- `view` ונמצא ב- `controller`
שםו בשם התקיימה בה נמצא ה- `view`.

ב- (?) `update` נגדיר את ה- `id` של ה- `div` ששם נרצה שיוצג מה שוחרר לנו מהשרת.

layout

יצירת דף עם עיצוב בסיסי שכל שאר הדפים ירשו ממנו – ויכילו את העיצוב שלו (בדומה ל- `masterPage` ב- `asp.net`).
דף ה- `layout` הוא `view` פשוט הנמצא תחת תיקית ה- `views` תחת תיקית ה- `shared` (ע"מ
שיכול להיות מוכר לכל התקיות).

דף זה יוכל לתמוך `Html` פשוטות ותגיות של `reazor`.
התגית (`@RenderBaby()` מצינית את המיקום אליו יכנס כל הדף שירש את ה- `layout`).
בד"כ נמקם את התגית זו במרכז הדף.
ע"מ לקבוע שדף מסוים ירש מ- `layout` נרשם:

```
@{
    Layout = "~/Views/Shared/LayoutName.cshtml"
}
```

שמירת מצב באתר:

Application

שמירת נתונים בשרת עבור כל הלקוחים. הנתונים נשמרים כל עוד לא עשו **restart** לשרת. יש לנעול את האובייקט ע"מ למנוע הכנסה ע"י 2 משתמשים בו זמן קצר דבר העולם לגרום שאחד מהם לא יוכל להכניסו.

צורת שימוש:

הכנסה: Application["count"] = 0;

הוצאה:

```
Application.Lock();
int x = (int)Application["count "];
x++;
Application["count "] = x;
Application.UnLock();
```

Cookies - עוגיות

שמירת מידע בקובץ קטן במחשב הלקוח. שמירת נתונים זו היא פרט גולש. שימוש עיקרי: שמירה פרטיהם על הלקוח (כמו שם) ע"מ שבפעם הבאה נדע עליו יותר פרטים בעת כניסה לאתר.

לעוגיה יש **לתת הגבלת זמן** אחרת היא נמחקת בעת סגירת הדף.

חשוב לדעת:

- העוגיה נשמרת על הדף ולכון 2 דפים לא יכירו את העוגיות הנמצאות אחד אצל השני.
- הדף נמצא אצל המשתמש כך שהוא יכול למחוק את העוגיות – אך אין להסתמך על שמירה זו.

דוגמה:

הכנסה:

```
HttpCookie cookie = new HttpCookie();
cookie.Name = "UserName";
cookie.Value = "SaraD";
cookie.Expires = DateTime.Now.AddYears(1);
Response.Cookies.Add(cookie);
```

הוצאה:

```
string x = Request.Cookies["UserName"].Value;
```

שים לב לשימוש ב-**request** ע"מ לקבל פרטיים השמורים על הדף וב- **response** ע"מ לתת פקודה לדף לעשות פעולות.

שמירת מצב באתר:

Application

שמירת נתונים בשרת עבור כל הלקוחים. הנתונים נשמרים כל עוד לא עשו **restart** לשרת. יש לנעול את האובייקט ע"מ למנוע כניסה ע"י 2 משתמשים בו זמנית דבר העולם לגרום שאחד מהם לא יצליח להיכנס.

צורת שימוש:

הכנסה: Application["count"] = 0;

הוצאה:

```
Application.Lock();
int x = (int)Application["count "];
x++;
Application["count "] = x;
Application.UnLock();
```

Cookies - עוגיות

שמירת מידע בזיכרון קטון במחשב הלקוח. ^{client} שמירת נתונים זו היא פרט גולש. שימוש עיקרי: שמירת פרטים על הלקוח (כמו שם) ע"מ שבפעם הבאה נדע עליו יותר פרטים בעת כניסה לאתר.

לעוגיה **יש לתת הגבלת זמן** אחרת היא נמחקת בעת סגירת הדף.

חשוב!

- העוגיה נשמרת על הדף ולן 2 דפים לא יכירו את העוגיות הנמצאות אחד אצל השני.
- הדף נמצא אצל המשתמש כך שהוא יכול למחוק את העוגיות – لكن אין להסתמך על שמירה זו.

דוגמא:

הכנסה:

```
HttpCookie cookie = new HttpCookie();
cookie.Name = "UserName";
cookie.Value = "SaraD";
cookie.Expires = DateTime.Now.AddYears(1);
Response.Cookies.Add(cookie);
```

הוצאה:

```
string x = Request.Cookies["UserName"].Value;
```

שים לב לשימוש ב-**request** ע"מ לקבל פרטיים השמורים על הדף וב- **response** ע"מ לתת פקודה לדף ל לעשות פעולות.

Session

אובייקט המאפשר לשמר נתונים עבור כל גולש בזמן הגלישה שלו (שימוש נפוץ: שמירת רשימת המוצרים שהגולש מעוניין לקנות).

ברגע שהמשתמש סגור את הדף – האובייקט נעלם.
יש אפשרות למחוק אותו גם ע"י הגדרת `timeout` או הצבת ערך `null`.
שים לב: האובייקט נשמר בשרת כך שהוא בטיחותי. יחד עם זה ע"מ לניהל איזה אובייקט קשור לאיזה גולש, יש לכל אובייקט קוד הנשמר אצל הגולש כעוגיה.
שימוש מרובה ב-Session עלול לגרום למערכת איטית.

הכנסה:

```
int value = 3;  
Session.Add("name", value);
```

היצאה:

```
int getValue = (int)Session["name"];
```

שימוש ב **action**_**method** **view**

תרגיל:

צרי controller controller בשם controller controller

מכל את ה actions-הבאים:

-list的家庭 view בו מוצגים שמות בני המשפחה.

-abont家庭 view בו מוצגים פרטים נוספים על המשפחה

צרי controller controller נוספת בשם classcontroller controller

מכל את ה actions-הבאים:

-list家庭 view בו מוצגים שמות הבנות בכיתה

-abont家庭 view בו מוצגים פרטים נוספים על הклассה

צרי את ה controller-controller הראשי בשם controller controller

מכל את ה - action-myinformation: ובו פרטים על עצמו

בדף הראשי{myinformation}צרי 2 לינקים המובילים

-familylist1 מוביל לדף list השיך לfamily-

-classlist2 מוביל לדף list השיך לclass-

בכל אחד מ-2 דפים אלו צרי לינק המוביל לדף ה-about-heory אליו
ובכל אחד מן הדפים יש לייצור לינק-חזרה-הmóvel לדף הקודם.

העברת נתונים מ-view- controller

תרגול:

צרי controller controller בשם homecontroller

בתוכו צרי action בשם index שם מקובל לדף הראשון במערכת [

ובו הציגי 2 לינקים:

1. פרטי משתמש - controller-user: [

ובתוכו action בשם show שלפיו את המשתמש הראשון הקיים ב-

והציגי את כל פרטי

[לא הפגישות שלו].]

2. פרטי פגישה - controller-meeting: [

בתוכו action בשם show שלפיו את הפגישה

הראשונה הקיימת ב-DB- והציגי את כל פרטייה [לא המשתתפים שלה

שימוש ב razor-וב htmlhelper

תרגול

1. צרי service בשם index בתוך controller-user.

בו תציגי בטבלה את כל פרטי ה user -בעמודה האחורונה
יהיה לינק המוביל לציגות ה SHOW-עבור ה user-ה הנוכחי כך:
חסר טבלה

יש לשנות את ה show:action - שיקבל את ה id - הנוכחי

{ בלינק של update יש לשרר את ה id-של השורה

וישלוף מה {DB - את פרטי {במקום את הראשון

2. צרי את כל הנ"ל עבור meeting

3. בציגות ה SHOW-אפשרי למשתמש לשנות פרטיים

הוסף כפטור save לשמרות הנתונים בDB -

כפטור זה יוביל ל post service - היקבל את כל הנתונים

יעדכן אותם ב , B -DB -ויחזיר את המשתמש לציגות index -

השתמשי ב chiong redirect to action -index -

שים לי: יש להציג את השדה -id גם אם אין

ברצונך לראותו

[אפשר ב [hidden] -הע"מ שב post -תקובלי זאת .

