



ANGULARJS
by Google

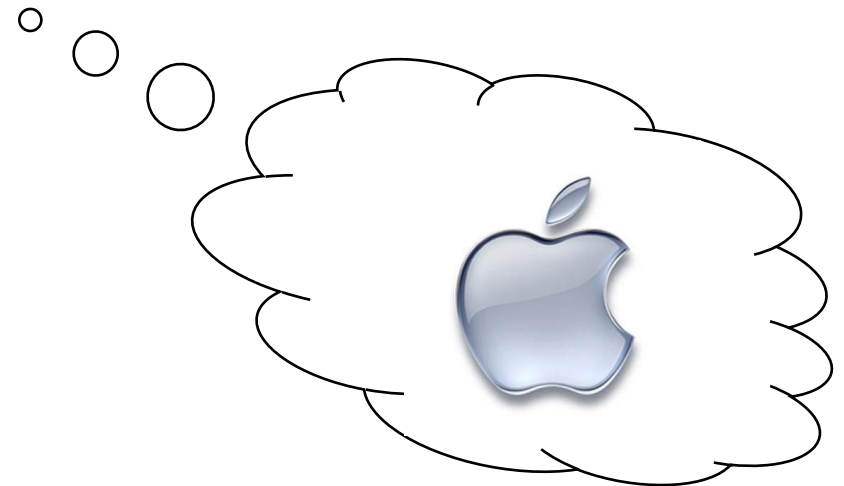
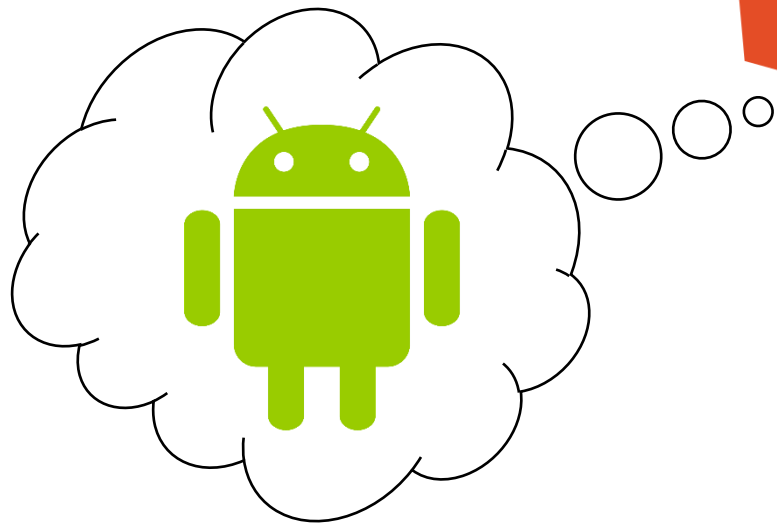
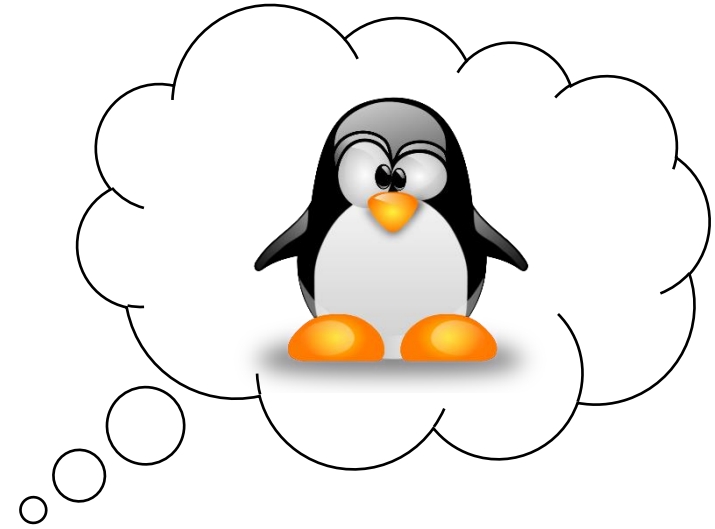
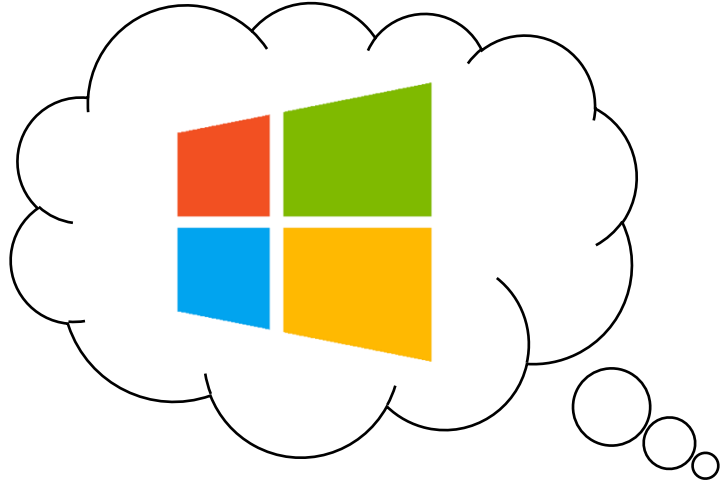


Agenda

1. Dlaczego AngularJS ?
2. Jak to działa ?
3. Główne założenia.
4. Struktura aplikacji.
5. Kilka niezbędnych rad.
6. To naprawdę działa – demo aplikacji.

Dlaczego  AngularJS ?

Dlaczego AngularJS ?



Dlaczego  AngularJS ?

 BACKBONE.JS ?

Dlaczego  AngularJS ?

 BACKBONE.JS ?

 ?

Dlaczego  AngularJS ?

 BACKBONE.JS ?

 ?

 Sencha ?

Dlaczego  AngularJS ?

 BACKBONE.JS ?

 ?

 Sencha ?

 ?

Dlaczego  AngularJS ?

 BACKBONE.JS ?

 ?

 ?

 Sencha ?

 ?

Dlaczego  AngularJS ?

 BACKBONE.JS ?

 ?

 ?



 **Sencha** ?

 ?

Dlaczego AngularJS ?

Zainteresowanie w ujęciu czasowym ?

☐ Porównaj z kategorią ?

☐ Nagłówki wiadomości ?

☐ Prognoza ?

