## 使用Repetition组件

Repetition组件的作用是根据数组提供的数据重复显示一组元素，Repetition对数据的每次循环会生成一个元素。Repetition的数据通过一个控制器管理。你可以手动对Repetition设置简单的数组数据，或者通过设置Repetition的contentController属性为RangeController来处理复杂类型的数据。

你可以使用Repetition来循环显示任意数量的其他组件。（比如MontageJS List组件，它就是使用Repetition实现的）使用可绑定的iteration.object模版属性表示当前的List项。

## 简单的Repetition

下面的例子展示了一个简单的Repetition组件的使用方式。它显示3个元素。每次循环都将从Repetition的content属性中取出一个值，根据这个值渲染当前重复元素。

<div data-montage-id="content" class="Content">  
 <ul data-montage-id="items">  
 <li data-montage-id="item"></li>  
 </ul>  
</div>

{  
 "owner": {  
 "properties": {  
 "element": { "#": "content" }  
 }  
 },  
 "items": {  
 "prototype": "montage/ui/repetition.reel",  
 "properties": {  
 "element": { "#": "items" }  
 },  
 "bindings": {  
 "content": { "<-": "@owner.myListProperty" }  
 }  
 },  
 "item": {  
 "prototype": "montage/ui/text.reel",  
 "properties": {  
 "element": { "#": "item" }  
 },  
 "bindings": {  
 "value": { "<-": "@items:iteration.object.quote" },  
 "classList.has('highlight')": { "<-": "@items:iteration.object.important" }  
 }  
 }  
}

.highlight {  
 font-weight: bold;  
}

var Component = require("montage/ui/component").Component;  
  
exports.Content = Component.specialize({  
 myListProperty: {  
 value: [{  
 "quote": "If music be the food of love, play on.",  
 "important": false  
 }, {  
 "quote": "O Romeo, Romeo! wherefore art thou Romeo?",  
 "important": true  
 }, {  
 "quote": "All that glitters is not gold.",  
 "important": false  
 }, {  
 "quote": "I am amazed and know not what to say.",  
 "important": false  
 }]  
 }  
});

结果请查看[Mfiddle](http://montagejs.github.io/mfiddle/#!/7882151)

## 使用RangeController设置Repetition

在这个例子中：

* RangeController被赋值给了Repetition的contentController属性用于管理Repetition数据。
* Text组件是Repetition的子组件，它的value值来自iteration.object属性。
* 点击Change Content按钮来替换Repetiion的数据为新的随机内容。

<div data-montage-id="component">  
 <button data-montage-id="button"></button>  
 <ul data-montage-id="repetition">  
 <li data-montage-id="value"></li>  
 </ul>  
</div>

{  
 "owner": {  
 "properties": {  
 "element": {"#": "component"}  
 }  
 },  
 "repetition": {  
 "prototype": "montage/ui/repetition.reel",  
 "properties": {  
 "element": {"#": "repetition"},  
 "contentController": {"@": "rangeController"}  
 }  
 },  
 "rangeController": {  
 "prototype": "montage/core/range-controller"  
 },  
 "value": {  
 "prototype": "montage/ui/text.reel",  
 "properties": {  
 "element": {"#": "value"}  
 },  
 "bindings": {  
 "value": {"<-": "@repetition:iteration.object.quote"}  
 }  
 },  
 "changeButton": {  
 "prototype": "digit/ui/button.reel",  
 "properties": {  
 "element": {"#": "button"},  
 "label": "Change Content"  
 },  
 "listeners": [  
 {  
 "type": "action",  
 "listener": {"@": "owner"}  
 }  
 ]  
 }  
}

var Component = require("montage/ui/component").Component;  
  
exports.Owner = Component.specialize({  
 constructor: {  
 value: function Owner() {  
 this.content = [{  
 "quote": "If music be the food of love, play on.",  
 "important": false  
 }, {  
 "quote": "O Romeo, Romeo! wherefore art thou Romeo?",  
 "important": true  
 }, {  
 "quote": "All that glitters is not gold.",  
 "important": false  
 }, {  
 "quote": "I am amazed and know not what to say.",  
 "important": false  
 }];  
 }  
 },  
  
 templateDidLoad: {  
 value: function () {  
 this.templateObjects.rangeController.content = this.content;  
 }  
 },  
  
 handleChangeButtonAction: {  
 value: function (evt) {  
 var randomContentIndex = Math.floor(Math.random() \* this.content.length);  
 this.templateObjects.rangeController.add(this.content[randomContentIndex]);  
 }  
 }  
});

点击[Middle](http://montagejs.github.io/mfiddle/#!/7883458)查看结果。

## Repetition中排序和过滤的用法

你可以使用FRB表达式对Repetition关联的RangeController的数据进行排序或者过滤。

* 对数据进行过滤。
* 当数据的filterPath值是false，这个数据不会显示在Repetiiton中。
* 使用sortPath对数据进行排序。

你当然可以使用复杂的表达式对数据进行排序和过滤，举个例子，Repetition的数据里有一个index属性是数字类型，我们可以使用!(index%2)来过滤掉奇数。

<div data-montage-id="component">  
 <button data-montage-id="filterButton"></button>  
 <button data-montage-id="sortButton"></button>  
  
 <ul data-montage-id="repetition">  
 <li data-montage-id="quote"></li>  
 </ul>  
</div>

{  
 "owner": {  
 "properties": {  
 "element": {"#": "component"}  
 }  
 },  
 "repetition": {  
 "prototype": "montage/ui/repetition.reel",  
 "properties": {  
 "element": {"#": "repetition"},  
 "contentController": {"@": "rangeController"}  
 }  
 },  
 "rangeController": {  
 "prototype": "montage/core/range-controller"  
 },  
  