4. צרי לינק לטבלה ה users:createuser -בעת לחיצה עליה יפתח view ובו אפשרות למלא

את כל פרטי ה user -בעת לחיצה על save ישמרו הנתונים בDB -

שים לי: יש ליצור actions2

1. מסוג get היחזיר את ה view

2. מסוג post היקבל את הנתונים תישמרו אותם בDB -

ויחזיר ע"י redirect to action לדף index -

5. הוסף עמודה נוספת עם לינק ל delete -בעת לחיצה עליו תמחק השורה מה DB -

cookie session application**תרגול**

1. צרי service המחזיר את כמות הגולשים באתר
צרי לינק המוביל ל-[index] מהדף הראשי[index]
השתמשי בAPPLICATION-ע"מ לשמר את כמות הגולשים.
2. צרי טופס login למערכת לפני החזרת הטופס הראשי ל-index service
בצעי בדיקה:
אם משתמש קיים cookie בשם login או session בשם זה - בדק את השם
והסימא השמורים שם אם הם קיימים ב-DB אם כן תוכל להחזיר את הדף ל-index.
במקרה שלא שומר cookie או session או שהשם והסימא לא תואמים למה
שמStored בDB, יש לתת למשתמש טופס אחר.
בשם LOGIN : יצירת service של GET ונתוך index יעשו redirect to action
אלין]
המשתמש יכנס בטופס זה את השם והסימא ויכול לבחור האם לשמר את הנתונים
לטוען ארוך
הטופס יראה כך:
חרס
שמירת מצב באתר מע"מ 14
להעתיק לפי הסימונים

[http://jqfundamentals.com/book/MVC Knowledge & Tips](http://jqfundamentals.com/book/MVC_Knowledge & Tips)

תוכן עניינים

2	لينكيم חשובים	1.
2	שם הפקד	2.
2	החוורת נתונים מה-view ל-controller והפוך	3.
3	בנייה רשומות תלויות	4.
4	ולידציות ב-MVC	5.
4	שימוש בתשתיית DataAnnotations5.1
9	שימוש בתשתיית FlunnetValidation	5.2.
10	ולידציות המבוצעות בצד הסרבר	5.3.
12	בעיות ידועות	5.4.
13	ערך "בחר" בשדה קובמו	6.
13	בעיה בביצוע אנימציה של jquery ב-IE	7.
14	פקדי טריק	8.
14	בעיה בולידציה עבור פקד Editor	8.1.
15	בעיה בביצוע אנימציה (slides וcdc) על פקד Grid של טריק	8.2.
15	הfonקציה load של ajax jquery מגיעה לשרת רק בפעם הראשונה - Ajax	9.
		Cache

15

1. לינקים חשובים

כותרת	נושא
לימוד MVC לעבור על כל 10 היסטרוניים מצד שמאל כהתחלת ואח"כ לעבור על הנושאים שצד שמאל, שייתר ננסים לעומק הנושאים www.asp.net/mvc	לימוד MVC
לימוד MVC http://msdn.microsoft.com/en-us/library/gg416514(VS.98).aspx	לימוד MVC
רשימת HtmlHelpers ק"מ"ים (לדוגמא: (@Html.TextBox	HtmlHelpers
תיעוד MVC כללי של מיקרוסופט http://msdn.microsoft.com/en-us/library/dd410596(v=vs.98).aspx	תיעוד MVC כללי של מיקרוסופט
jQuery של שפת ה-ASP.NET http://www.w3schools.com/jquery/jquery_ref_selectors.asp /http://jqfundamentals.com/book	jQuery

2. שם הפקד

חשוב לדעת כי בכל ה-`HtmlHelper` הקיימים, ישנה משמעות כפולה לשם הפקן:

- שם של המאפיין המחבר אליו ב-Binding במקורה שמודבר ב-View העובד מול מודל נתונים. זו למעשה הדריך להגדיר את Binding לפקד. אם מדובר על אובייקט מורכב כגון Person שמכיל בתוכו מאפיין מסווג CityName
 - Address שמכיל בתוכו מאפיין בשם CityName, כדי להגדיר TextBox פקד TextBox הקשור לשדה CityName (@Html.TextBox(Address.Name))

3. החזרת נתונים מה-view ל-controller והפוך

- controller יכול לשולח נתונים ל-View בשתי דרכים:
 - מילוי נתונים ב- ViewBag/ ViewData Controller. הוא יכול לשים ערכים באובייקטים אלו וה-View יכול לגשת אליהם. שיב האובייקטים הנ"ל משמשים לאותה מטרה, ה- ViewData הינה המירה ViewData2 MVC2 ואילו ה- ViewData נסוך בגישה MVC3. ההבדל ביןיהם הוא סוג האובייקט. ה- ViewData הינו מסוג Dictionary<string, object>. ה- ViewData["CustomerNum"] = 1 הינו אובייקט דינמי וע"כ השמת ערך תבצע בדרכו. הבאה: ViewData.CustomerNumber = 1
 - @Html.DropDownListFor(model => model.LanguageId, new SelectList(ViewBag.AllLanguages, "Id", "Name"))
 - ובודה עם מודל נתונים, ה- controller ימייר את האובייקט עם הנתונים הנדרשים להציגו ושולח אותו לו-View: return View(Customer). ViewData[@model.Customer]
 - לדוגמא: @Html.EditorFor(model => model.Title)
 - מומלץ לעבוד תמיד עם מודל המותאם לצרכי ה-View, מה שנקרה ViewModel ולא עם המודלים של שכבת ה-DAL, זה מומלץ גם מסיבות אבטחה כיון שהמשתמש יכול לעפעמים דרך ה-URL שהוא כותב להזין נתונים למודל שמאחוריו ה-View, ולכן תמיד נשים ב-ViewModel שעבורם ה-View רק את הנתונים שאנו מאפשרים להזין. לדוגמא: נתן המגדר על משתמש אם הוא Admin או לא, לא נשווה אף פעם במודול המוצג ב-View.
 - בכיצוע post ל-form ב-view, ישלו כל הנתונים הקיימים על הטופס ל-Action שהגדר ושתלו בתוכו הפרמטרים המוגדרים ב-Action על פי שמם. למעשה, ה-Action מקבל request וידע לעמירו לפרטיטם שהוגדר. לדוגמא: אם הגדרתי פקד טקסט בשם Name, ביצוע post הערך הקים בפקד ישתלה לתוך פרמטרו רשות Name את הערך ב-Action. אם ה-Action מחייב מסוג מחרקה המכילה בתוכה

מיפויים נוספים, הערך של הפקד Name ישתל בתוך מאפיין בשם Name אם קיים במחלקה שהתקבלה כפרמטר ב-`Action`.

- המאפיין Request.Form המוכר ב-`Action` שנקרא `C-Post`, מכיל את כל הנתונים שהועברו ב-`request`, ניתן להשתמש בו לצרכי debugging כדי להבין למשל מדוע פרמטר מסוים ב-`Action` לא יוכל ערך.
- עפ"י הנ"ל, רשימות המוצגות בפקדי קומבו לדוגמא, לא תעבירנה בביטוי `-Action-Post` למורתו שהן מוגדרות במודל שאיתו עובד `view`. כלומר, בביטוי `post` ל-`view`, יש צורך למלאת שוב את פקד `html` (לדוגמא, למלאת שוב את המאפיינים המתאימים במודל המוחדר `l-View`), אך ב-`ajax form`, פקד `html` הקומבו לא נטענים מחדש מחדך ולכך הנתונים נשאים ב-`view`.
- מומלץ לשילוח רשימות למילוי פקד קומבו וכדי דרך `ViewBag`/`ViewData` ולא מודול.
- בקריאה ל-`Action` שלא ע"י `form` ל-`post` (לדוגמא: `Html.ActionLink`), יש להבין שהקריאה מתבצעת ב-`get` וע"כ אין אפשרות להעביר דרכו את כל המודול אלא ורק מאפיינים שונים בצעע עברום המירה למחזרות והופר. המיפוי לתוכה הפורטטים של `Action` מקבל געזה בזורה לנ"ל אלא שבמקרה זה עברים רק הפורטטים שהוגדרו במפורש בקריאה ל-`action`.

