**AngularJS**

目次

[概要 2](#_Toc515559572)

[基本的な実装 2](#_Toc515559573)

[コントローラーの使用 3](#_Toc515559574)

[フィルター 4](#_Toc515559575)

[ループで使用できる変数 5](#_Toc515559576)

[イベントリスナーの登録 5](#_Toc515559577)

[フォームの基本 7](#_Toc515559578)

[基本 7](#_Toc515559579)

[様々なフォーム要素の実装 8](#_Toc515559580)

# 概要

* JavaScriptのMVCフレームワーク
* 公式サイト：「<https://angular.io/>」
* 必要なモジュール：「https://code.angularjs.org/1.7.0/angular.min.js」

## 基本的な実装

<!DOCTYPE>

<html lang="ja" ng-app>

<head>

<meta charset="UTF-8">

<title>AngularJSの練習</title>

<script src="https://code.angularjs.org/1.7.0/angular.min.js"></script>

<script src="../js/myscript.js"></script>

</head>

<body>

<h1>AngularJSの練習</h1>

<input type="text" ng-model="name">

<p>こんにちは！{{name}}</p>

</body>

</html>

ポイント！

* + htmlタグに「ng-app」属性を付与する　⇒　AngularJS使用時は必須
  + javascriptソースにangularJSを読込む（上記はオンラインのとき）
  + inputタグに「ng-name」属性を付与　⇒　「{{属性名}}」の形式で参照することができる

## コントローラーの使用

入力値を編集してから出力したい場合に使用する

▽HTMLの実装

<div ng-controller="MainController">

<p>{{users.length}} users.</p>

<ol>

<li ng-repeat="user in users">{{user.name}}</li>

</ol>

</div>

ポイント！

* + コントローラーを使用するタグの親タグに「ng-contorller」属性を付与
  + 属性値はJavaScriptソースと紐づける
  + 「ng-repeat」属性によって繰り返し処理を実装できる
  + 属性値は「仮引数 in 実引数」　※Javaの拡張For文のような書き方

▽JSの実装

angular.module('myapp', [])

.controller('MainController', ['$scope', **function**($scope) {

$scope.users = [

{"name":"ryo", "age": 24},

{"name":"emi", "age": 25}

];

}]);

ポイント！

* + moduleメソッドの第一引数は、htmlタグの「ng-app」属性値と紐づけ
  + controllerメソッドの第一引数は、「ng-contoller」属性値と紐づけ

## フィルター

データを整形するために使用する

▽基本形

{{user.name|uppercase}}

ポイント！

* + 記述は「{{変数 | フィルター}}」

▽フィルター一覧

|  |  |
| --- | --- |
| lowercase | 大文字を小文字に変換 |
| uppercase | 小文字を大文字に変換 |
| json | オブジェクトをJSON形式に変換 |
| orderBy:変数名 | 配列を指定する条件でソート  　※降順の場合は変数名の前に「-」を付与 |
| filter:フィルタ文字列 | 配列を指定する条件でフィルタリング |
| limitTo:件数 | 配列のm～n件目を取得 |
| currency | 通貨型式に変換 |
| date | 日付を整形 |
| number | 数値を文字列として整形 |

▽フィルターの自作

angular.module('myApp' , [])

.**filter**('myFilter', function(){

return function(value){

// filterの処理

return filteredValue;

}

});

ポイント！

* + 「フィルター関数を返す」関数を実装する

▽リアルタイムのフィルタリングの実装

<div ng-controller="MainController">

フィルタ文字列：<input type="text" ng-model="filter.name">

<ol>

<li ng-repeat="user in users|filter:filter|orderBy:'age'">{{user.name}} {{user.age}}</li>

</ol>

</div>

ポイント！

* + 「ng-model=”filter.name”」とすることでnameフィールドのみを対象にフィルタリングが可能
  + 「|」でつなげることで複数のフィルタを付与可能（実装例では、文字列フィルタ⇒ソート）

## ループで使用できる変数

「{{$ループ変数名}}」の形式でループ専用の変数を使用できる

|  |  |
| --- | --- |
| index | 現在の繰り返し回数を取得（0～） |
| first | 最初の要素かどうかの真偽を取得 |
| middle | 真ん中の要素がどうかの真偽を取得 |
| last | 最後の要素かどうかの真偽を取得 |

