

Templates, Control of Flow, and Containerless Bindings

John Papa

@john_papa

<http://johnpapa.net>



pluralsight
hardcore developer training

Outline

- **Named Templates**
- **Control of Flow**
- **Binding Contexts**
- **Inline Templates**
- **Dynamically Choosing a Template**
- **Template Binding Helpers**
- **Containerless Bindings**

Templates

- **Re-use code**
- **Encapsulate responsibility for a specific rendering**
- **Knockout supports many popular templating engines**
 - jQuery Templates
 - Underscore
- **Knockout has native templates**

Named Templates in <script> tags

- Encapsulate a template for re-use

```
<div data-bind="template: {name: 'personTpl1'}"></div>

<script type="text/html" id="personTpl1">
  <span data-bind="text: firstName"></span>
  <span data-bind="text: lastName"></span>
  <button data-bind="click:selectPerson">Add</button>
</script>
```

DEMO

Named Templates



Outline

- **Named Templates**
- **Control of Flow**
- **Binding Contexts**
- **Inline Templates**
- **Dynamically Choosing a Template**
- **Template Binding Helpers**
- **Containerless Bindings**

Control of Flow

if

- If truthy condition

ifnot

- If falsy condition

foreach

- Execute for each item in a list

with

- Shortcut to execute for the object

Control of Flow with a Template

- Pass the context for the template with “foreach”

```
<tbody
  data-bind="template: {name: 'productsTmpl', foreach: lines}">
</tbody>

<script type="text/html" id="productsTmpl">
  <tr>
    <td style="width: 100px;">
      <input data-bind="value: quantity" />
    </td>
    ...
  </tr>
</script>
```


Conditional Control of Flow

Any “truthy” expression

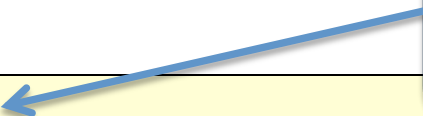


```
<p data-bind="if: lines().length > 0">  
  <span>Total value:</span>  
  <span data-bind="text: grandTotal()"></span>  
</p>
```

Change Context “with” Control of Flow

```
<div>  
  <div data-bind="text: model().brand"></div>  
  <div data-bind="text: model().name"></div>  
</div>
```

Change the context
with “with”



```
<div data-bind="with: model">  
  <div data-bind="text: brand"></div>  
  <div data-bind="text: name"></div>  
</div>
```

Parent Binding Contexts

- **Sometimes in templates you want to change data binding scope (Data Context)**
 - \$data
 - \$parent
 - \$parents
 - \$root

```
<button data-bind="click: $parent.addItem">Add</button>
```

DEMO

Control of Flow and Binding Contexts



Outline

- **Named Templates**
- **Control of Flow**
- **Binding Contexts**
- **Inline Templates**
- **Dynamically Choosing a Template**
- **Template Binding Helpers**
- **Containerless Bindings**

Inline Templates with Control of Flow

- If not reusing it, there is no need to name a template
- Control of flow elements create an implicit template

```
<tbody data-bind="foreach: lines">  
  <tr>  
    <td style="width: 100px;">  
      <input data-bind="value: qty"/>  
    </td>  
    ...  
  </tr>  
</tbody>
```

Template is
created
anonymously
and implicitly

Knockout's Native Template Engine

- **Templates inside DOM elements**

- `<script>`
- Other DOM elements like `<div>`

- **Anonymous / Inline templates**

- Templates without a name
- Shortcuts to Anonymous template binding
 - `if`
 - `ifnot`
 - `with`
 - `foreach`



All Part of the
Native Template Engine
in Knockout

No external
templating dependency

if and ifnot

```
<div data-bind="template: {if: isSelected}">  
  <span data-bind="text:name"></span>  
</div>
```

```
<div data-bind="if: isSelected">  
  <span data-bind="text:name"></span>  
</div>
```

“if” shortcut

```
<div data-bind="template: {ifnot: isSelected}">  
  <span data-bind="text:name"></span>  
</div>
```

```
<div data-bind="ifnot: isSelected">  
  <span data-bind="text:name"></span>  
</div>
```

“ifnot” shortcut

foreach and with

```
<div data-bind="template: {foreach: products}">  
  <span data-bind="text:name"></span>  
</div>
```

```
<div data-bind="foreach: products">  
  <span data-bind="text:name"></span>  
</div>
```