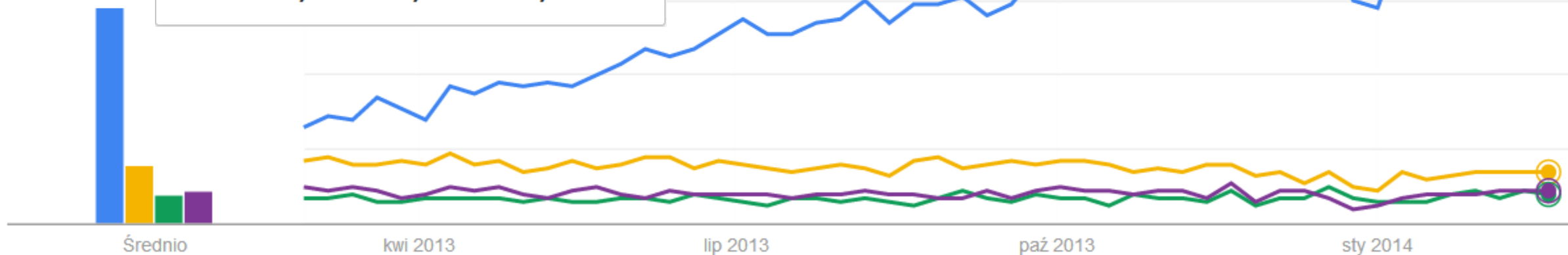
16-22 lut 2014

■ angular.js+angularjs+"angular js": 100

■ backbone.js+backbonejs+"backbone js": 14

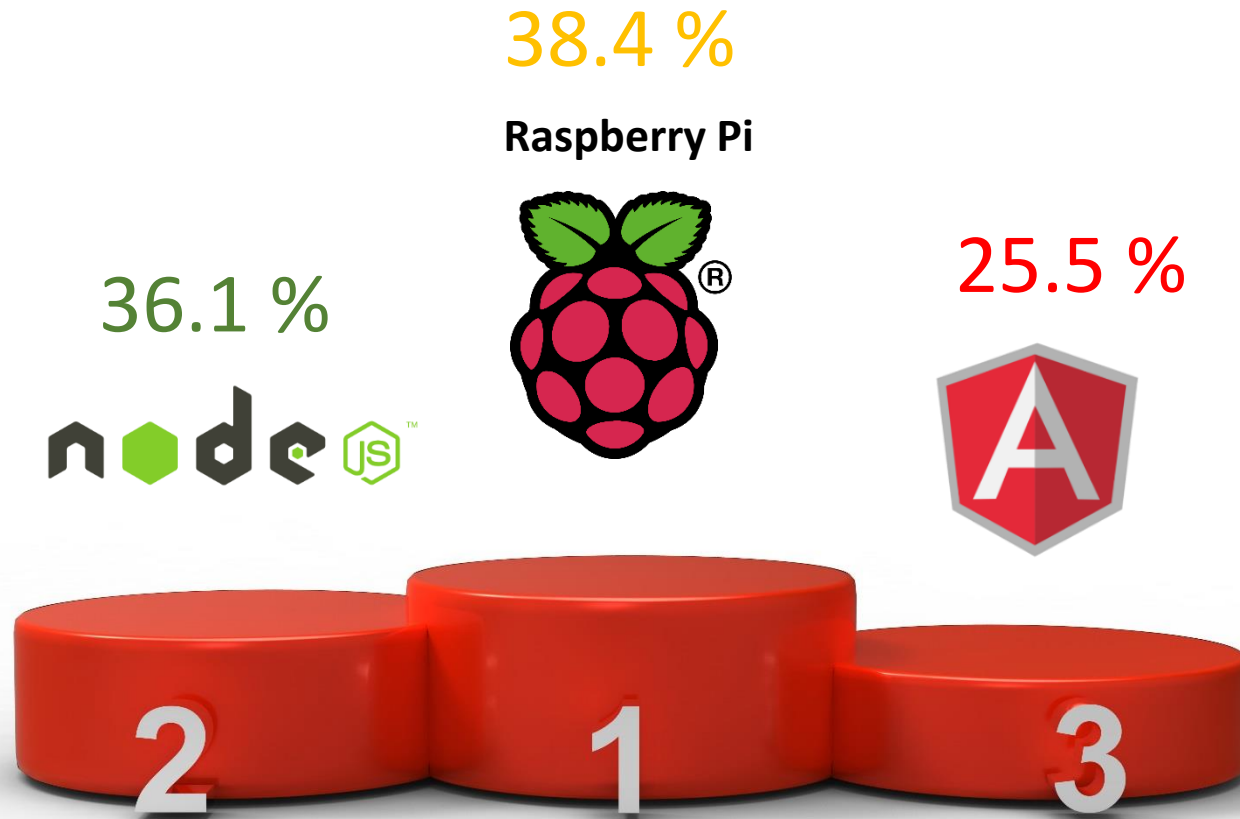
■ ember.js+emberjs+"ember js": 8

■ knockout.js+knockoutjs+"knockout js": 9



Hasło `can.js+canjs+"can js"` nie było wyszukiwane dość często, by pojawić się na wykresie. Spróbuj wybrać dłuższy okres.

Dlaczego AngularJS ?



Dlaczego  AngularJS ?

106 NEAT THINGS

BUILT WITH



Dlaczego  AngularJS ?