4. בניית רשימות תלויות

ישן הרבה שיטות לבניית פקד רשימה שהתוכן שלם תלוי באיבר הנבחר בפקד אחר (לדוגמה קומבו ערימה המתמלא רק לאחר בחירת ארץ בקומבו ארצות). נター כאן כמה שיטות ודוגמאות חשובים.

- מילוי הרשימה התלויה ע"י קרייה לפונקציה ב-`Server` תוך שימוש ב-`Ajax`. להלן דוגמא למילוי קומבו:

```
<script type="text/javascript">
$(function()
{
    $("#@Html.FieldIdFor(model => model.OfferModel.CategorySubID)").change(function()
    {
        var selectedItem = $(this).val();
        var selectedText = $(this).find(":selected").text();
        var ddlProduct = $("#@Html.FieldIdFor(model => model.OfferModel.ProductID)");
        var ddlManufacturer = $("#@Html.FieldIdFor(model => model.OfferModel.ManufacturerID)");
        var ddlModel = $("#@Html.FieldIdFor(model => model.OfferModel.ModelID)");

        if(selectedItem > 0)
            $.ajax({
                cache:false,
                type: "GET",
                url: "@(Url.Action("GetProductListByCategorySubId", "Offer"))",
                data: { "CategorySubId": selectedItem, "addEmptyStateIfRequired": "true"
            },
            success: function (data) {
                ddlProduct.removeAttr('disabled');
                ddlProduct.html('');
                $.each(data, function(id, option) {
                    ddlProduct.append($('' + option.name + '').val(option.id));
                });
            },
            error:function (xhr, ajaxOptions, thrownError){
                alert('Failed to retrieve products.');
            }
        });
        else
        {
            ddlProduct.val(0);
            ddlProduct.attr('disabled', true);
        }
    });
});
```

הערה[1]: רישום לאירוע `change` של הפקד `CategorySubID`

```
$($("#@Html.FieldIdFor(model => model.OfferModel.CategorySubID)").change(function() {
    var selectedItem = $(this).val();
    var selectedText = $(this).find(":selected").text();
    var ddlProduct = $("#@Html.FieldIdFor(model => model.OfferModel.ProductID)");
    var ddlManufacturer = $("#@Html.FieldIdFor(model => model.OfferModel.ManufacturerID)");
    var ddlModel = $("#@Html.FieldIdFor(model => model.OfferModel.ModelID)");

    if(selectedItem > 0)
        $.ajax({
```

הערה[2]: מצביאת הפקד התלו依 – קומבו מוצרים

```
        cache:false,
        type: "GET",
        url: "@(Url.Action("GetProductListByCategorySubId", "Offer"))",
        data: { "CategorySubId": selectedItem, "addEmptyStateIfRequired": "true"
    },
    success: function (data) {
        ddlProduct.removeAttr('disabled');
        ddlProduct.html('');
        $.each(data, function(id, option) {
```

הערה[3]: הקריאה ל-`server` לקבלת הרשימה תבוצע ב-`Ajax` כדי לחסור רפשוש של כל הדף

```
        ddlProduct.append($('' + option.name + '').val(option.id));
    });
},
error:function (xhr, ajaxOptions, thrownError){
    alert('Failed to retrieve products.');
}
```

הערה[4]: יקרא Action בשם `GetProductListByCategorySubId` ב-`Offer Controller`

```
},
```

הערה[5]: הגדרת הפורטטים שישלחו ל-`Post`

```
success: function (data) {
    ddlProduct.removeAttr('disabled');
    ddlProduct.html('');
    $.each(data, function(id, option) {
```

הערה[6]: רישום לאירוע `succedd` של ה-`Action`

```
        ddlProduct.append($('' + option.name + '').val(option.id));
    });
},
error:function (xhr, ajaxOptions, thrownError){
    alert('Failed to retrieve products.');
}
```

הערה[7]: מעבר על רשימה הנתונים שהתקבלה ומילוי קומבו מוצרים על ידה

```
});
```

```

        }
        ddlManufacturer.val(0);
        ddlModel.val(0);
        ddlManufacturer.attr('disabled', true);
        ddlModel.attr('disabled', true);
    });
});

```

</script>

- חשוב בכל השיטות לחשב על נושא ה-performance ובמידת האפשר לטעו מעבר ל-server. לדוגמה: בדף קיימים שני פקדי radio, בבחירה ה-radio השני מאפשר פקד קומבו אשר מציג רשימת ארצות לבחירה. מכיוון שרשימה הארצות אינה תלויה בערך נבחר בפקד אחר, חשוב לשימוש לב למלאת את הרשימה רק פעם אחת. אפשר לבצע זאת ברישום לאי-ריעוע click על ה-radio השני ממש כפי שהוזכר בדוגמא הקודמת, אלא שחשוב להוסיף שם תנאי שבודק וקורא לשורת לקבלת הרשימה רק במקרה שהקומבו עדין רק.

5. וולידציות ב-MVC

5.1 שימוש בתשתיות DataAnnotations

השיטה הרווחת לוולידציות הינה שימוש ב- DataAnnotations המימוש בצורה של Attributes על מודל הנתונים מה שמאפשר להשתמש באוטן הגדרות גם בוולידציה ברמת המסר וגם בוולידציה ברמת ה-service או שכבת ה-BL האחראים על שימרת הנתונים.

דוגמא להגדרת `:DataAnnotation`

```
[Range (1, int.MaxValue, ErrorMessage = "Product ID should be greater than 1")]
public int ProductID { get; set; }

System.ComponentModel.DataAnnotations :namespace
```

שניהם הרבה Attributes מוכנים הק"מם תחת ה-`ValidationAttribute`. דוגמא ל-`DateRangeAttribute` המשמש ניתן תמיד להוסיף עוד `attributes` ע"ז ירושה מהמחלקה `ValidationAttribute`. דוגמא ל-`EndDate` שנקבע לבדיקה טווח ערכים, לדוגמה: `StartDate` צריך להיות קטן מ-`EndDate`.

```
public class DateRangeAttribute : ValidationAttribute
{
    public TimeSpan Range { get; private set; }

    public String OtherPropertyName { get; private set; }

    public ValidationMode DateMode { get; private set; }

    //Constructor to take in the property name that is supposed to be checked
    public DateRangeAttribute(String otherPropertyName, ValidationMode
dateMode, TimeSpan range)
    {
        OtherPropertyName = otherPropertyName;
        DateMode = dateMode;
        Range = range;
    }

    /// <summary>
    /// check if it is valid
    /// </summary>
    /// <param name="value"></param>
    /// <param name="validationContext"></param>
    /// <returns></returns>
```

```

protected override ValidationResult IsValid(object value, ValidationContext validationContext)
{
    if (validationContext == null) return ValidationResult.Success;

    //Get the property info for the object passed in.
    PropertyInfo otherDateProperty =
        validationContext.ObjectType.GetProperties()
            .Where(propertyInfo => propertyInfo.Name ==
OtherDatePropertyName).FirstOrDefault();

    //Convert both values to DateTime and compare
    DateTime otherDate;
    DateTime valueDate;

    object otherDateValue =
otherDateProperty.GetValue(validationContext.ObjectInstance, null);
    if (otherDateValue == null || value==null)
        return ValidationResult.Success;

    if (!DateTime.TryParse(otherDateValue.ToString(), out otherDate) ||
!DateTime.TryParse(value.ToString(), out valueDate))
        return ValidationResult.Success;

    TimeSpan result;
    if (DateMode == ValidationMode.Bigger)
        result = valueDate-otherDate;
    else
        result = otherDate-valueDate;

    if (result > Range)
        return new ValidationResult(this.ErrorMessage);

    return ValidationResult.Success;
}
}

```

בתשתיות של אירוקס קיימים כבר מספר Attributes שהו נדרש בפרויקטם באירוקס. ה-Attributes קיימים בתחום `.Validations\ValidationAttributes` בנתיב `IROX.Infrastructure.Common.FW4` הפרויקט עבר MVC ווסף כמו `Attributes` חדשים תחת ה- `System.Web.MVC`. התווסף למשל attribute `ValidationSummary` ב掣ם Comapre המאפשר להשוות בין שני שדות.

