**Лекція 23. Робота з моделями в Angular**

1. Директива ng-model.
2. Об’єкт.
3. Директива ng-blur.
4. Чекбокси.

## Директива ng-model

Нехай ми маємо звичайний текстовий інпут. Зробимо так, щоб у міру введення даних в інпут ці дані потрапляли в змінну **$scope.message**. Для цього інпуту слід додати атрибут **ng-model**, значенням якого слід вказати 'message':

<input ng-model="message">

Тепер у міру введення даних в інпут миттєво буде змінюватися **$scope.message** і, відповідно, ми можемо вивести **{{message}}** в будь-якому зручному нам місці - ця змінна теж буде змінюватися миттєво (за введенням кожного символу):

<input ng-model="message">

<p>Ваше повідомлення: {{message}}</p>

Давайте спробуємо запустити цей код:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

});

</script>

</head>

<body ng-controller="MyController">

<input ng-model="message">

<p>Ваше повідомлення: {{message}}</p>

</body>

</html>

## Завантаження

У попередньому прикладі **$scope.message** спочатку було порожнім, потім туди потрапили дані з інпуту. Але насправді зв'язок двосторонній - якщо поміняти **$scope.message**, то значення інпуту також зміниться.

Давайте зробимо так, щоб після завантаження сторінки в **$scope.message** вже лежало якесь значення (у нашому випадку 'Привіт!'):

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.message **=** ‘Привіт!’;

});

<input ng-model="message">

<p>Ваше повідомлення: {{message}}</p>

У цьому випадку після заходу на сторінку в інпуті (і в абзаці теж) вже стоятиме значення 'Привіт!', яке ми зможемо редагувати на нове.

Давайте спробуєм:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.message **=** ‘Привіт!’;

});

</script>

</head>

<body ng-controller="MyController">

<input ng-model="message">

<p>Ваше повідомлення: {{message}}</p>

</body>

</html>

## Об'єкт

В атрибуті **ng-model** можна передавати не тільки звичайні змінні (числа та рядки), але й елементи масиву та властивості об'єкта.

Що мається на увазі: якщо у нас, наприклад, даний об'єкт **$scope.user** зі значенням *{name: 'Іван', age: 30}*, то за допомогою інпуту можна змінювати лише ім'я (ключ name) або тільки вік (ключ age). Для цього в **ng-model** слід передати **user.name** для імені та **user.age** для віку.

Давайте зробимо два інпути: один для зміни імені, а другий – для зміни віку:

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.user **=** {name: ‘Іван’, age: 30};

});

<input ng-model="user.name">

<input ng-model="user.age">

<p>Ім’я: {{user.name}}, вік: {{user.age}}</p>

Запустіть наступний код і перегляньте, як це працює:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.user **=** {name: ‘Іван’, age: 30};

});

</script>

</head>

<body ng-controller="MyController">

<input ng-model="user.name">

<input ng-model="user.age">

<p>Ім’я: {{user.name}}, вік: {{user.age}}</p>

</body>

</html>

## Директива ng-blur

У попередніх прикладах у міру введення даних в інпут зміни HTML коді відбувалися миттєво. Однак така поведінка іноді може заважати.

Якщо потрібно, щоб зміни відбувалися після введення даних в інпут - використовуйте атрибут **ng-blur**. У ньому вказується функція (з круглими дужками на кінці, як і в ng-click), яка виконається втратою фокусу інпутом.

Давайте розв'яжемо наступне завдання: дано інпут, у нього вводиться число, за втратою фокусу знайдемо корінь із цього числа і запишемо в абзац нижче.

Для цього в інпут додамо дві директиви: *ng-model="num"* буде записувати дані з інпуту **$scope.num**, а *ng-blur="showSqrt()"* виконає функцію **$scope.showSqrt** після закінчення введення даних:

<input ng-model="num" ng-blur="showSqrt()">

Реалізуємо функцію **$scope.showSqrt** в контролері - витягнемо в ній корінь з **$scope.num** і запишемо його в **$scope.sqrt** :

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

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

$scope.sqrt **=** Math.sqrt($scope.num);

}

});

Тепер у нас доступна змінна **$scope.sqrt** – виведемо її в абзаці під інпутом:

<input ng-model="num" ng-blur="showSqrt()">

<p>Корінь числа: {{sqrt}}</p>

Остаточний код буде виглядати так:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

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

$scope.sqrt **=** Math.sqrt($scope.num);

}

});

</script>

</head>

<body ng-controller="MyController">

<input ng-model="num" ng-blur="showSqrt()">

<p>Корінь числа: {{sqrt}}</p>

</body>

</html>

### Важливе зауваження

Зверніть увагу, що при використанні **ng-blur** атрибут **ng-model** обов'язково вказувати.