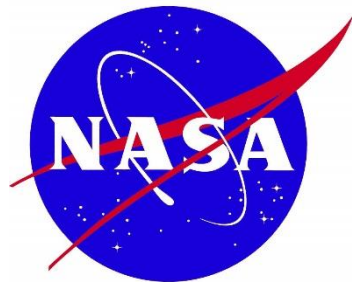
Google™

You Tube

PS3
PlayStation 3

NASCAR®

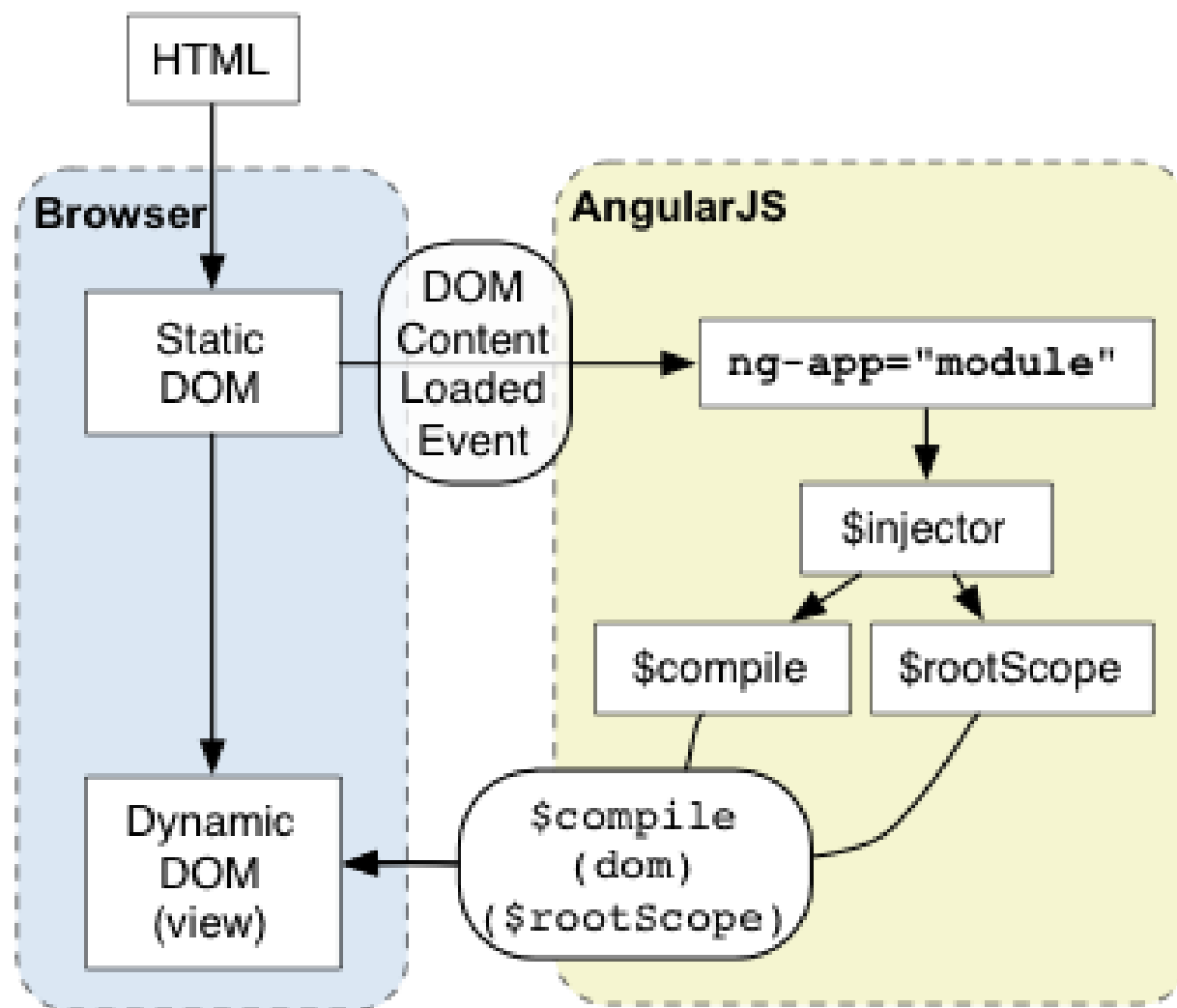
ITAP
TACC



msnbc

Jak to działa  ?

Jak to działa ?