באופן דינמי הולידציות יעבדו ב-`server side` . השימוש נעשה ע"י הגדרת `Action` וקישור `Action` עבור פעולה ה- `Post` שלו. באמצעות `Action` על אחוריות המתכוна לבדוק אם המודל ולידי לגמרי, זאת ניתן לבדוק ע"י האובייקט `ModelState` הקיים בכל `controller` וככל מופיע בשם `isValid`. במקרה שהמודל אינו ולידי, על המתכוונת להחזיר את ה- `View` שהציג קודם ורק בהחזרת ה- `View` יוצאו הודעות השגיאה למשתמש. כדי שיוצאו הודעות השגיאה, ניתן להשתמש ב- `ValidationMessageFor` `ValidationMessageFor` ב掣ם `HtmlHelper` . לדוגמה:

```

<% using (Html.BeginForm()) {%
    <%: Html.ValidationSummary(true) %>

    <fieldset>
        <legend>Fields</legend>

        <div class="editor-label">

```

```

<%: Html.LabelFor(model => model.CourseID) %>
</div>
<div class="editor-field">
    <%: Html.TextBoxFor(model => model.CourseID) %>
    <%: Html.ValidationMessageFor(model => model.CourseID) %>
</div>

```

כפי שניתן לראות שם גם helper המציג סיכון של כל השגיאות הקיימות במסך: `Html.ValidationSummary`

וילדיותה בצד הקליינט
 ניתן לגזור לוילדיותה לעובד בצד הקליינט ולהוסיף את פעולה ה-Post רק בשבייל לבדוק תקינות מידע, במקורה צזה בעת הכתנת ה-View. מוג'נרטים מאפיינים דינמיים על פקד ה-`html` המגדירים את כל המדרש עבור הפעלת הוילדיותה בצד הקליינט. סקריפט הנמצא בצד הקליינט יודע לקרוא אותם ולבצע את הנדרש בהתאם.
 לדוגמה עבור `LengthAttribute` שהוגדר עם ה��ועה ספציפית מעל מאפיין `Name` במודול, יחולל קטע ה-`view`:

```
<input id="Name" data-val="true" data-val-length="4" data-val-length-max="4" />
```

עובד ההזעה שתופיע בצד:

```
<span class="field-validation-valid" data-val-msg-for="Name" data-val-replace="true" />
```

(הערך `data-val-replace` שולט על הfrmater האם להציג את כל ההודעות שהו על שדה זה או רק את האחרונה מבינה)

שפת ה `*-data` (ובמילים נכונות יותר: `Custom Data Attributes`), היא שפה חדשה ומומלצת המתגעה כחלק מ-`HTML5`. ניתן להוסיף כל attribute שהוא המתחילה במילה `data` על כל אלמנט Html שהוא. את ה-`attributes` קוראים סקריפטים הקיימים בצד הלוקו ומציעים על פיהם את הנדרש. כך ניתן למשל לבנות סקריפט המכיר מאפייניהם כך של הרוצה יוסיף את המאפיינים לאלמנטים וכך יוכל ליכל את יכולות הסקריפט.

כגון עבדת הוילדיותה, ישנו קובץ סקריפט בשם `jquery.validate.unobtrusive.js` שיודע להתמודד עם המאפיינים המוגדרים עבור כל סוג של וילדיותה ולבדק את התקינות בצד הקליינט (הלוגיקה של בדיקת התקינות משוכפל פעמי אחת לצד השרת ופעם אחת בצד הקליינט) וכל הרוצה להשתמש ביכולות אלו מוזמן לגירט את המאפיינים הללו.

כל ה- `ValidationAttributes` של `DataAnnotation` מיושן כבר את גינרט המאפיינים ועכ' מוכנים לשימוש בצד הקליינט.

כדי להפעיל את הוילדיותה בצד הקליינט יש צורך לבצע מספר פעולות:
 1. להוסיף בקובץ הקונפיג תחת `appSetting` את ההגדרות הבאות (יתכן שגם בירית המודול):

```
<appSettings>
    <add key="ClientValidationEnabled" value="true" />
    <add key="UnobtrusiveJavaScriptEnabled" value="true" />

```

2. להוסיף את הסקריפטים הבאים תחת התיקייה `Scripts`:

- `jquery-XXX.js`
- `jquery.validate.js`
- `MicrosoftAjax.js`
- `MicrosoftMvcAjax.js`
- `MicrosoftMvcValidation.js`

3. להוסיף ברמת כל דף (והכי נכון להוסיף את זה בדף המסתור עם אחת) את השימוש ב-`helper` הבא:
`Html.EnableClientValidation(true);`

4. להוסיף עבור כל פקץ שורצים להציג עבורו את הוילדיותה את השימוש ב-`Helper` הבא:
`@Html.ValidationMessageFor(model => model.AccountType)`

וילדיותה בצד הקליינט עבור Custom Validations

כדי ליצור Validation Attribute חדש עלינו לרשט מ-**ValidationAttribute** (או לחילופין מ-**ק"מ** כבר, כגון **required**). החדש יודע בברירת מחדל לעבוד בצד השרת. כדי לגרום לו-**attribute**-**attribute** החדש שהוספנו לבדוק בצד הקליניט עליו לעמוד בקצב קשה כפי המבואר להלן.

כפי שהסביר לעיל, כדי שוילידציה תעבור בצד הקליניט יש צורך לאיג'נרט מאפיין-***-data** ו"ללמד" את ה-**JS** הקיים בצד הקליניט לקרוא אותם. על מנת כן ק"מ הממשק **IClientValidation** (שכל המאפיינים הק"מים ב-**DataAnnotation** כבר מממשים אותו). על ה-**attribute** החדש שיצרנו למש את הממשק.

מיושם הממשק דרוש למש את הפונקציה **GetClientValidationRules** שצרכיה להחזיר את הودעת השגיאה כולל הפרמטרים הרלוונטיים לבדיקה עטוף בתוך מחלקת של המלחת איך הופכים את זה למאפיין **val-data**. לדוגמה:

```
ErrorMessage="my error message"  
ValidationType="rangedate"  
validationParameters["min"] = minValue.ToString()  
ValidationParameters["max"] = maxValue.ToString()
```

באופן הנ"ל למדתי את ה-**JS** לחולל את הפרמטרים הבאים:

```
Data-val="true"  
data-val-rangedate="my error message"  
data-val-rangedate-min="5"  
data-val-rangedate-max="10"
```

עתה נותר לכתוב את ה-**js** שיעד להכיר את המאפיינים שלו ולבדק את הוילידציה החדשה שהוספה. לשם כך ניתן להשתמש בחלקים מוכנים כבר, לדוגמה: חיפוש כל מאפיין ה-**val-data** על אלמנט מסוים זהו **js** ק"ם כבר. ה-**jQuery** מאפשר להוסף מתודה **(\$.Validator.AddMethod)**, כך אפשר להוסף פונקציה חדשה שמשמשת את בדיקת הוילידציה החדשה שלו. לא נ燒ך לפרט עוד כיצד בדיקת מממשים זאת, ניתן ללמוד זאת יותר בلينק הבא:
<http://thewayofcode.wordpress.com/2012/01/18/custom-unobtrusive-jquery-validation-with-data-annotations/> המדגים שימוש של client validation (בנוסף, קיימת דוגמא במאגר הידע תחת הנושא **MVC\Examples\MvcCustomValidationExample**).

מעבר הוילידציות Localization

האסטטיבי **DataAnnotation** כולל תמיכה בשפות עבור הודעות השגיאה שיוצגו למשתמש. לשם כך מופיעו הוילידציות מכילים שני פרמטרים נוספים:

- **ErrorMessageResourceType** – סוג המחלוקת של **Resources** המכילה את התרגומים של הודעות השגיאה
- **ErrorMessageResourceName** – שם ה-**resource** במחלוקת הנ"ל

מאפיינים אלו מיועדים לעובדה מול **Resources** של **.net**.

(מעבר מי שאינה מכירה את מנגנון **resources** של **.net**, הסבר בכמה מילימ: יש ליצור קובץ **resources.resx** (או **MyResources.en-US.resx** עבור עברית ו-**MyResources.he-IL.resx** עבור ספרדית) ולהזמין אותו כזיהוי תרבות文化 culture-**עפ"י** (סימנת עפ"י) ב-

עבור אנגלית ו- MyResources.resx עבור ברירת המחדל) ולהכניס את כל התרגומים הנדרשים (מומלץ ליצר את קובץ התרגומים בפרויקט נפרד ייעוד). מגנון ה-localization של net. יודע לפנות לקובץ resource-ה המתאים עפ"י השפה המוגדרת במכשיר המריץ את האפליקציה (יתן גם ליזום גישה לקובץ resource מסויים ע"י שימי ה- culture של ה- Thread.CurrentCulture.CurrentUICulture.Thread.CurrentCulture thread).