 "quote": {  
 "prototype": "montage/ui/text.reel",  
 "properties": {  
 "element": {"#": "quote"}  
 },  
 "bindings": {  
 "value": {"<-": "@repetition:iteration.object.quote"}  
 }  
 },  
  
 "filterButton": {  
 "prototype": "digit/ui/button.reel",  
 "properties": {  
 "element": {"#": "filterButton"},  
 "label": "Filter"  
 },  
 "listeners": [  
 {  
 "type": "action",  
 "listener": {"@": "owner"}  
 }  
 ]  
 },  
  
 "sortButton": {  
 "prototype": "digit/ui/button.reel",  
 "properties": {  
 "element": {"#": "sortButton"},  
 "label": "Sort"  
 },  
 "listeners": [  
 {  
 "type": "action",  
 "listener": {"@": "owner"}  
 }  
 ]  
 }  
}

var Component = require("montage/ui/component").Component;  
  
exports.Owner = Component.specialize({  
 constructor: {  
 value: function Owner() {  
 this.content = [{  
 "quote": "If music be the food of love, play on.",  
 "important": false  
 }, {  
 "quote": "O Romeo, Romeo! wherefore art thou Romeo?",  
 "important": true  
 }, {  
 "quote": "All that glitters is not gold.",  
 "important": false  
 }, {  
 "quote": "I am amazed and know not what to say.",  
 "important": false  
 }];  
 }  
 },  
  
 templateDidLoad: {  
 value: function () {  
 this.templateObjects.rangeController.content = this.content;  
 }  
 },  
  
 handleFilterButtonAction: {  
 value: function () {  
 var rangeController = this.templateObjects.rangeController;  
 // toggle filterPath to either filter by "important" key or not filter  
 rangeController.filterPath = rangeController.filterPath ? "" : "important";  
 }  
 },  
  
 handleSortButtonAction: {  
 value: function () {  
 var rangeController = this.templateObjects.rangeController;  
 // toggle sortPath to either filter by "quote" key or not filter  
 rangeController.sortPath = rangeController.sortPath ? "" : "quote" ;  
 }  
 }  
});

点击[Mfiddle](http://montagejs.github.io/mfiddle/#!/7884201)查看结果。

RangeController使用FRB来[过滤](http://documentup.com/montagejs/frb/#tutorial/filter)和[排序](http://documentup.com/montagejs/frb/#tutorial/sorted)。你可以在绑定表达始中使用FRB函数。比如：

{  
 "bindings": {  
 "filteredEvens": {"<-": "numbers.filter{!(%2)}"},  
 "sorted": {"<-": "numbers.sorted{}"}  
 }  
}

## 允许用户选择Repetiton循环生成的元素

你需要进行以下设置来开启Repetition的"选择"功能：

* 设置Repetition的isSelectionEnabled属性为true。
* selectedCss规则会自动被添加给选中的元素。

注意用户是可以同时选择多个元素的。

<div data-montage-id="component">  
 <div data-montage-id="repetition">  
 <p data-montage-id="value"></p>  
 </div>  
 <select data-montage-id="select"></select>  
 <p data-montage-id="log"></p>  
</div>

{  
 "owner": {  
 "properties": {  
 "element": {"#": "component"}  
 }  
 },  
 "repetition": {  
 "prototype": "montage/ui/repetition.reel",  
 "properties": {  
 "element": {"#": "repetition"},  
 "isSelectionEnabled": true,  
 "contentController": {"@": "rangeController"}  
 }  
 },  
 "rangeController": {  
 "prototype": "montage/core/range-controller",  
 "properties": {  
 "selection": []  
 }  
 },  
  
 "value": {  
 "prototype": "montage/ui/text.reel",  
 "properties": {  
 "element": {"#": "value"}  
 },  
 "bindings": {  
 "value": {"<-": "@repetition:iteration.object.quote"}  
 }  
 },  
  
 "select": {  
 "prototype": "digit/ui/select.reel",  
 "properties": {  
 "element": {"#": "select"},  
 "contentController": {"@": "rangeController"},  
 "labelPropertyName": "quote"  
 }  
 },  
 "log": {  
 "prototype": "montage/ui/text.reel",  
 "properties": {  
 "element": {"#": "log"}  
 },  
 "bindings": {  
 "value": {"<-": "@owner.message"}  
 }  
 }  
}

.selected {  
 color:red;  
}

var Component = require("montage/ui/component").Component;  
  
exports.Owner = Component.specialize({  
 constructor: {  
 value: function Owner() {  
 this.content = [{  
 "quote": "If music be the food of love, play on.",  
 "important": false  
 }, {  
 "quote": "O Romeo, Romeo! wherefore art thou Romeo?",  
 "important": true  
 }, {  
 "quote": "All that glitters is not gold.",  
 "important": false  
 }, {  
 "quote": "I am amazed and know not what to say.",  
 "important": false  
 }];  
 }  
 },  
  
 templateDidLoad: {  
 value: function () {  
 this.templateObjects.rangeController.content = this.content;  
 //Observe the selection for changes  
 this.templateObjects.rangeController.addRangeAtPathChangeListener(  
 "selection", this, "handleSelectionChange");  
 }  
 },  
  
 handleSelectionChange: {  
 value: function (plus, minus) {  
 this.message = "Selection changed from: "  
 + (minus[0] ? minus[0].quote : "nothing")  
 + " -> "  
 + (plus[0] ? plus[0].quote : "nothing");  
 }  
 }  
});

点击[Mfiddle](http://montagejs.github.io/mfiddle/#!/7884716)查看结果。