آموزش KnockoutJS ـ آموزش value binding در KnockoutJS

هدف

value binding مقدار المان مربوطه ی DOM را در view model تان به یک مشخصه مرتبط می کند. این binding معمولا در form element هایی مانند <select>, <input> و <textarea> مفید است.

زمانی که کاربر در form control مربوطه این مقدار را ویرایش می کند، value binding مقدار موجود در view model شما را آپدیت می کند. همچنین وقتی شما این مقدار را در view model تان آپدیت کنید، این کار باعث آپدیت شدن مقدار form control در screen می شود.

نکته: در صورتی که با checkbox ها یا radio button ها کار میکنید، می توانید برای خواندن و نوشتن حالت check شده ی المان خود از checked binding و نه value binding استفاده کنید.

مثال:

```
<p>Login name: <input data-bind="value: userName" /></p>
<p>Password: <input type="password" data-bind="value: userPassword" /></p>

<script type="text/javascript">
  var viewModel = {
    userName: ko.observable(""),       // Initially blank
    userPassword: ko.observable("abc"), // Prepopulate
  };
</script>
```

پارامترها

- پارامتر اصلی

KnockoutJS محتوای مشخصه ی value المان را بر روی مقدار پارامتر شما قرار می دهد. تمامی مقادیر قبلی overwrite می شوند.

اگر این پارامتر مقدار قابل مشاهده ای باشد، در این صورت binding هر زمان که این مقدار تغییر کند مقدار المان را آپدیت می کند. اما اگر این پارامتر قابل مشاهده نباشد، تنها یک بار مقدار المان را تنظیم می کند و دیگر آن را آپدیت نخواهد کرد.

اگر شما چیزی غیر از عدد یا رشته (برای مثال شیء یا آرایه) را در کدتان قرار دهید، در این صورت متن نمایش داده شده با yourParameter.toString() برابری می کند (استفاده از مقادیر غیرعددی یا غیر رشته ای اغلب چندان مفید نیست به همین دلیل بهترین کار استفاده از مقادیر رشته ای و عددی است).
هر زمان که کاربر مقدار را در form control مربوطه ویرایش می کند، KnockoutJS مشخصه ی آن را در view model شما آپدیت می کند. زمانی که value تغییر می کند و کاربر focus را بر روی گره ی دیگری از DOM انتقال می دهد (یعنی بر change event) KnockoutJS همواره سعی خواهد کرد تا view model شما را آپدیت کند. اما شما می توانید با استفاده از پارامتر valueUpdate که در زیر توضیح داده شده است بر اساس رویدادهای دیگر به روز رسانی را فعال کنید.

- پارامترهای اضافی
  - valueUpdate

اگر پارامتری به نام valueUpdate در binding شما وجود داشته باشد، در این صورت کار این پارامتر تعریف کردن رویدادهای اضافی در مرورگر است که KnockoutJS باید به غیر از change event برای شناسایی تغییرات از آن استفاده کند. مقادیر رشته ای زیر بیشترین کاربرد را در این پارامتر دارند.

- "input": زمانی که مقدار المان <input> یا <textarea> تغییر می کند، view model شما را آپدیت می کند. توجه داشته باشید که این رویداد تنها در مرورگرهایی پشتیبانی می شود که تا حد زیادی به روز باشد (مثلا IE ورژن 9 و بالاتر).
- "keyup": زمانی که کاربر کلیدی را رها کند view model شما را آپدیت می کند.
- "keypress": زمانی که کاربر کلیدی را فشار دهد view model شما را آپدیت می کند. این رویداد برخلاف keyup تا زمانی که کاربر دستش را بر روی کلید نگه دارد به صورت مکرر آپدیت می شود.
- afterkeydown: به محض اینکه کاربر شروع به تایپ کاراکتری بکند، view model شما را آپدیت می کند. این عمل با گرفتن keydown event مرورگر و handle کردن این رویداد به صورت ناهماهنگ میسر می شود. این رویداد در برخی از مرورگرهای تلفن همراه پشتیبانی نمی شود.
- valueAllowUnset

به نکته ی 2 در پایین مراجعه کنید. توجه داشته باشید که valueAllowUnset تنها وقتی کاربردی است که برای کنترل کردن selection در یک المان <select> استفاده شود. valueAllowUnset بر المان های دیگر اثری نمیگذارد.