הבעיה מתחילה כאשר לא מעוניינים לשמש במנגנון התמיכה בשפות כפ"י שהוא, אם מפני שהתרגומים נשמרים במקור חיצוני אחר או מפני שלפרויקט המכיל את המודלים אין אפשרות להזכיר את ה-project המכיל את ה-resources או מפני כל סיבה אחרת.

לשם כך ניתן ליצור Custom Validation Attribute הירש מ- FormatErrorMessage פונקציה זו תהיה אחראית לשולוף את הההודה מאותו מקור חיצוני. השיטה המקובלת היא להמשיך לשימוש במאפיין ErrorMessageResourceName כאשר הפונקציה FormatErrorMessage עשויה שימושה במאפיין זה על-מנת לשולוף את הערך הנדרש מן DB לדוגמא.

(קיים שתשתית Bairukos המוננטה מענה כליל עbor מקרה שעובדים עם קובץ resources של דוטנט אך לא מעוניינים שפרויקט המודלים יכול את פרויקט ה-resources. התשתיות כוללת שימוש חדש של כל ה-keyים ב- DataAnnotation ומאפשרת להגדיר באופן סטטי פעם אחת בעת העלאת המערכת מיהו קובץ ה-resources שאנחנו צריכים לעבד אותו. התשתיות קיימת בפרויקט FW4. Infrastructure.Common.IROX.Validation נתיב (Validations).

כפי שהסביר בסעיף הקודם ("וילידציה בצד הקליינט עבור MVC בימיום attribute חדש הינה התמיכה בוילידציה בצד הקליינט. במקרה זה הפתרון יהיה הרבה יותר פשוט כיון שאנו יכולים להשתמש כבר באותה פעולה של ה-Attribute שאנחנו יוצרים על מנת ליצור את שפט ה-val-validation המאפשרת את הוילידציה בצד הקליינט.

דוגמא:

1. שימוש ה-Attribute החדש

```
public class RangeLocalizationAttribute : RangeAttribute
{
    private ILocalizationService _localizationService;
    public RangeLocalizationAttribute(int minimum, int maximum)
        : base(minimum, maximum)
    {
        _localizationService = EngineContext.Current.Resolve<ILocalizationService>();
    }
    public override string FormatErrorMessage(string name)
    {
        var resFormat = _localizationService.GetResource(this.ErrorMessageResourceName);
        if (string.IsNullOrEmpty(resFormat))
        {
            return new LocalizedString(name).Text;
        }
        return resFormat;
    }
}
```

הערה[8]: 1. החזרת הודעה השגיאה עפ"י ErrorMessageResourceName תוך שימוש במשמעותו אחר

2. שימוש במחלקה הקיימת ModelClientValidationRangeRule המכילה את המידע לגבי גינורט מאפייני ה- :data-val

```
public class RangeLocalizationAttributeAdapter :
DataAnnotationsModelValidator<RangeLocalizationAttribute>
{
    public RangeLocalizationAttributeAdapter(ModelMetadata metadata, ControllerContext context, RangeLocalizationAttribute attribute)
```

```

        : base(metadata, context, attribute)
    }
}
public static void SelfRegister()
{
    DataAnnotationsModelValidatorProvider.RegisterAdapter(typeof(RangeLocalizationAttribute),
    typeof(RangeLocalizationAttributeAdapter));
}
public override IEnumerable<ModelClientValidationRule> GetClientValidationRules()
{
    return new[]
    {
        new ModelClientValidationRangeRule(ErrorMessage, Attribute.Minimum,
        Attribute.Maximum)
    };
}
}

```

3. קרייה ב-global.asax.cs לפונקציה :SelfRegister

```
RangeLocalizationAttributeAdapter.SelfRegister()
```

5.2 שימוש בתשתית FlunnetValidation

האוסףלי FlunnetValidation הינו אוסףלי חינמי שניתן להוריד באינטרכט המממש וולידציות על מודלים ונתרם ג'א-
.MVC

הוא מכיל ג'א רשיית חוקי וולידציה הנדרס לגדרה על המודל. חוקי הוולידציה מוגדרים במחלקה נפרדת כאשר מען
המחלקה המקורית קיים הקשר בין המחלקה לוולידציה שלה. דוגמא:

```

[Validator(typeof(OfferValidator))]
public class OfferModel : BaseNopModel
{
    public int OfferID { get; set; }
    public int ProductID { get; set; }
    public int ModelID { get; set; }
    public int? ConditionsId { get; set; }
    public int ManufacturerID { get; set; }
    public int CategorySubID { get; set; }
    public bool CategorySubIdDisabled { get; set; }
}

public class OfferValidator : AbstractValidator<OfferModel>
{
    public OfferValidator(ILocalizationService localizationService)
    {
        RuleFor(x =>
x.CountryID).NotEmpty().WithMessage(localizationService.GetResource("Offer.CountryId.Required"));
        .When(x=>x.CountryIdDisabled = false);
        RuleFor(x =>
x.StartDate).NotEmpty().WithMessage(localizationService.GetResource("Offer.StartDate.Required"));
        RuleFor(x =>
x.EndingDate).NotEmpty().WithMessage(localizationService.GetResource("Offer.EndingDate.Required"));
    }
}

```

```

        RuleFor(x =>
x.StartTime).NotEmpty().WithMessage(localizationService.GetResource("Offer.StartTime.Required"));
        RuleFor(x =>
x.EndingTime).NotEmpty().WithMessage(localizationService.GetResource("Offer.EndingTime.Required"));
        RuleFor(x =>
x.Description).NotEmpty().WithMessage(localizationService.GetResource("Offer.Description.Required"));
    }
}

```

הבעיה היא שרק חלק מן חוקי הולידציה שהוא מכיל תומכים בוולידציה בצד הקליינט.

להלן רשימת חוקי הולידציה הנתמכים בצד הקליינט:

- NotNull/NotEmpty (required)
- Matches (regex)
- InclusiveBetween (range)
- CreditCard
- Email
- EqualTo (cross-property equality comparison)
- Length

5.3. וולידציות המתבצעות בצד הסרבר

כפי הנכתב לעיל, בביצוע וולידציות בצד הסרבר, מtbody postback לדף, בודקים שהמודל אינו וolid ומחזירים שוב את אותו ה-View. בהתקנה זו ישנה בעייתיות מסוימת בביצוע הולידציות. על מנת להבין איזה, בין ראשית כיצד מתנהג ביצוע ל postback (Ajax.BeginForm ajax form Html.BeginForm html form) postback (Html.BeginForm) (Ajax Form)

לא מתבצע post מלא, למחרות שמחזירים שוב את אותו ה-view מה-controller, אין התייחסות למודל החדש שחוור והדף אינו מציג שוב. לכן, הערכים בכל השדות לא יתרוקן כלל תוכן של פקדי קומבו. חובה להכיל את הסקריפט jquery.unobtrusive-ajax

Html Form

מתבצע post מלא לדף, למחרות שערכים שהמשתמש הzin אינם נאבדים ואין התייחסות למודל החדש. לעומת זאת, תוכן של שדות קומבו למשל, יתרוקנו ולן הבחירה של המשתמש ג'כ' תעלם. הדף מציג מחדש רק הגדרו על פקד מסויים שיולה תחילה כ-disabled, גם בביצוע ה-post הוא יהיה disable

מסקנות לגבי הולידציות:

בשימוש ב - Ajax Form ו- Html Form – וולידציות סרבר לא תעבורנה כיון שהדף אינן מציג מחדש האינדיקציה עם הودעת השגיאה. ערכיו הדף כולל הרשימות ישרו. בשימוש ב - Html Form – וולידציות סרבר תעבורנה, הערכים שהzin המשמש ישארו אך רישומות שלא נשלחו שוב – view יתרוקן (ומילא גם בחירת המשתמש). הבעיה קיימת כאשר ישן רישומות בדף שהתملאו ע"י סקריפט ajax למשל רישומה של פקד קומבו התלויה בבחירה ערך בפקד קומבו אחר.