Główne założeni



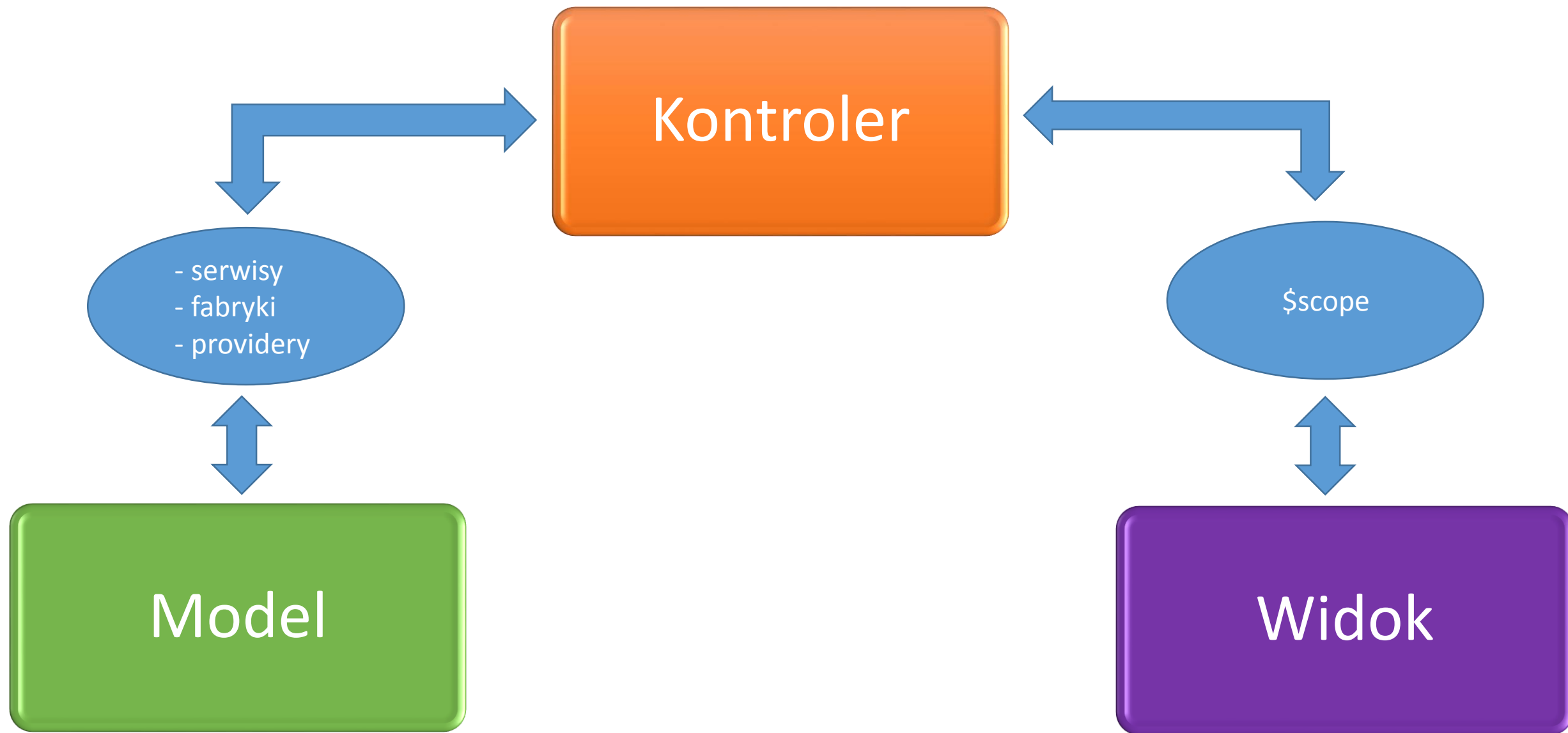


Superheroic JavaScript MVW Framework

wzorzec MVC

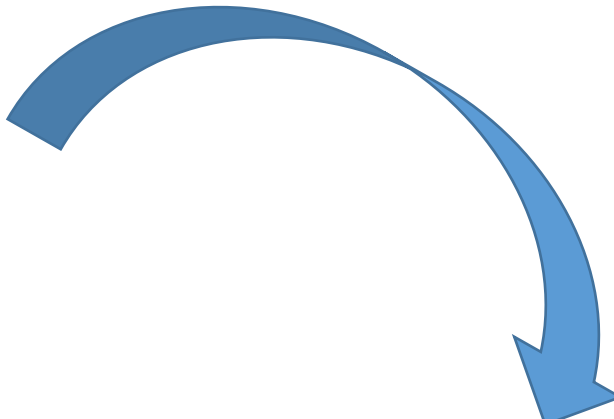


Wzorzec MVC



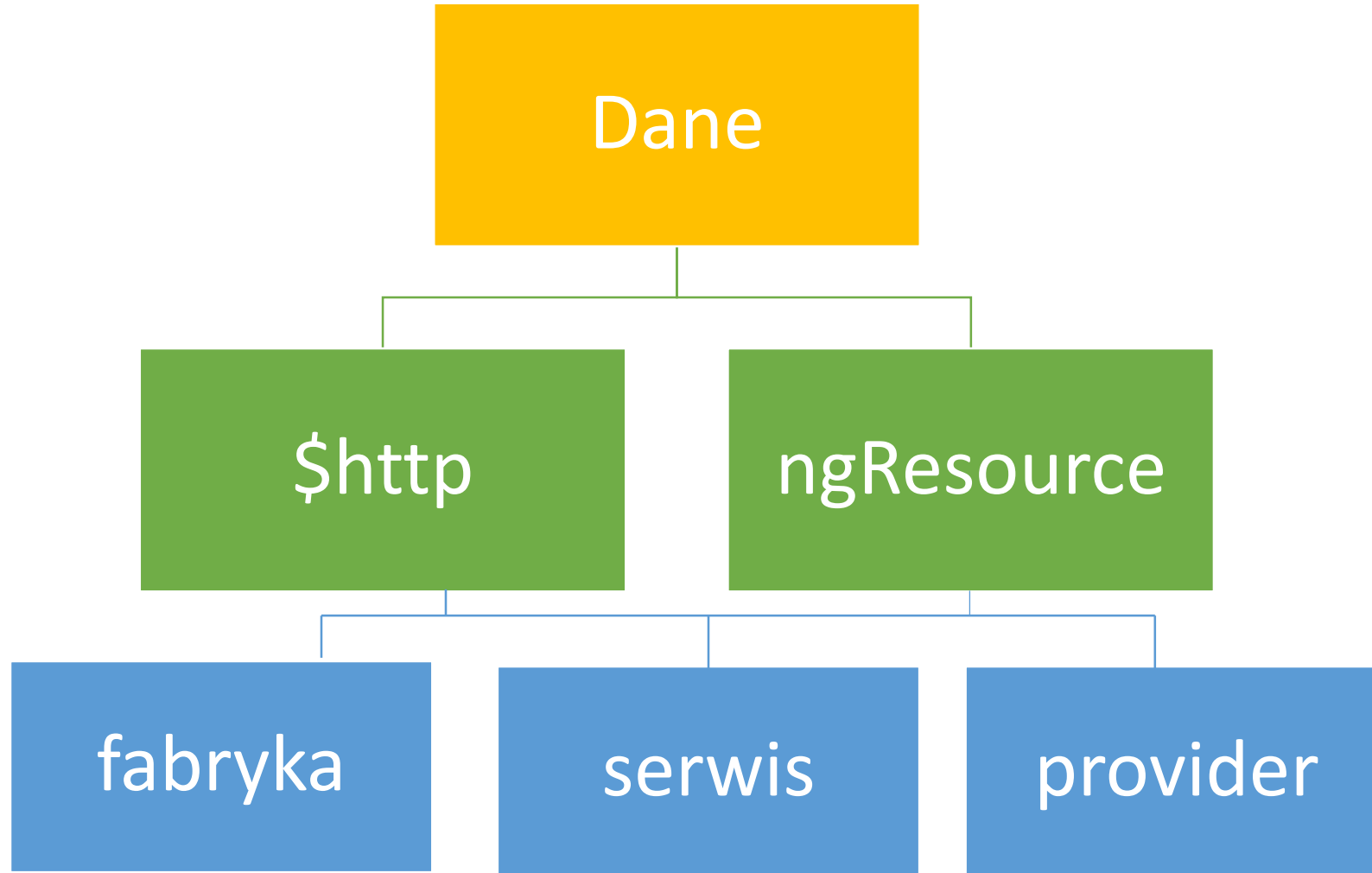
Kontroler – obsługa żądań użytkownik

```
myAppModule.controller('ProductsController', function ($scope, productsFactory) {  
  
    $scope.products = [];  
  
    $scope.updateProduct = function (newPrice) {  
        productsFactory.productManagerToController.get({ productID: 1 }, function (product) {  
            product.price = newPrice;  
            product.save();  
        });  
    };  
  
    $scope.getAllProducts = function () {  
        $scope.products = productsFactory.getProducts();  
    }  
  
    $scope.addProduct = function (newProduct) {  
        productsFactory.productManagerToController.save({}, newProduct);  
    }  
  
    $scope.deleteProduct = function (id) {  
        productsFactory.productManagerToController.remove({ productId: id });  
    };  
});
```



```
<div ng-controller="ProductsController" ng-init="getAllProducts()" class="panel panel-default">  
    <div class="input-group">  
        <input ng-model="SearchPattern" class="" placeholder="Search" type="text" class="form-control">  
    </div>  
    <table>  
        <tr>  
            <th>Name</th>  
            <th>Price</th>  
            <th>Count</th>  
            <th>Actions</th>  
        </tr>  
        <tr ng-repeat="product in products | filter:SearchPattern">  
            <td>{{product.Name}}</td>  
            <td>{{product.Price}}</td>  
            <td>{{product.Count}}</td>  
            <td>  
                <button class="btn btn-default" ng-click="updateProduct(product.Price+10)">Update</button></td>  
                <button class="btn btn-default" ng-click="deleteProduct(product.Id)">Delete</button></td>  
            </td>  
        </tr>  
    </table>  
</div>
```

Modele aplikacji – zarządzanie danymi



Modele aplikacji – zarządzanie danymi

```
$http.get('api/GetProducts', {}).success(
function (data, status, headers, config) {
    // data from server
}).error(
function (data, status, headers, config) {
    // error
});

var dataToSend = {name:'NewProduct1'};

$http.post('api/AddProduct', dataToSend, {}).success(
function (data, status, headers, config) {
    // data from server
}).error(
function (data, status, headers, config) {
    // error
});

myAppModule.factory('productsFactory',['$http',function($http){
    return {
        get: function (productId) {
            return $http.get('api/GetProduct' + '/' + productId);
        }
    };
}]);
```

\$http

```
myAppModule.factory('productsFactory', ['$resource', function ($resource) {
    var factory = {};
    var products = [];

    var productManager = $resource('http://localhost:56537/api/Product/:action/:id',
    { productID: '@productID' }, {
        getAllProducts: { method: 'POST', params: { action: "GetProducts" }, isArray: true }
    });

    factory.productManagerToController = productManager;

    factory.getProducts = function () {
        productManager.getAllProducts({}).$promise.then(function (products) {
            products.forEach(function (product) {
                products.push(product);
            })
        },
        function (response) {
            // error
        });
        return products;
    }
    return factory;
}]);

productsFactory.productManagerToController.get({ productID: 1 }, function (product) {
    product.price = newPrice;
    product.save();
});

$scope.getAllProducts = function () {
    $scope.products = productsFactory.getProducts();
}
```

ngResource

Modele aplikacji – zarządzanie danymi

```
myAppModule.factory('productsFactory', function () {  
  
    var factory = {};  
  
    var products = [];  
  
    factory.getProducts = function () {  
        return products;  
    }  
  
    return factory;  
});
```

fabryka

```
myAppModule.provider('productsProvider', function () {  
  
    this.$get = function () {  
        var products = [];  
        return {  
            getProducts: function () {  
                return products;  
            }  
        }  
    };  
});
```

provider

```
myAppModule.service('productsService', function () {  
  
    this.products = [];  
  
    this.getProducts = function () {  
        return products;  
    };  
});
```