آموزش KnockoutJS ـ آموزش دریافت فوری value update ها از ورودی ها

اگر می خواهید برای دریافت آپدیت های فوری در view model تان <input type="text" /> یا <textarea> را bind کنید، در این صورت از textInput binding استفاده کنید. این binding در اغلب مرورگرها بهتر از valueUpdate option های دیگر پشتیبانی می شود.

آموزش KnockoutJS ـ آموزش کار با drop-down لیست ها (المان های <select>)

KnockoutJS به خوبی این نوع از لیست ها را پشتیبانی می کند. value binding در کنار options binding کار می کند تا برای شما این امکان فراهم شود که نه فقط مقادیر رشته ای بلکه مقادیری که اشیاء دلخواه جاوااسکریپت هستند را نیز بخوانید و بنویسید. اگر بخواهید که این امکان را برای کاربر فراهم کنید تا از بین مجموعه ای از model object ها یکی را انتخاب کند، این binding کمک زیادی به شما می کند. برای مشاهده ی مثال این مطلب به options binding کنید. یا برای مدیریت لیست های چند انتخابی (multi-select lists) به selectedOptions binding مراجعه کنید.

همچنین می توانید در کنار المان <select> از value binding هم استفاده کنید، در المان <select> از options binding استفاده نشده است. در این حالت می توانید یا المان های <option> خود را در markup تعیین کنید یا اینکه با استفاده از template binding یا foreach binding آن ها را بسازید. حتی می توانید داخل المان های <optgroup> option ها را به صورت تو در تو استفاده کنید و خود KnockoutJS مقدار انتخابی مناسب را تنظیم خواهد کرد.

آموزش KnockoutJS ـ آموزش استفاده از valueAllowUnset در کنار المان های <select>

به طور معمول وقتی که در المان <select> از value binding استفاده می کنید، معنی آن این است که شما از مقدار مدل مربوطه می خواهید تا مشخص کند که کدام آیتم در <select> انتخاب شده است. اما در صورتی که مقدار مدل را بر روی چیزی قرار دهید که در لیست هیچ ورودی متناظری برای آن وجود نداشته باشد، چه اتفاقی خواهد افتاد؟ در این صورت رفتار پیش فرض KnockoutJS overwrite کردن مقدار مدل شما است تا آن را به حالتی ریست کند که پیش از این در dropdown انتخاب شده است. به همین ترتیب از ایجاد ناهماهنگی در رابط کاربری و مدل جلوگیری می شود.

با این حال ممکن است که شما این رفتار را نپسندید. اگر به جای حالت بالا می خواهید که KnockoutJS این اجازه را به model observable شما بدهد که مقادیری را دریافت کند که در <select> هیچ مقدار متناظری برای آن وجود نداشته باشد، بایستی valueAllowUnset: true را تعیین کنید. در این صورت هر زمان که مقدار مدل شما را نتوان در <select> ارائه کرد، <select> به سادگی در آن زمان هیچ مقدار انتخاب شده ای ندارد که نتیجه را به صورت جای خالی نمایش می دهد. وقتی که در آینده کاربری ورودی ای را از dropdown انتخاب می کند، طبق معمول این جای خالی در مدل شما نوشته خواهد شد. برای مثال :

```
<p>
  Select a country:
  <select data-bind="options: countries,
             optionsCaption: 'Choose one...',
```

```
            value: selectedCountry,
            valueAllowUnset: true"></select>
</p>

<script type="text/javascript">
  var viewModel = {
    countries: ['Japan', 'Bolivia', 'New Zealand'],
    selectedCountry: ko.observable('Latvia')
  };
</script>
```

در مثال بالا selectedCountry مقدار 'Latvia' را نگهداری می کند و به این دلیل که option متناظری در dropdown وجود ندارد، در این صورت dropdown خالی خواهد بود.

