**Лекція 25. Робота з деякими корисними директивами в Angular**

1. Директива ng-show.
2. Директива ng-hide.
3. Директива ng-class.
4. Фільтри.

## Директива ng-show

Давайте розберемо директиву **ng-show**. Вона використовується для показу або приховування елементів.

У цю директиву слід передавати змінну. Якщо ця змінна має значення **true** – елемент показаний, якщо **false** – прихований.

Ось приклад такого елемента: якщо змінна $scope.check має значення true – елемент буде показаний і навпаки:

<p ng-show="check">Абзац з текстом.</p>

Давайте до цього абзацу прив'яжемо чекбокс: якщо він позначений – абзац буде показаний, а якщо не помічений – прихований:

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

<p ng-show="check">Абзац з текстом.</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 type="checkbox" ng-model="check">

<p ng-show="check">Абзац з текстом.</p>

</body>

</html>

## Директива ng-hide

Аналогічно, тільки навпаки, працює директива ng-hide, тільки якщо для цієї директиви передати true - то елемент сховається, а якщо false - з'явиться.

Тобто при використанні ng-hide наш абзац за замовчуванням буде показано, а при використанні ng-show – прихований.

Приклад:

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

<p ng-hide="check">Абзац з текстом.</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 type="checkbox" ng-model="check">

<p ng-hide="check">Абзац з текстом.</p>

</body>

</html>

## Директива ng-class

Наступна корисна директива, яка нам потрібна, називається **ng-class**. Вона дозволяє використовувати CSS класи до елемента залежно від значень змінних.

Нехай у нас даний клас.colored, який фарбує елемент в зелений колір:

.colored {

color: green;

}

Давайте прив'яжемо цей клас до абзацу за допомогою ng-class. Оскільки переданий true - то абзац буде пофарбовано:

<p ng-class="{colored: true}">Абзац з текстом.</p>

Якщо передати false – то елемент не буде пофарбований:

<p ng-class="{colored: false}">Абзац з текстом.</p>

Давайте прив'яжемо чекбокс – якщо він позначений, то абзац буде зеленим, а якщо не помічений – не буде:

.colored {

color: green;

}

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

<p ng-class="{colored: check}">Абзац з текстом.</p>

Давайте спробуємо запустити цей код:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

<style>

.colored {

color: green;

}

</style>

<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="check">

<p ng-class="{colored: check}">Абзац з текстом.</p>

</body>

</html>

## Багато класів

У ng-class можна прив'язати не один клас, а кілька. Нехай у нас є два класи:.colored та.bordered. Прив'яжемо їх до двох різних чекбоксів:

.colored {

color: green;

}

.bordered {

border: 1px solid green;

}

<input type="checkbox" ng-model="check1">

<input type="checkbox" ng-model="check2">

<p ng-class="{colored: check1, bordered: check2}">Абзац з текстом.</p>

Давайте спробуємо запустити цей код:

<!DOCTYPE html>

<html ng-app="myApp">

<head>

<meta charset="utf-8">

<style>

.colored {

color: green;

}

.bordered {

border: 1px solid green;

}

</style>

<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="check1">

<input type="checkbox" ng-model="check2">

<p ng-class="{colored: check1, bordered: check2}">Абзац з текстом.</p>

</body>

</html>

## Фільтри

Фільтри можуть застосовуватись на виведенні значень змінних таким чином: **{{змінна | фільтр}}**. У цьому випадку фільтр щось зробить зі значенням змінної перед виведенням, наприклад, фільтр uppercase переведе рядок у верхній регістр.

Крім того, фільтри застосовуються в ng-repeat таким чином: **ng-repeat="elem in arr | фільтр"**. У цьому випадку цикл виведе не всі елементи масиву, а елементи відповідно до фільтра.

Отже, давайте подивимося, які корисні фільтри є в Angular.

## Фільтри uppercase та lowercase

Фільтри uppercase та lowercase переводять рядки у верхній та нижній регістр відповідно.

Давайте подивимося на їхню роботу на наступному прикладі. Нехай даний масив з користувачами:

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

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

});

Виведемо їх у вигляді списку <ul> так, щоб імена були у верхньому регістрі:

<ul>

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

</ul>

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) {

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

});

</script>

</head>

<body ng-controller="MyController">

<ul>

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

</ul>

</body>

</html>

Фільтр lowercase працює аналогічно, тільки переводить рядок у нижній регістр.

