主要内容

Main Contents

1. KnockoutJS简介

Two-way binding

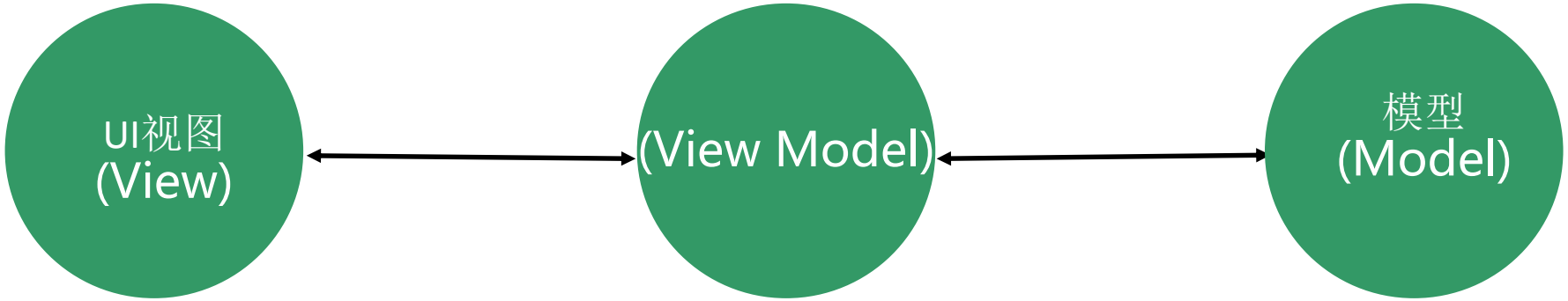
- Declarative Bindings
- Automatic UI Refresh

Dependency Tracking

Trivially extensible

2. 双向的数据绑定

3. MVVM



4. KnockoutJS的引入

01

```
<script type="text/javascript" src="../../../libs/knockout.js"></script>
<script>
var viewModel = {
            attr:ko.observable()
};
</script>
```

02

```
<input class="form-control" data-bind= "value:attr , style:{color:'red'}" />
```

03

```
ko.applyBindings(viewModel);
```

5. KnockoutJS Bindings

Binding Handler	Function/Expression
visible /enable /if /ifnot /disable /checked /hasFocus /uniqueName	true/false; ko.observable(true); array.length>0
foreach /options /selectedOptions	ko.observableArray(); Array
click /submit	function(){}
value /textInput	String; ko.observable(‘’)
style /event /attr /css	{attrName:function(){}} ; {attrName:expression}
template	{name:’templateId’,data:{}}
html /text	String

6. KnockoutJS Functions And Libraries

Function/Library Name	Extensions
ko.observable(); ko.observableArray(); ko.computed(); ko.pureComputed()	subscribe() > dispose(); extend
ko.applyBindings()	ko.applyBindingsToDescendants; ko.applyBindingsToNode
ko.toJSON(); ko.toJS(); ko.unwarp(); ko.isObservable()	
ko.utils	unwrapObservable(); compareArrays() …
ko.components	register(); isRegistered(); register(); $b (all components)

ko.bindingHandlers	if;anable;…
	ko.bindingHandlers.myHandler = {
	init: function(){},
	update: function(){}
	}

ko.subscribable	ko.subscribable.fn['track']
ko.extenders	ko.extenders.myExtension = function(target,params){}
ko[‘funcName’]	ko[‘funcName’] = function(){}

7. Knockout Context Objects

$root
$component

$data
$context
$context.$data

$parent
$parentContext
$parentContext.$data

8. Knockout Custom Bindings

01

Definition

```
ko.bindingHandlers.myHandler = {

    init:function(element,valueAccessor,allBindings, viewModel, bindingContext){

    },
    update:function(element,valueAccessor,allBindings, viewModel, bindingContext){

    }
};
```

02

Applying

```
<div class="container" data-bind="myHandler: expression"></div>
```

9. Knockout Custom Components

01

Definition

```
function createViewModel(){
    var vm = function(params){
        var self = this;
        self.attr = params.attr;
    };
    return vm;
}
function createTemplate(){
    return "<div data-bind='value:attr'></div>";
}
ko.components.register('my-component',{
    viewModel:createViewModel(),
    template:createTemplate()
});
```

02

Applying

```
<my-component params=" {attr:''} "></my-component>
```

10. Knockout Custom Functions

01 Definition

```
ko.observableArray.fn.filterByProperty =
function(prop,val){
  return this().filter(function(item){
     return item[prop] === val;
  });
};
```

02 Applying

```
var content = ko.observableArray();
var newContent = content.filterByProperty(propName,val);
```

11. Knockout Observable Extensions

01 Definition

```
ko.extenders.myExtension = function(target,params){

}
```

02 Applying

```
viewModel.attr = ko.observable('').extend(
                myExtension: {

                }
);
```

12. More

- KnockoutJS official web site : http://knockoutjs.com/
- Mapping Plugin : KOMapper(http://knockoutjs.com/documentation/plugins-mapping.html)