اگر valueAllowUnset فعال نشده باشد، KnockoutJS selectedCountry را با undefined ، overwrite می کند. به گونه ای که با مقدار ورودی 'Choose one...' مطابقت داشته باشد.

آموزش KnockoutJS ـ آموزش به روز کردن مقادیر مشخصه های قابل مشاهده و غیرقابل مشاهده

اگر برای اینکه form element ای را به یک مشخصه ی قابل مشاهده وصل کنید از value استفاده کنید، KnockoutJS می تواند یک binding دو طرفه ای ایجاد کند به گونه ای که تغییرات بر یکدیگر اثرگذار باشد.

با این حال اگر برای متصل کردن form element ای به یک مشخصه ی قابل مشاهده (مثلا یک رشته ی ساده ی قدیمی یا یک عبارت دلخواه جاوا اسکریپت) از value استفاده کنید، دراین صورت KnockoutJS مانند زیر عمل می کند:

- اگر شما به مشخصه ی ساده ای اشاره کنید (یعنی این مشخصه تنها یک مشخصه ی معمولی در view model شماست) در این صورت KnockoutJS حالت اولیه ی form element ها را بر روی مقدار مشخصه قرار می دهد و زمانی که این form element ویرایش شود، KnockoutJS تغییرات را به صورت نوشتاری به مشخصه ی شما برگشت می دهد. KnockoutJS نمی تواند تشخیص دهد که چه موقعی مشخصه ی شما تغییر می کند (زیرا این مشخصه قابل مشاهده نیست) به همین دلیل این binding یک binding یک طرفه است.
- اگر بخواهید به یک مشخصه ی غیرساده اشاره کنید (مثلا نتیجه ی فراخوانی یک تابع یا عملیات مقایسه) در این صورت KnockoutJS حالت اولیه ی form element را بر روی آن مقدار قرار می دهد. اما وقتی کاربر form element را ویرایش می کند، KnockoutJS نمی تواند تغییری write کند. در این حالت مقدار تنها یک بار تنظیم می شود و binding به صورت پیوسته به تغییرات واکنش نشان نمی دهد.

مثال:

```
<!-- Two-way binding. Populates textbox; syncs both ways. -->
<p>First value: <input data-bind="value: firstValue" /></p>

<!-- One-way binding. Populates textbox; syncs only from textbox to model. -->
<p>Second value: <input data-bind="value: secondValue" /></p>

<!-- No binding. Populates textbox, but doesn't react to any changes. -->
<p>Third value: <input data-bind="value: secondValue.length > 8" /></p>

<script type="text/javascript">
  var viewModel = {
    firstValue: ko.observable("hello"), // Observable
    secondValue: "hello, again"       // Not observable
  };
</script>
```

آموزش KnockoutJS ـ آموزش استفاده از value binding در کنار checked binding

checked binding بهتر است که برای مقید کردن یک view model property در برابر مقدار یک checkbox (<input type='checkbox'>) یا radio button (<input type='radio'>) استفاده شود. اگر شما value binding را در یکی از این المان ها در کنار checked binding استفاده کنید، در این صورت value binding کار option checkedValue را انجام خواهد داد. از value binding می توان در کنار checked binding استفاده کرد و از آن در کنترل کردن مقداری که برای آپدیت کردن view model شما استفاده می شود می توان بهره برد.

آموزش KnockoutJS ـ آموزش ایجاد تعامل با جی کوئری

جی کوئری در صورتی که موجود باشد، KnockoutJS از آن برای مدیریت رویدادهای رابط کاربری مانند change استفاده می کند. برای این که این رفتار را غیرفعال کنید و به KnockoutJS بفهمانید که همیشه از event handling های بومی استفاده کند، می توانید پیش از فراخوانی ko.applyBindings آپشن زیر را در کدتان قرار دهید:

```
ko.options.useOnlyNativeEvents = true;
```

وابستگی ها

به غیر از کتابخانه ی اصلی KnockoutJS وابستگی دیگری ندارد.