“foreach”
shortcut

```
<div data-bind="template:  
  {if: selectedProduct, data: selectedProduct}">  
  <span data-bind="text:name"></span>  
</div>
```

```
<div data-bind="with: selectedProduct">  
  <span data-bind="text:name"></span>  
</div>
```

“with”
shortcut

DEMO

Native Template Engine: Inline Templates



Outline

- **Named Templates**
- **Control of Flow**
- **Binding Contexts**
- **Inline Templates**
- **Dynamically Choosing a Template**
- **Template Binding Helpers**
- **Containerless Bindings**

Dynamically Change Templates

- Swap between multiple templates
- Bind the name of the template

```
<div data-bind="template: {name: templateChoice}">
<script type="text/html" id="tmplSummary">
    ...
</script>
<script type="text/html" id="tmplDetails">
    ...
</script>
```

```
my.vm.templateChoice = function () {
    return showDetails() ? "tmplDetails" : "tmplSummary";
};
```

Control of Flow to Toggle Templates

- if and ifnot bindings

```
<div data-bind="ifnot: showDetails()">
    ...
</div>
<div data-bind="if: showDetails()">
    ...
</div>
```

```
my.vm.showDetails = ko.observable(false);
```

DEMO

Dynamically Choosing a Template



Outline

- **Named Templates**
- **Control of Flow**
- **Binding Contexts**
- **Inline Templates**
- **Dynamically Choosing a Template**
- **Template Binding Helpers**
- **Containerless Bindings**

Template Bindings

name

Id of an element that contains the template

foreach

Renders the template in foreach mode
(once for each item)

data

Object to supply as data for the template.
If omitted, uses foreach context or the current context

afterRender

Callback invoked after DOM elements are rendered

afterAdd

Callback invoked after DOM elements are added

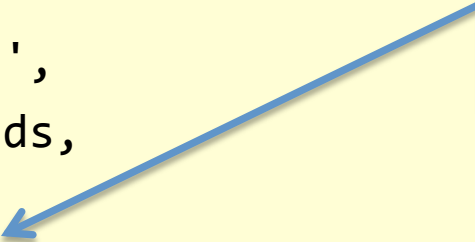
beforeRemove

Callback invoked before DOM elements are removed

Template Binding Helpers

```
<ul data-bind="template: {  
  name: 'friendsTemplate',  
  foreach: model().Friends,  
  beforeRemove: showAni,  
  afterAdd: hideAni}">  
</ul>
```

Callback to a method
in the viewmodel



DEMO

Template Binding Helpers



Outline

- **Named Templates**
- **Control of Flow**
- **Binding Contexts**
- **Inline Templates**
- **Dynamically Choosing a Template**
- **Template Binding Helpers**
- **Containerless Bindings**

Containerless Control of Flow Bindings

- **Comment syntax**
 - Unlike traditional Javascript template in <script>
- **Use a template, without having a template!**
 - What?!
 - He's nuts!
- **Comment based control flow syntax**
 - if
 - ifnot
 - foreach
 - with
 - template

All Part of the
Native Template Engine
in Knockout

Containerless Examples

```
<!-- ko with: selectedPerson -->
  <span data-bind="text: name"></span>
  <input data-bind="value: salary"></input>
<!-- /ko -->
```

Reduces Unneeded Elements

```
<ul>
  <li class="category">Acoustic Guitars<li>
    <!-- ko foreach:acousticProducts -->
      <li>
        <span data-bind="text: shortDesc"></span>
      </li>
    <!-- /ko -->
  </ul>
```

Moves binding logic outside of elements

DEMO

Containerless/Comment Bindings



Summary

- **Native Template Engine**

- Named Templates and Anonymous / Inline Templates

- **Control of Flow**

- if, ifnot, with, foreach, template

- **Binding Contexts**

- \$data, \$parent, \$parents, \$root

- **Dynamically Choosing a Template**

- template: {name: myObservableProperty}

- **Template Binding Helpers**

- name, data, foreach, afterRender, afterAdd, beforeRemove

- **Containerless/Comment Bindings**

- <!-- ko foreach: products -->

- **External Templates**

- <https://github.com/ifandelse/Knockout.js-External-Template-Engine>

For more in-depth **online** developer **training** visit



on-demand content from authors you **trust**