פתרונות אפשריים:

1. שימוש ב- form ajax – בוצע את הולידציה כפוף: 1. בצד הסרבר 2. סקריפט יידי בצד הקליינט.

תאור התהילך:

- המשמש הzin את כל הערכים הנדרשים ע"י וולידציית קלינט אך לא הzin נכון את המאפיין שנבדק ע"י וולידציה סרבר
- המשמש לחץ על כפתור submit
- הדף מגלה שככל השדות תקינים (בצד הקליינט) ולן מבצע post
- נרשם לאיורע submit בקלינט, נבדק שם את הולידציה באופן יידי, ובמקרה שהוא לא תקין נציג לו עם הודעת השגיאה ונחזיר false. לדוגמה:

```

$('form').submit(function () {
    var editor = $('#OfferModel_Description').data('tEditor');
    var upload = $('#attachmentFile').data('tUpload');

    if(editor.value() == "") {
        var errorMessage = "@(T("Offer.Description.Required"))" ;
        $("#messageForDescription").html(errorMessage);
        //Return false regardless of validation to stop form submitting
        return false;
    }
    else
    {
        $("#messageForDescription").html('');
        return true;
    }
});

```

- מtbody post לדף, הגענו ל-action שהגדרנו כיעד בפעולה ה-post
- הקוד ב-`action` בדק האם המודל ולידי (`ModelState.IsValid`), המודל אין ולידי כיון שהגדרנו ולידיית סרבר על המאפיין המדובר.
- ה-`action` שב את אותו ה-view
- ה-view אינו מתיחס למודל החדש שקיבל, הדף אינו מציר מחדש האינדיקציה שצירם בצד הקליינט באירוע submit מוצגת למשתמש

הசזרנות בפתרון זה:

- הוולידייזיה נכתבת באופן יידי בצד הקליינט
- לא מtbody שימוש בתשתיות המובנית לתצוגת האינדיקציה של השגיאה (לדוגמא: פקדן טריריך מציגים מסגרת אדומה ואך רקע אדום סביב הפקד שאינו ולידי)

היתרונות בפתרון זה:

- אין צורך לדאוג למלא שוב את הרשימות
- הדף אינו מציר שוב ולכן חווית המשתמש טובה יותר

2. שימוש ב-`html form` – משתמש בוולידייזיה סרבר ונדאג למילוי הרשימות בכל ביצוע postback.

תיאור התהילה:

- המסר מכיל בין השאר שתי רשימות מלאות אחת בשניה. בהזנת הארץ בקומו ארצות מתמאל קומו states עפ"י הארץ שנבחרה, המילוי מtbody/ajax.
- המשתמש הזין את כל העריכים הנדרשים ע"י ולידייזיות קלינט כולל הארץ ו-state, אך לא הזין כוון הארץ המופיע שנבדק ע"י ולידייזיות סרבר
- המשתמש לחץ על פקח submit
- מtbody post לדף כיון שכל השדות תקינים (בצד הקליינט), ה-`action` הרלוונטי נקרא הקוד ב-`action` בדק האם המודל ולידי (`ModelState.IsValid`), המודל אין ולידי כיון שהגדרנו ולידייזיות סרבר על המאפיין המדובר.
- ה-`action` מtbody שוב את אותו ה-view וודאג למלא שוב את קומו ארצות וקומו states עפ"י הארץ שהמשתמש בחר. (שני פקדים הקומו מוקורים למאפיינים ב- ViewData/ViewBag או למאפיינים במודל. בהעלאה ראשונית של הדף רשימת states-rijka (אך קיימת ב-`ViewData`). בביצוע post, מילאים את רשימת states עפ"י הנבחר בשדה הארץ).
- ה-view מציר מחדש כולל האינדיקציה של הוולידייזיה שהתבצעה בצד הסרבר.

הசזרנות בפתרון זה:

- במקרה שכל השדותקיימים חוץ מהשדה הנבדק בוולדיצית סרבר, מותבצע postback לדף, הדף מציר מחדש והרשימות התלויות צricsות להשלףשוב מה-DB – המשתמש ממתין יותר זמן.
- היתרונות בפתרון זה:**
- אין הכפלה של וולדיציות
 - ציור האידיומטיקה של הולידיציה מותבצעת ע"י התשתית של הפקדים ולא באופן ידני.

5.4. **וולדיציות סרבר על מס' מאפיינים מצומצם מתוך המודול כולם**

במקרה שמשמעותם לבצע וולדיציה ורק על חלק קטן מהמודול לדוגמה: קיים מודול המכיל שדות עם הרהբ ממד וולדיציות אך ישנו view שמציג את כל השדות לкриאה בלבד כאשר רק שני שדות הין שודוט לכתיבתן ומוכן שרק עליהם נרצה להפעיל את הולידיציה (חבל לבדוק שוב את המודול כולו שמיലא לא ניתן לעריכה ע"י המשמש), נוכל לייצר form סביב ואוטם שדות ושלוחם ב-Post כפרמטר תחת אובייקט שנבנה במיחוד לצורך זה שמכיל רק את השדות הרלוונטיים לבדיקה. ה-ActionController יקבע כפרמטר את התת מודול הקטן שבנינו יוכל להשתמש במאפיין ModelState.IsValid כדי לבדוק את תקינותו.

דוגמא:

הערה[6]: זהו לתה המודול שיופיע
המכיל את השדה UserQty הנדרש
לולדיציה

```
@using (Html.BeginForm("Buy", "Offer", new Nop.Web.Models.Offer.SubOffer() { UserQty = Model.UserQty })) {
    {
        <table border="1" style="border-collapse:collapse; border-style: solid; border-width: thin; border-color: Silver; width: 100%; background-color:#f8f8f8;">
            <tbody>
                <tr >
                    <td class="section-title" colspan="2" >
                        @T("Offer.PriceAndQuantity")
                    </td>
                </tr>
                <tr class="row" >
                    <td class="item-name">
                        @Html.TextBoxFor(x => x.Qty, new { style = "width:120px;" })
                        @T("Offer.Qty")&ampnbsp&ampnbsp&ampnbsp
                        @Html.TextBoxFor(x => x.UserQty, new { style = "width:120px;" })
                        @Html.ValidationMessageFor(model => model.UserQty)
                    </td>
                    <td style="width:20%">
                        <div id="lblQty"></div>
                    </td>
                </tr>
            </tbody>
        </table>
        <input type="submit" />
    }
}

[HttpPost]
public ActionResult Buy(SubOffer sub)
{
    if (ModelState.IsValid)
    {
    }
    else
    {
    }

    return RedirectToAction("BillingAddress", "Checkout");
}
```

הדגשה חשובה:

בד"כ אפשר להמנع מצב זה כיוון שניתן יותר לבנות ViewModel "יעדי עבור כל View וע"כ ה-ViewModel

היעדי לדף זה יכול אך ורק את הולידציות המדרשות בדף זה.

5.5. בעיות ידועות

5.5.1 בעיה בשימוש-b-FlunetValidation Data Annotations יחד עם FlunetValidation באוטומאמודל

בגדרת וolidity required של DataAnnotation תתקבל הודעה הבא:

Validation type names in unobtrusive client validation rules must be unique. The following validation type was seen more than once: required

הסיבה היא שהיא Flunet מיצרת באופן אוטומטי ולידייזיט required על כל שדה שאינו יכול לקבל null (לדוגמא שדה מסוג int) כך ששימוש בשניהם באותו מודול תוביל לכפילות של וolidity על אותו מאפיין.

הפתרון: לבצע את וolidity ה-`required` או לחלופין להמנע בשימוש בשניהם על אותו מודול.