serwis

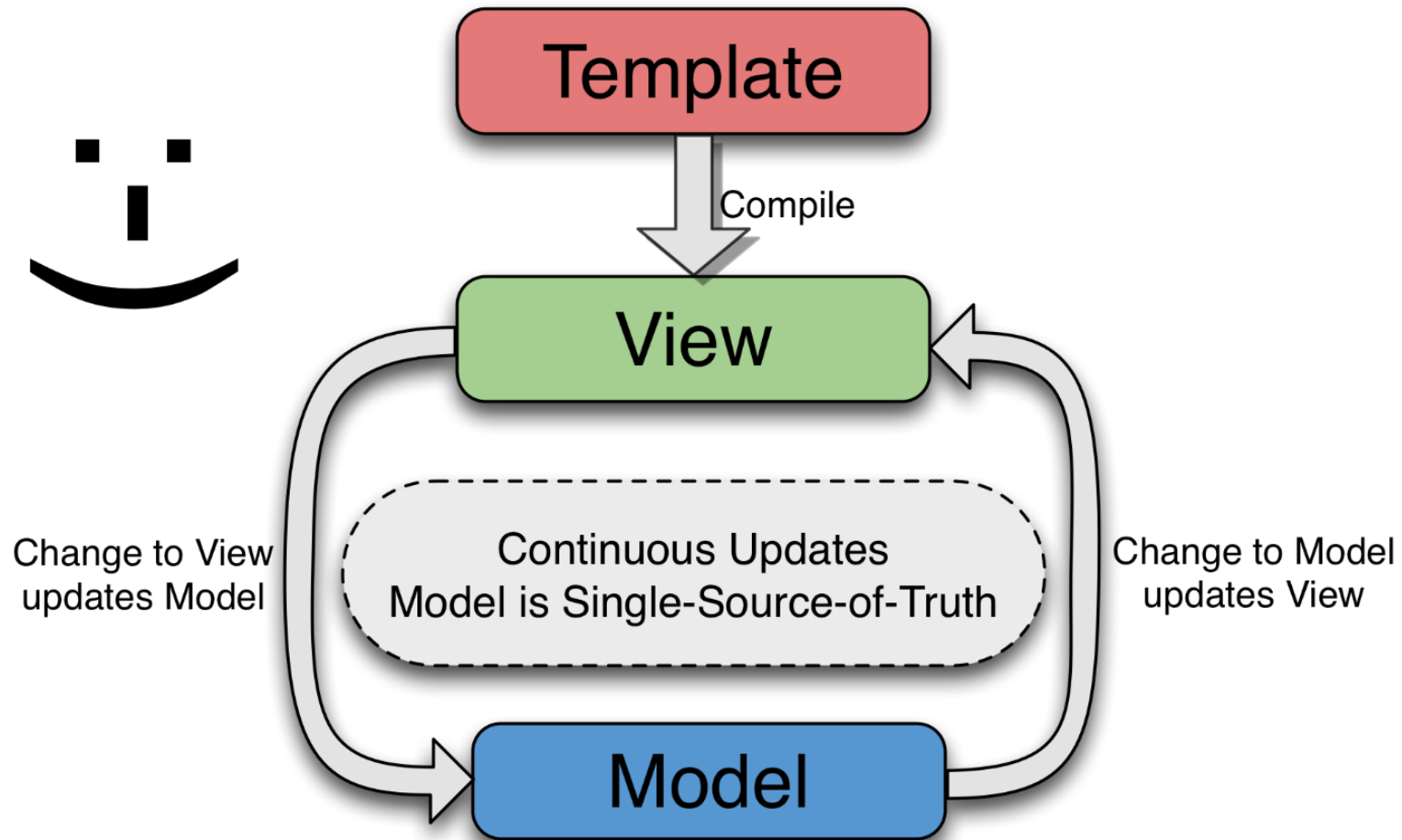


Superheroic JavaScript MVW Framework

wzorzec MVC

two way data binding

Two w y data binding



Two w y data binding

HTML

```
1 <div ng-controller="MyCtrl">
2   <input type="text" ng-model="name"/>
3   <div>{{ hello }}, {{name}}!</div>
4 </div>
```

JavaScript

```
1 var myApp = angular.module('myApp', []);
2
3 function MyCtrl($scope) {
4   $scope.hello = 'Hello';
5   $scope.name = ' ';
6 }
```

Big World

Hello, Big World!



Superheroic JavaScript MVW Framework

wzorzec MVC

two way data binding

rozszerzenie HTML

AngularJS - dyrektywy

zdarzenia

- ngDbclick
- ngClick
- ngChange
- ngChacked
- ngCopy
- ngCut
- ngFocus
- ngKeydown, press, up
- ngMousedown, enter, leave, move, over, up
- ngSelected
- ngBlur
- ngPaste

dane

- ngModel
- ngRepeat
- ngForm
- ngSubmit
- ngList
- ngPluralize

widok

- ngStyle
- ngClass
- ngShow
- ngHide
- ngDisabled
- ngReadonly
- ngView
- ngBindHtml
- ngBindTemplate
- ngCloak
- ngIf
- ngOpen
- ngSwitch

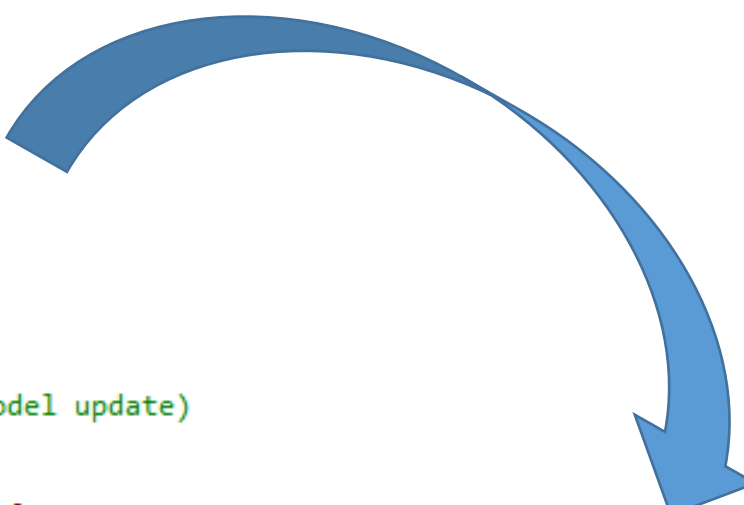
AngularJS - dyrektywy

```
<div ng-controller="ProductsController" ng-init="getAllProducts()" class="panel panel-default">
  <div class="input-group">
    <input ng-model="SearchPattern" class="" placeholder="Search" type="text" class="form-control">
  </div>
  <table>
    <tr>
      <th>Name</th>
      <th>Price</th>
      <th>Count</th>
      <th>Actions</th>
    </tr>
    <tr ng-repeat="product in products | filter:SearchPattern">
      <td>{{product.Name}}</td>
      <td>{{product.Price}}</td>
      <td>{{product.Count}}</td>
      <td>
        <button class="btn btn-default" ng-click="updateProduct(product.Price+10)">Update</button></td>
        <button class="btn btn-default" ng-click="deleteProduct(product.Id)">Delete</button></td>
      </td>
    </tr>
  </table>
</div>
```



AngularJS – tworzenie własnych dyrektyw

```
var INTEGER_REGEXP = /^\-?\d*$/;
myAppModule.directive('integer', function () {
  return {
    require: 'ngModel',
    link: function (scope, elm, attrs, ctrl) {
      ctrl.$parsers.unshift(function (viewValue) {
        if (INTEGER_REGEXP.test(viewValue)) {
          // it is valid
          ctrl.$setValidity('integer', true);
          return viewValue;
        } else {
          // it is invalid, return undefined (no model update)
          ctrl.$setValidity('integer', false);
          return undefined;
        }
      });
    }
  };
});
```



```
<form>
  <div class="newProductForm">
    <div>
      Name
      <input type="text" class="form-control" ng-model="newProduct.Name" />
    </div>
    <div>
      Price
      <input type="text" class="form-control" ng-model="newProduct.Price" />
    </div>
    <div>
      Count
      <input type="text" class="form-control" ng-model="newProduct.Count" integer/>
    </div>
    <div>
      <button class="addButton btn btn-default" ng-click="addProduct(newProduct)">Add product</button>
    </div>
  </div>
</form>
```