## イベントリスナーの登録

▽実装例（HTML）

<div ng-controller="MainController">

<table class="table table-bordered table-hover">

<thead class="bg-success"><tr>

<td>name</td>

<td>age</td>

</tr></thead>

<tbody><tr ng-repeat="user in users" ng-controller="userItemCtrl">

<td>{{user.name}}</td>

<td>{{user.age}}</td>

<td><button class="btn btn-primary" **ng-click**="increment()">+1</button></td>

</tr></tbody>

</table>

</div>

ポイント！

* + 「onclick」の代わりに「ng-イベント名」属性を付与する

▽実装例（js）

angular.module('myapp', [])

.controller('MainController', ['$scope', **function**($scope) {

$scope.users = [

{"name":"ryo", "age": 24},

{"name":"emi", "age": 25},

{"name":"ru-", "age": 3}

];

}])

.controller('userItemCtrl', ['$scope', **function**($scope){

$scope.increment = **function**(){

$scope.user.age++;

};

}]);

▽イベント一覧

|  |  |
| --- | --- |
| ng-click | クリック |
| ng-dbclick | ダブルクリック |
| ng-mousedown | マウスボタンが押される |
| ng-mouseup | マウスボタンが離れる |
| ng-mouseenter | 要素の上にマウスポインタが乗る |
| ng-mouseleave | 要素の上からマウスポインタが離れる |
| ng-mousemove | 要素の上でマウスポインタが移動 |
| ng-mouseover | 要素にマウスポインタが重なった |
| ng-mouseout | 要素上に重なっていたマウスポインタが離れる |
| ng-focus | 要素にフォーカス |
| ng-blur | 要素からフォーカスが外れる |
| ng-keydown | キーを押す |
| ng-keypress | キーを押し続ける |
| ng-keyup | キーを離す |
| ng-change | 値を変更 |
| ng-copy | コピー |
| ng-cut | カット |
| ng-paste | ペースト |
| ng-submit | サブミット |

# フォームの基本

## 基本

<form name="sample\_2.form" ng-submit="add()">

<p>Name: <input type="text" name="name" ng-model="user.name"></p>

<p>Age: <input type="text" name="name" ng-model="user.age"></p>

<p><input type="submit" value="add"></p>

</form>

ポイント！

* + ng-model属性を付与
  + 属性値は「オブジェクト名.フィールド名」
  + コントローラーで「$scope.オブジェクト.フィールド」の記述で参照可能

## 様々なフォーム要素の実装

▽実装例

<div ng-controller="mainCtrl">

<form name="myForm" novalidate>

<p>Name:

<input type="text" name="name" ng-model="model.name" required ng-minlength="3" ng-maxlength="8">

<span ng-show="myForm.name.$error.required">Required!</span>

</p>

<p>Email:

<input type="email" name="email" ng-model="model.email">

<span ng-show="myForm.email.$error.email">Not valid email!</span>

</p>

<p>URL:

<input type="url" name="url" ng-model="model.url">

<span ng-show="myForm.url.$error.url">Not valid URL!</span>

</p>

<p>Memo:

<textarea bg-maxlength="200" name="memo" ng-model="model.memo"></textarea>

{{200 - model.memo.length}}

</p>

<p>18+ :

<input type="checkbox" ng-model="model.adult" name="adult" ng-true-value="'adult'" ng-false-value="'child'">

</p>

<p>Phone:

<input type="radio" name="phone" ng-model="model.phone" value="iPhone">iPhone

<input type="radio" name="phone" ng-model="model.phone" value="Android">Android

</p>

<p>Gender:

<select name="gender" ng-model="model.gender" ng-options="gender for gender in ['man', 'woman']" ng-init="model.gender='man'"></select>

</p>

</form>

<pre>{{model|json}}</pre>

</div>

ポイント！

* + バリデートの利用には「ng-バリデート名」の属性を付与する
  + 「フォームname.インプットname.$error.バリデート名」でエラーの有無を確認できる
  + checkbox
    - デフォルトではtrue/false
    - 「ng-true-value/ng-false-value」を付与することで文字列をsubmit可能
    - 記述例：「ng-true-value=”’trueValue’”」
  + select
    - ng-optionsでoptionタグを自動生成
    - 記述例：「ng-options=”field for field in [val1, val2]”」
    - ng-initで初期値を設定
    - 記述例：「ng-init=”model.field=’valel’”」