6. ערך "בחר" בשדה קובמו

יש לנו ב-Helper DropDownList מכיל אפשרות להזנת טקסט "בחירה", הערך המתתקבל מהוורמה הינו הטקסט שזה לא אפשרhet לבעוד וolidity של Required על שדה הקובמו

דוגמא שימוש:

```
helper[10]: ד הינו שכתב באפליקציה ומוביל DB/cache את התרגולים מ-HTML
```

```
@(Html.DropDownList("CategoryId", Model.CategoryList, "Offer.Select").Text, new { style = "width:125px;" }))
```

7. בעיה בביבטו אונימציה של IE-jquery

תיקור הבעיה:

שימוש בפונקציית האונימציה של jquery, לדוגמה: slidedown, slidetoggle, fidgetoggle ווד, עובד מצוין ב-IE browsers שונים כגון google chrome, firefox ועוד, אך אינו עובד ב-IE

פתרונות:
IE לא יודע לבצע את האונימציה על תוכן המכיל הגדרה של position: absolute או position: relative. בוחרנו קוד זה מוחתוון שUMBOT צלוי האונימציה. במקרה שהתוכן מכיל שימוש בפקד צד שלישי, ניתן מחד שברינדור הפקד משתמשים ג"כ בהגדרת ה-`position:static`, יש לשנות את סקריפט CSS של אותו פקד. פתרון נוסף שאנשים מציעים אך לא נסה בניתוח Bairon: למתן `Height:101%` במקומ גובה אבסולוטי.

דוגמא לביבטו אונימציה לגיריד אשר אחת ל-10 שניות מחליף את תוכנו ומציג את סט הרשומות הבא:
ב-Body אנחנו מכינים מקום עבור הגיריד:

```
<div style="height: 250px; width:400px" id="partialRequests">  
    <div id="griddiv"/>  
    @Html.Action("HomePageOffersGrid", "Offer", new { ChooseId = 1 });  
    </div>  
</div>
```

בסקריפט אנחנו מתחילהם ט"י מאריך אחת ל-10 שניות מבצע שוב את הקראיה ל-`action` הנ"ל, והוא ב-`action` יודע להביא את סט הרשומות הבא. התוצאה מוצגת באונימציה:

```
setInterval(function () {  
  
    $('#griddiv').remove();  
  
    var $newdiv = $(document.createElement('div'));  
    $newdiv.attr("id", "griddiv");
```

```

$newdiv.hide();
$('#partialRequests').append($newdiv);

var url = "Offer/HomePageOffersGrid?ChooseId=1&random=" + Math.random() * 99999;
$newdiv.load(url, function (response, status, xhr) {

    if (status != "success") {
        $newdiv.html('an error has occurred');
    }
    else {
        $newdiv.slideToggle(4000);
    }
});
}, 10000);

```

8. פקדי טריריק

8.1. בעיה בвалиידציה עבור פקד Editor

הפקד אינו מפעיל את הwalidציה מצד הקליינט. הפתרון:

1. יש להוריד את הרישום האוטומטי של Jquery בדף המכיל את הפקד:

```

<script type="text/javascript">
    @(Html.Telerik().ScriptRegistrar()
        .jQuery(false)
    )
</script>

```

2. יש לבצע את הwalidציה באופן ידני מצד הקליינט באירוע submit. יש להחזיר false כדי לעצור את ה- submit (פעולה העצירה לא תעבור אם לא נבצע את הורדת הרישום כנ"ל)

```

$( '#form0' ).submit(function () {

```

```

    var editor = $('#OfferModel_Description').data('tEditor');
    var upload = $('#attachmentFile').data('tUpload');

    if(editor.value() == "") {
        var errorMessage = "@(T("Offer.Description.Required"))" ;
        $('#messageForDescription').html(errorMessage);
        //Return false regardless of validation to stop form submitting
        return false;
    }
    else {
        $('#messageForDescription').html('');
        return true;
    }
}

```

;});

3. אם לא ניתן להוריד את הרישום האוטומטי של jQuery בכלל פקדים אחרים הדורשים אותו, לא יהיה פתרון לביצוע הולדייצה בצד הקליינט. שני הפתרונות האפשריים מתוארים בסעיף 5.3 "ולדייצות המבוצעות בצד הסרבר".

8.2. בעיה בביצוע אнимציה (slides и slideToggle) על פקד Grid של Telerik.

האנימציה אינה עובדת ב-IE אך כן עובדת ב-browsers אחרים.
הפתרון: יש להוריד את ההתיחשות ל-position relative/absolute על פקד הגדרת סקריפט `newTelerikCommonStyle.css`. בבדיקה עם גריד די מנוון וلتצוגה בלבד היינו צריכים לתקן רק את הגדירה הכללית עבו הגדרה:
החלפנו את השורה הבאה:

```
/* Grid */.t-grid{position:relative;zoom:1}
```

לשורה זו:

```
/* Grid */.t-grid{zoom:1}
```

ראי פירוט בסעיף 7 על בעית האנימציה ב-IE.

9. הפונקציה load של jquery ajax מגיעה לשרת רק בפעם הראשונה - Ajax Cache

תיאור הבעיה:
הfonקציית load הינה פונקציה שמקבלת url ו吐וענת את התוצאה לתוך פקד על הדף. הבעיה היא שאם מדובר ב-URL קבוע, הקריאה בפועל ל-url תבצע רק בפעם הראשונה, לאחר מכן תשלוף התוצאה מה-cache של ajax.

פתרון:
פתרון פשוט אך עובל: לאgeom ל-URL שלא להיות קבוע, ניתן לבצע זאת ע"י שרשור של פרמטר במקבל מספר רנדומלי. לדוגמה:
`var url = "Offer/HomePageOffersGrid?random=" + Math.random() * 99999;`

```
$newdiv.load(url, function (response, status, xhr) {});
```

ניתן לחילופין לבטל את ה-cache של ajax:
`$(document).ready(function() { $.ajaxSetup({ cache: false }); })();`
אך צריך לשים לב שהוא מבטל את ה-cache עבור כל פעולות ajax בדף הנוכחי.
(ישנה גם אפשרות לבטל את ה-cache של ajax על קריאת ajax ספציפית אך האפשרות זמינה רק עבור הfonקציית ajax שמבצעת קריאה אסינכורונית לשרת):
`$.ajax({url: "myurl", success: myCallback, cache: false});`

מאתורי הקלעים ה-jquery מושרש פרמטר ל-url פרמטר של תאריך ושעה כדי לגרום ל-url להיות שונה בכל קרייה.
הבדל בין `$.ajaxSetup()` הינו ש-`$.ajaxSetup()` מאפשר להגדיר ערכי ברירות מחדל עבור כל פעולה ה-`ajax` הקיימות ואילו `$.ajax()` עובד על המופיע הספציפי.

.10

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using System.Web.Mvc;
6
7  namespace Lesson2.Controllers
8  {
9      public class FirstController : Controller
10     {
11         //-----Controller-----
12         // מחלקה המיוצרת עי מיקרוסופט שמרתת לתוך בינו הדפים לנתונים בינו
13         // המחלקה מכילה Actions על שם טיפוס הנתונים המוחזר ActionResult
14         // הפונקציות במחלקה יחזירו טיפוס זה או יורשים שלו //
15         //View זה מציין View
16         //-----View-----
17         //csHTML דפים עם סימת <%
18         //HTML, JavaScript, cs... דפים אלו מכילים, ...
19         // הקומpileר יתרגם את הדף לקוד HTML נקי ואנו הוא ישלח לדף //
20         //-----
21         // איז הדף מפעיל אקשין (פונקציות) בקונטולר ? איז הדף //
22         //1.URL-----(html)address/pagename.html (action) address/
23         // controllername/actionname .../First/Index
24         //2.LINK
25         //3.Submit
26
27         // ובכל אחד מהם איז בהרצה - השרת יوذע את מי להריץ ראשוני ? //
28         // אם יש באתר הרבה Controllers יש הרבה
29         // מוגדרים הגדירות Action Controller והן לשמות אותם נרצה להחbill //
30         // בתקייה App_Start בקובץ RoutConfig הניתוב - שם נשנה את שם ה
31         // איז יוצרים דף בקבילות מקום המתאים ? //
32         // עםידה על שם ה Action ללחוץ ימני Add View, חבצע את הפעלה //
33         public ActionResult Index()
34         {
35             // אם לא כתוב שם הדף לפתיחת Controller, השבתו תיקייה השמה שם
36             // הפונקציה תציג דף ששמו כשם האקשין ומיקומו בתקייה השמה שם
37             // ה
38             return View();
39         }
40         public ActionResult Hamara(double shkalim)
41         {
42             // במקום לפניו למסד הנתונים של בנק ישראל או לשרת של בנק //
43             // ישראל נמציא מידע כרגע
44             double Dolar = 3.5;
45             //----- איז מבירים מידע מהשרת ללקוח? -----
46             //1.ViewBag
47             // מילוני המנוהל עי מיקרוסופט וublisher נתונים //
48             // מהשרת לקוחות
49             // אנו ממציאות את שמות המשתנים ומערכות כאשר //
50             //key שמות המשתנים //
51             //value ערך המשתנים //
52             // מושב לשים לב שהוסף תקע לכל פונקציה בנפרד - לא מוכך //

```