AngularJS - filtry

```
<body>
  <div ng-controller="Ctrl">
    Enter number: <input ng-model='val'><br>

    Default formatting: {{val | number}}<br>
    No fractions: {{val | number:0}}<br>
    Negative number: {{-val | number:4}}

  </div>
</body>
```

Enter number:

Default formatting: 1,234.568

No fractions: 1,235

Negative number: -1,234.5679

number

```
<body>
  <span ng-non-bindable>{{1288323623006 | date:'medium'}}</span>:
    {{1288323623006 | date:'medium'}}<br>
  <span ng-non-bindable>{{1288323623006 | date:'yyyy-MM-dd HH:mm:ss Z'}}</span>:
    {{1288323623006 | date:'yyyy-MM-dd HH:mm:ss Z'}}<br>
  <span ng-non-bindable>{{1288323623006 | date:'MM/dd/yyyy @ h:mma'}}</span>:
    {{1288323623006 | date:'MM/dd/yyyy @ h:mma'}}<br>
</body>
```

```
{{1288323623006 | date:'medium'}}: Oct 29, 2010 5:40:23 AM
{{1288323623006 | date:'yyyy-MM-dd HH:mm:ss Z'}}: 2010-10-29 05:40:23 +0200
{{1288323623006 | date:'MM/dd/yyyy @ h:mma'}}: 10/29/2010 @ 5:40AM
```

date

```
<table class="friend">
  <tr>
    <th><a href="" ng-click="predicate = 'age'; reverse=!reverse">Age</a></th>
  </tr>
  <tr ng-repeat="friend in friends | orderBy:predicate:reverse">
    <td>{{friend.name}}</td>
    <td>{{friend.phone}}</td>
    <td>{{friend.age}}</td>
  </tr>
</table>
```

Name (^)	Phone Number	Age
John	555-1212	10
Mary	555-9876	19
Mike	555-4321	21
Julie	555-8765	29
Adam	555-5678	35

orderBy

AngularJS – tworzenie własnych filtrów

```
myAppModule.module('filters', []).  
  filter('truncate', function () {  
    return function (text, length) {  
      if (text.length <= length) {  
        return text;  
      }  
      else {  
        return String(text).substring(0,length);  
      }  
    };  
  });
```



```
<div ng-app="MyModule">  
  <div>  
    <input type="text" ng-model="text" placeholder="" />  
  </div>  
  <div>  
    <p>{{text|truncate:5}}</p>  
  </div>  
</div>
```



weqweweqweqweqweqw

weqwe

AngularJS – zarządzanie widokami częściowymi

```
myAppModule.config(function ($routeProvider) {  
  $routeProvider  
    .when('/List',  
    {  
      controller: 'ProductsController',  
      templateUrl: 'List.html'  
    })  
    .when('/AddProduct',  
    {  
      controller: 'ProductsController',  
      templateUrl: 'AddProduct.html'  
    })  
    .otherwise(  
    {  
      redirectTo: '/'  
    })  
  });  
});
```



```
<form>  
  <div class="newProductForm">  
    <div>  
      Name  
      <input type="text" class="form-control" ng-model="newProduct.Name" />  
    </div>  
    <div>  
      Price  
      <input type="text" class="form-control" ng-model="newProduct.Price" />  
    </div>  
    <div>  
      Count  
      <input type="text" class="form-control" ng-model="newProduct.Count" />  
    </div>  
    <button class="addButton btn btn-default" ng-click="addProduct(newProduct)">Add product</button>  
  </div>  
</form>
```

```
<div ng-controller="ProductsController" ng-init="getAllProducts()" class="panel panel-default">  
  <div class="input-group">  
    <input ng-model="SearchPattern" class="" placeholder="Search" type="text" class="form-control">  
  </div>  
  <table>  
    <tr>  
      <th>Name</th>  
      <th>Price</th>  
      <th>Count</th>  
      <th>Actions</th>  
    </tr>  
    <tr ng-repeat="product in products | filter:SearchPattern">  
      <td>{{product.Name}}</td>  
      <td>{{product.Price}}</td>  
      <td>{{product.Count}}</td>  
      <td>  
        <button class="btn btn-default" ng-click="updateProduct(product.Price+10)">Update</button>  
        <button class="btn btn-default" ng-click="deleteProduct(product.Id)">Delete</button>  
      </td>  
    </tr>  
  </table>  
</div>
```