## Сумісний ng-repeat для списку та інпут з ng-blur

Нехай у нас дано масив із користувачами. Давайте виведемо вміст цього масиву у вигляді списку <ul>, а потім додамо інпут, за допомогою якого ми будемо додавати нові елементи до цього списку.

Висновок у вигляді списку ви вже вмієте робити, наводжу цей код без коментарів:

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.users **=** ['Микола', 'Василь', 'Петро'];

});

<ul>

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

</ul>

Додамо тепер інпут з **ng-model** і **ng-blur** :

<input ng-model="newUser" ng-blur="addUser()">

<ul>

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

</ul>

Після введення даних в інпут буде викликатися функція **$scope.addUser**. У цій функції буде доступна змінна **$scope.newUser**, в якій зберігається новий введений користувач. Додамо його до масиву **$scope.users** за допомогою JavaScript методу push:

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.users **=** ['Микола', 'Василь', 'Петро'];

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

$scope.users.push($scope.newUser);

}

});

Ось, загалом, і все. Зберемо все разом і запустимо:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.users **=** ['Микола', 'Василь', 'Петро'];

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

$scope.users.push($scope.newUser);

}

});

</script>

</head>

<body ng-controller="MyController">

<input ng-model="newUser" ng-blur="addUser()">

<ul>

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

</ul>

</body>

</html>

Після запуску цього коду ви можете вбити нового користувача в інпут, втратити фокус з інпуту і у вас автоматично з'явиться новий пункт списку (нагадую: ng-repeat працює таким чином, що додавання дублів до списку неможливе).

## Чекбокси

Давайте тепер розберемося з чекбоксами. Їм теж допустимо задавати атрибут **ng-model** - у цьому випадку змінну, зазначену в цьому атрибуті, покладеться **true**, якщо чекбокс відзначений і **false** - якщо не відзначений.

Давайте спробуємо на практиці - поставимо в **ng-model** змінну **checked** і виведемо її в абзац нижче:

<input type="checkbox" ng-model="checked">

<p>Значення чекбокса: {{checked}}</p>

Давайте запустимо:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

});

</script>

</head>

<body ng-controller="MyController">

<input type="checkbox" ng-model="checked">

<p>Значення чекбокса: {{checked}}</p>

</body>

</html>

## Чекбокс та ng-change

Давайте вивчимо наступний атрибут - **ng-change**. Він дозволяє задавати функцію, яка спрацьовуватиме зі зміни інпуту (будь-якого - і текстового, і чекбокса).

Давайте зробимо так, щоб по зміні стану чекбоксу викликалася функція **$scope.checkboxChange** та виводила певне повідомлення:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

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

$scope.message **=** 'чекбокс змінено, стан: '**+**$scope.checked;

}

});

</script>

</head>

<body ng-controller="MyController">

<input type="checkbox" ng-change="checkboxChange()" ng-model="checked">

<p>Сообщение: {{message}}</p>

</body>

</html>

## Лічильник натискань чекбоксів

Реалізуємо лічильник натискань по чекбоксу (розберіть цей код самостійно):

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

var i **=** 0;

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

$scope.counter **=** i;

i**++**;

}

});

</script>

</head>

<body ng-controller="MyController">

<input type="checkbox" ng-change="checkboxChange()" ng-model="checked">

<p>Поточний стан: {{checked}}</p>

<p>Кількість змін чекбокса: {{counter}}</p>

</body>

</html>

## Атрибут ng-checked

Розберемо тепер атрибут **ng-checked**, який дозволяє задавати стан чекбоксу – відзначений чи не відзначений. Цей атрибут пишеться змінна. Якщо вона дорівнює true – чекбокс відзначений, якщо false – не відзначений.

Давайте зробимо змінну **$scope.check**, яка зробить чекбокс з *ng-checked="check"* вже відміченим під час заходу на сторінку:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.check **=** true;

});

</script>

</head>

<body ng-controller="MyController">

<input type="checkbox" ng-checked="check">

</body>

</html>

## Змінюємо стан чекбоксу за подією

Давайте тепер зробимо так, щоб після заходу на сторінку чекбокс було відзначено, а після натискання на кнопку позначка знімалася:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.check **=** true;

$scope.changeCheck **=** function () { $scope.check **=** false;}

});

</script>

</head>

<body ng-controller="MyController">

<button ng-click="changeCheck()">Зробити чекбокс невідмічиним</button>

<input type="checkbox" ng-checked="check">

</body>

</html>

## Зміна стану на протилежний

А тепер зробимо так, щоб після натискання на кнопку чекбокс змінював свій стан на протилежний:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

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

<script>

var myApp **=** angular.module('myApp', []);

myApp.controller('MyController', function MyController($scope) {

$scope.check **=** true;

$scope.changeCheck **=** function () {$scope.check **=** **!**$scope.check;} });

</script>

</head>

<body ng-controller="MyController">

<button ng-click="changeCheck()">Змінити стан чекбокса</button>

<input type="checkbox" ng-checked="check">

</body>

</html>