```
    מפונקציה לפונקציה  
47     ViewBag.dolarim= shkalim / Dolar;  
48     // ממהר וטוענים את הדף מחדש נתוני המשתמש אובדים - יש לשמר גם  
     אותם בוין בג  
49     ViewBag.shkalim = shkalim;  
50     // אם רוצים שהפונקציה תחזיר לדף שונה משם הפונקציה - יש לציין //  
     את שמו  
51     return View("Index");  
52 }  
53 public ActionResult HomePage()  
54 {  
55     // אם רוצים דף שיושב בתיקייה של קונטROLLER אחר יש לציין גם את //  
     שם הקונטROLLER  
56     return View("Home/Index");  
57 }  
58 }  
59 }
```

```
1
2
3
4 <h2>המרת שקלים לדולרים</h2>
5 <form action="Hamara" >
6   <input type="text" name="shkalim" value="@ViewBag.shkalim" /> הכנס סכום
7   <br /> ב שקלים
8   <input type="submit" value="המר לדולרים" />
9 </form>
10 @ViewBag.dolarim
11
```

תרגילים

תרגיל 1

צרי מחשבון המרת מטבע:

- פקד לקליטת ערך המטבע
- תיבת בחירה (Select) לבחרת שער החליפין: דולר, שקל, יורו.
- לחץ "המרת" יציג את תוצאה ההמרה על אלמנט `م`.

הדרך:

שלב א - **הגדיר ב Controller קבועים** שיכילו את שער ההמרה, העזרי בהם לחישוב ההמרה.

שלב ב - הגדרי מחלוקת Model בשם Rate. **הגדיר במודל קבועים** שיכילו את שער ההמרה, השתמשי בהם לחישוב ההמרה.

תרגיל 2

צרי אתר המכיל את כל רכבי ארכיטקטורת MVC. האתר יאפשר להציג חישובים על מלון.

Model

צרי מחלוקת (סטטית) לחישוב גDAL מלון

פונקציות: חישוב שטח, חישוב היקף

הפונ' מקבלת את הנתונים לחישוב ומחזירה את התוצאה.

Controller

צרי את הפעולות הבאות:

פעולה המציגת את המשפט "ברוכים הבאים לאתר שלנו"

פעולה המקבלת פרמטרים מתאימים ומחזירה חישוב שטח או חישוב היקף

View

צרי וו המאפשר בחירת החישוב המתאים (תיבת Select או לחצני radio), והציגו.

תרגיל 3

צרי אתר להציג נתונים אודוט ישות, ארצות וערים.

- הצגי רשימת ישות בתיבת בחירה.
- בעת בחירת ישות תציג רשימת ארצות מתאימה בתיבת בחירה.
- בעת בחירת אرض תציג רשימת ערים מתאימה ברשימת `عن`.
- יש לדאוג שהערך הנבחר בתיבת הרשימה ישאר נבחר גם לאחר הפניה לשרת.

הדרך:

- יש ליצור מחלוקת סטטית אשר תכיל רשימות סטטיות של ישות, ערים וארצות.
- בעת בחירת פרט ברשימה, יש לפנות ל Action Shimלא את הרשימות הנוספות בהתאם.
- זכריו: נתונים ViewBag אינם נשמרים לאחר קריאה חוזרת ל Action.

② - server 6 3/7

url is now p3(1c) w/ 6
localhost:1234

link p3(2)
javascript p3(3)

http client - request client in http you are 6
3/7 - " diring

viewbag - view / controller in

Physical path 16, 10, 2, 33

IIIS - ~~an~~ ~~small~~ ~~sign~~
~~and~~ ~~pen~~

final fix for perl -> run -> netmgr
ITS | 2003-03-17

Application Pools - if idle longer than
sites → brown

Identity- f, even n.s.e. $\int_{-1+2\pi i}^{1+2\pi i}$ $\frac{dz}{z^2}$ $\int_{-\infty}^{\infty}$ $\frac{dt}{t^2 + 1}$

May 0-07 1921, 13³°F 1.2, 100³ 1.2 - wind 40

IIS - windows → IIS, IIS, op, IIS, Exchange - use

Apache-Linux

Bindling (Port)NIC
HTTP → WWW → Internet information services →
In port and FTP service in port
IIS 6

render, (3,5)

Nov. 21, 1955

Syntax/Sample	Razor	Web Forms Equivalent (or remarks)
Code Block	@{ int x = 123; string y = "because.;" }	<% int x = 123; string y = "because.;" %>
Expression (Html Encoded)	@model.Message	<%: model.Message %>
Expression (Unencoded)	 @Html.Raw(model.Message) 	<%= model.Message %>
Combining Text and markup	@foreach(var item in items) { @item.Prop }	<% foreach(var item in items) { %> <%: item.Prop %> <% } %>
Mixing code and Plain text	@if (foo) { <text>Plain Text</text> }	<% if (foo) { %> Plain Text <% } %>
Mixing code and plain text (alternate)	@if (foo) { @:Plain Text is @bar }	Same as above
Email Addresses	Hi philha@example.com	Razor recognizes basic email format and is smart enough not to treat the @ as a code delimiter
Explicit Expression	ISBN@(isbnNumber)	In this case, we need to be explicit about the expression by using parentheses.
Escaping the @ sign	In Razor, you use the @@foo to display the value of foo	@@ renders a single @ in the response.
Server side Comment	@* This is a server side multiline comment *@	<%-- This is a server side multiline comment --%>
Calling	@(MyClass.MyMethod<AType>()	Use parentheses to be explicit

generic method)	about what the expression is.
Creating a Razor Delegate	<pre>@{ Func<dynamic, object> b = @@item; } @b("Bold this")</pre>	Generates a Func<T, HelperResult> that you can call from within Razor. See this blog post for more details.
Mixing expressions and text	Hello @title. @name.	Hello <%: title %>. <%: name %>.

NEW IN RAZOR v2.0/ASP.NET MVC 4

Conditional attributes	<pre><div class="@className"></div></pre>	When className = null <div></div> When className = "" <div class=""></div> When className = "my-class" <div class="my-class"></div>
Conditional attributes with other literal values	<pre><div class="@className foo bar"> </div></pre>	When className = null <div class="foo bar"></div> <i>Notice the leading space in front of foo is removed.</i> When className = "my-class" <div class="my-class foo bar"> </div>
Conditional data-* attributes. <i>data-* attributes are always rendered.</i>	<pre><div data-x="@xpos"></div></pre>	When xpos = null or "" <div data-x=""></div> When xpos = "42" <div data- x="42"></div>
Boolean attributes	<pre><input type="checkbox" checked="@isChecked" /></pre>	When isChecked = true <input type="checkbox" checked="checked" /> When isChecked = false <input type="checkbox" checked="false" />

		/>
URL Resolution with tilde	<pre><script src="~/myscript.js"> </script></pre>	When the app is at / <pre><script src="/myscript.js"> </script></pre> When running in a virtual application named MyApp <pre><script src="/MyApp/myscript.j s"> </script></pre>

client side
server side

MVC

server → client מ Reply יתג

get → URL API URL ↗
" " " " (a href) link ↗

HTML file accept and do submit ↗

Request → API

URL need ? param = ... - Get ↗

Post ↗

client server do

return View()

View | controller - client | server do Reply API

view model, view base, view model near ViewBag ↗

model will pass view with view, and also for view Model - Model ↗

HTML file ↗

shape syntax many kind of API function to method API ↗

model ↗

request ↗

Server do logic work and client do ↗

Viewbag - Viewbag ↗ API ↗

Model - Model ↗ " "

if and if HTML tag if if - helper