Superheroic JavaScript MVW Framework

wzorzec MVC

two way data binding

rozszerzenie HTML

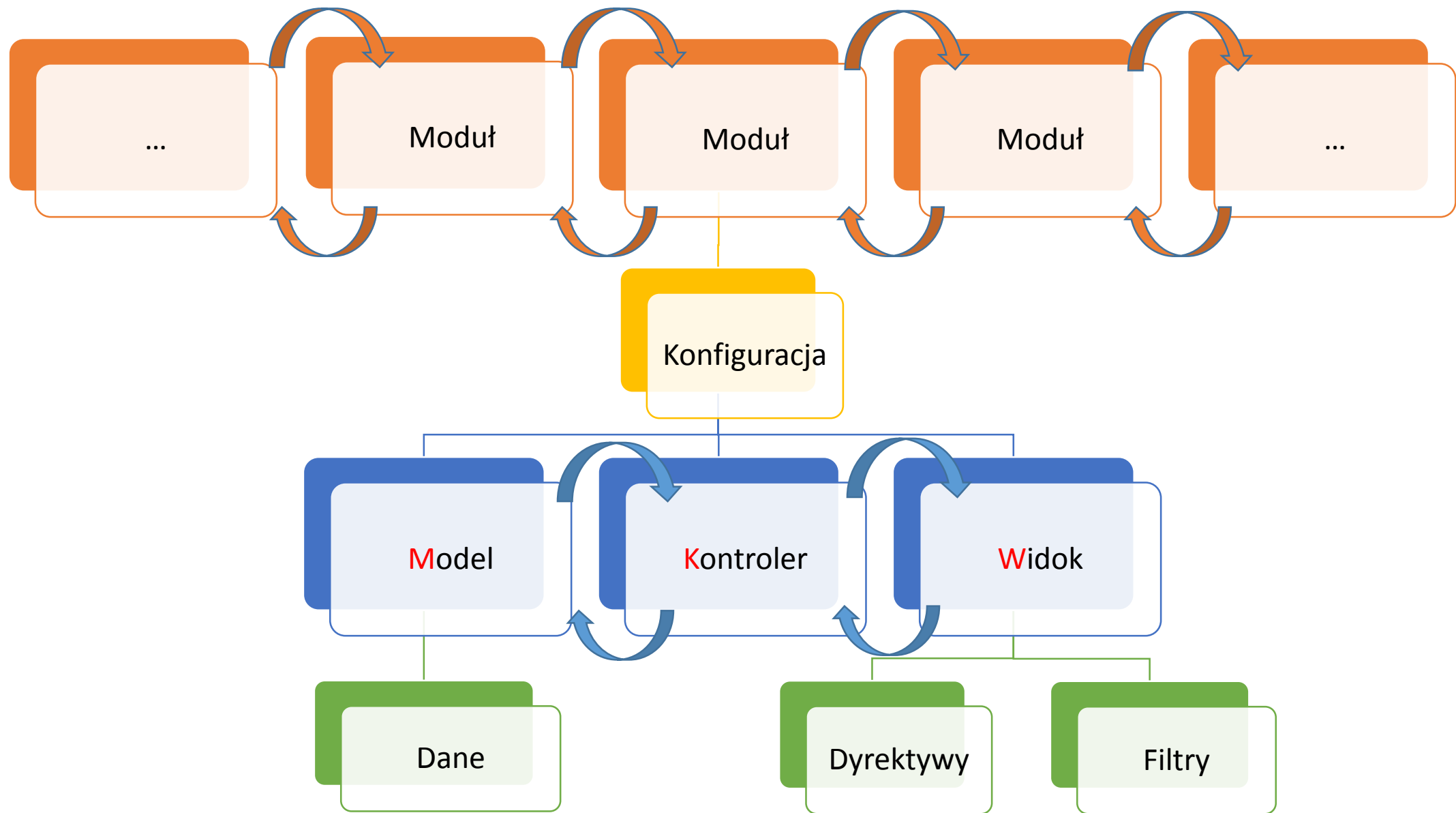
testy jednostkowe

AngularJS – testy jednostkowe

```
1. describe('Buzz Client', function() {
2.   it('should filter results', function() {
3.     input('user').enter('jacksparrow');
4.     element(':button').click();
5.     expect(repeater('ul li').count()).toEqual(10);
6.     input('filterText').enter('Bees');
7.     expect(repeater('ul li').count()).toEqual(1);
8.   });
9. });
```

Struktura Aplikacji

Struktura aplikacji



Zalety  wady ?

Zalety wady ?



Internet Explorer 8



Oficjalna dokumentacja AngularJS

Kilka niezbędnych rd

Kilka niezbędnych r d

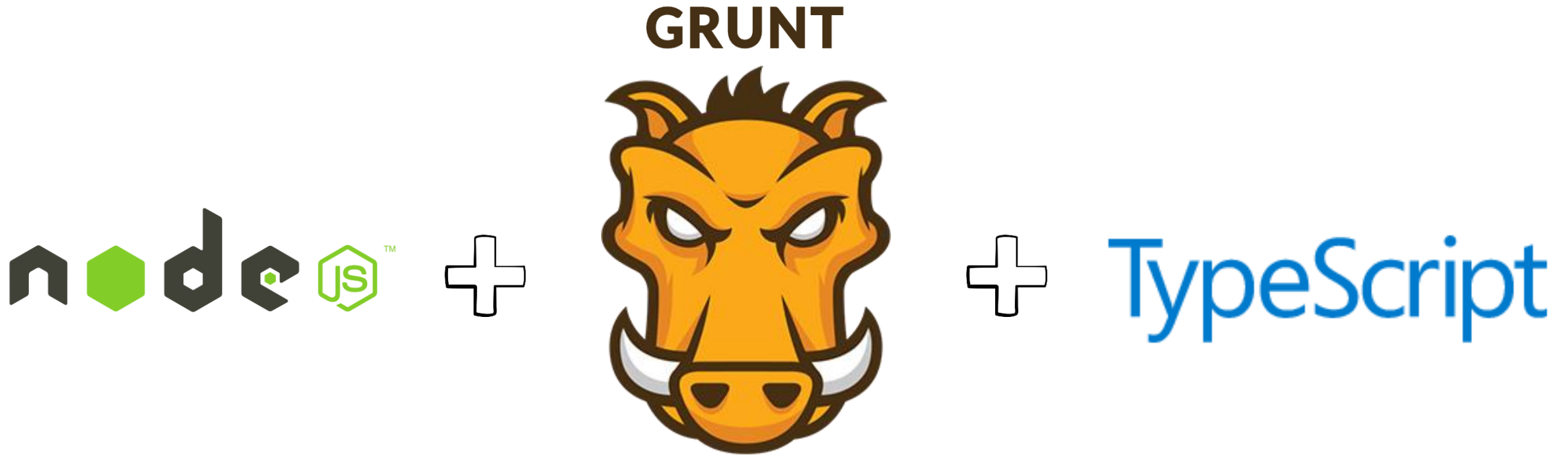
Uważaj na :

- \$scope
- \$rootScope
- Kontrolery

Unikaj :

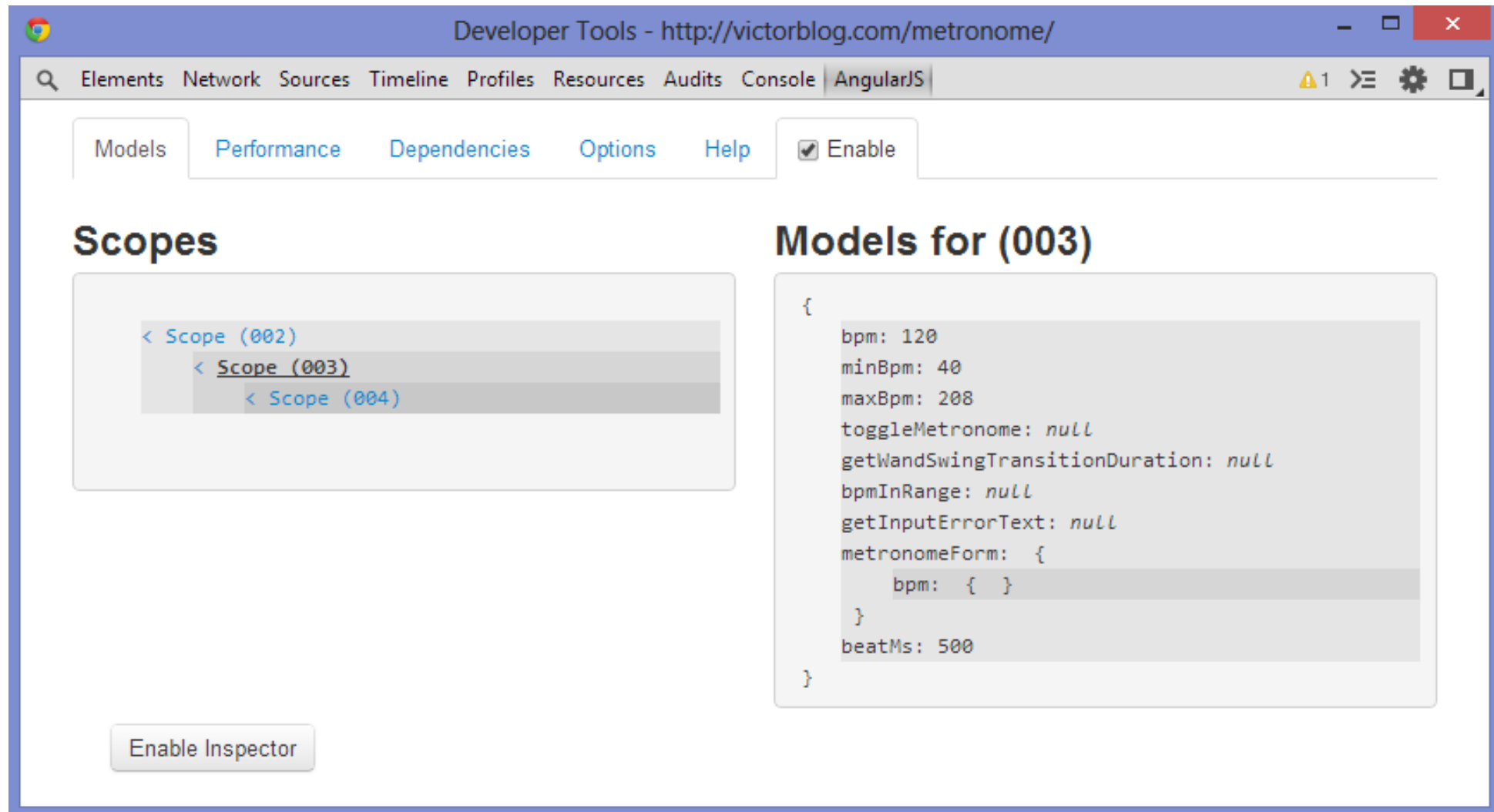
- Logika w widoku
- Odwołania z poziomu kontrolera do obiektów DOM
- Natłoku dyrektyw w jednym elemencie
- Zmienne i metody globalne