## Фільтр date

Фільтр **date** дозволяє форматувати дату. Дата має бути у форматі *'рік-місяць-день'* або у форматі *'рік-місяць-день години:хвилини:секунди'* (можна також і [об'єкт Date](http://old.code.mu/javascript/date/Date.html) ).

Застосування: **{{змінна | date: формат}}**. Формат вказується спеціальними командами, між якими може бути будь-які роздільники.

Що за команди: наприклад, день можна задати так – **dd** , місяць так – **mm** , а рік – ось так – **yyyy**.

Приклад: дана дата *'2017-12-31'* , давайте виведемо з неї день. Це робиться так: **{{'2017-12-31' | date: 'dd'}}** - виведе '31'.

Виведемо місяць: **{{'2017-12-31' | date: 'mm'}}** - виведе '12'.

Виведемо рік: **{{'2017-12-31' | date: 'yyyy'}}** - виведе '2017'.

Змінимо формат цієї дати на 'день.місяць.рік': **{{'2017-12-31' | date: 'dd.mm.yyyy'}}** - виведе '31.12.2017'.

## Фільтр orderBy

Фільтр **orderBy** дозволяє сортувати дані при виведенні за допомогою ng-repeat.

Давайте подивимося на його роботу на наступному прикладі. Нехай даний масив з користувачами:

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

$scope.users **=** [

{name: 'Микола', age: 20},

{name: 'Василь', age: 19},

{name: 'Петро', age: 21},

];

});

Виведемо їх у вигляді списку <ul>, відсортувавши за віком:

<ul>

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

</ul>

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) {

$scope.users **=** [

{name: 'Микола', age: 20},

{name: 'Василь', age: 19},

{name: 'Петро', age: 21},

];

});

</script>

</head>

<body ng-controller="MyController">

<ul>

<li ng-repeat="user in users | orderBy: 'age'">

{{user.name}} - {{user.age}}

</li>

</ul>

</body>

</html>

Можна змінити порядок сортування на зворотний, передавши другий параметр так: **orderBy: 'age': true**. Значення true змінює порядок на зворотний, а значення false - сортування за зростанням (за умовчанням).

Давайте зробимо селект, в якому вибиратиметься поле для сортування, і чекбокс, натискання на який змінюватиме порядок сортування:

<!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 **=** [

{name: 'Микола', age: 20},

{name: 'Василь', age: 19},

{name: 'Петро', age: 21},

{name: 'Аня', age: 27},

{name: 'Дмитро', age: 30},

];

});

</script>

</head>

<body ng-controller="MyController">

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

<select ng-model="field">

<option>name</option>

<option>age</option>

</select>

<ul>

<li ng-repeat="user in users | orderBy: field: reverse">

{{user.name}} - {{user.age}}

</li>

</ul>

</body>

</html>

## Фільтр limitTo

Фільтр **limitTo** дозволяє обмежити кількість елементів масиву під час виведення за допомогою ng-repeat. Як користуватися: {{змінна | limitTo: кількість}} або {{змінна | limitTo: кількість: початок}}.

Давайте зробимо два інпути: у першому вводиться кількість користувачів, яке слід показати, а в другому - з якого користувача показувати:

<!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 **=** [

{name: 'Микола', age: 20},

{name: 'Василь', age: 19},

{name: 'Петро', age: 21},

{name: 'Аня', age: 27},

{name: 'Дмитро', age: 30},

];

});

</script>

</head>

<body ng-controller="MyController">

<input ng-model="number">

<input ng-model="start">

<ul>

<li ng-repeat="user in users | orderBy: number: start">

{{user.name}} - {{user.age}}

</li>

</ul>

</body>

</html>

## Фільтр filter

Фільтр **filter** дозволяє реалізувати пошук по масиву.

Давайте зробимо інпут, в який можна вводити літери імені користувача, тим самим обмежуючи список видимих ​​користувачів:

<!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 **=** [

{name: 'Микола', age: 20},

{name: 'Василь', age: 19},

{name: 'Петро', age: 21},

{name: 'Аня', age: 27},

{name: 'Дмитро', age: 30},

];

});

</script>

</head>

<body ng-controller="MyController">

<input ng-model="search">

<ul>

<li ng-repeat="user in users | filter: search">

{{user.name}} - {{user.age}}

</li>

</ul>

</body>

</html>