Kilka niezbędnych narzędzi – kompilacja i utrzymanie



The JavaScript Task Runner

Kilka niezbędnych rd – debugowanie



The screenshot displays the AngularJS Developer Tools interface within a web browser window. The browser's address bar shows the URL `http://victorblog.com/metronome/`. The Developer Tools toolbar includes tabs for Elements, Network, Sources, Timeline, Profiles, Resources, Audits, Console, and AngularJS. The AngularJS tab is active, showing a sub-menu with Models, Performance, Dependencies, Options, and Help. An 'Enable' checkbox is checked.

The main content area is divided into two panels:

- Scopes:** This panel shows a hierarchical tree of scopes. The current selection is `< Scope (003)`, which is a child of `< Scope (002)`. Below it, `< Scope (004)` is also visible.
- Models for (003):** This panel displays the model data for the selected scope. The data is as follows:

```
{
  bpm: 120
  minBpm: 40
  maxBpm: 208
  toggleMetronome: null
  getWandSwingTransitionDuration: null
  bpmInRange: null
  getInputElementText: null
  metronomeForm: {
    bpm: { }
  }
  beatMs: 500
}
```

At the bottom left of the interface, there is a button labeled 'Enable Inspector'.

Kilka niezbędnych r d – debugowanie

Performance

☐ Log to console

Watch Tree

Scope (002) | toggle

- `function () {var a=d.url(),b=f.$$replace;if(!o||a!=f.absUrl())o++,c.$evalAsync(function(){c.$broadcast("$locationChangeStart",f.absUrl(),a).defaultPrevented};f.$$parse(a):(d.url(f.absUrl(),b),i(a)));f.$$replace=!1;return o}`
- `function () {return c.hash()}`

Scope (003) | toggle

- `bpm`
- `bpm`
- `toggle` `transition-duration: {{getWandSwingTransitionDuration()}};`
- `{swingLeft: swingLeft, swingRight: swingRight,`

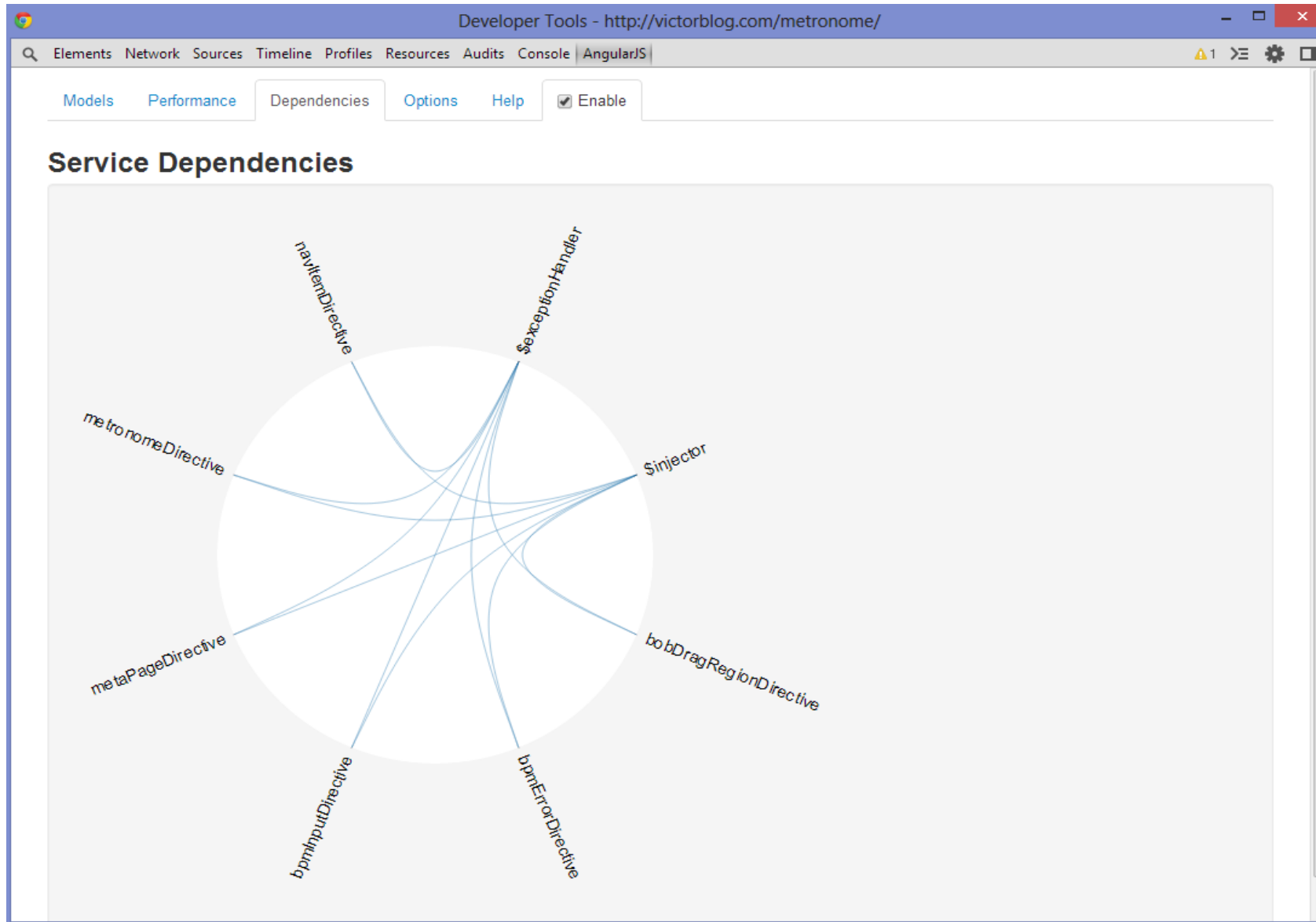
Watch Expressions

`metronomeForm.bpm.$invalid` | 16.7% | 1.000ms

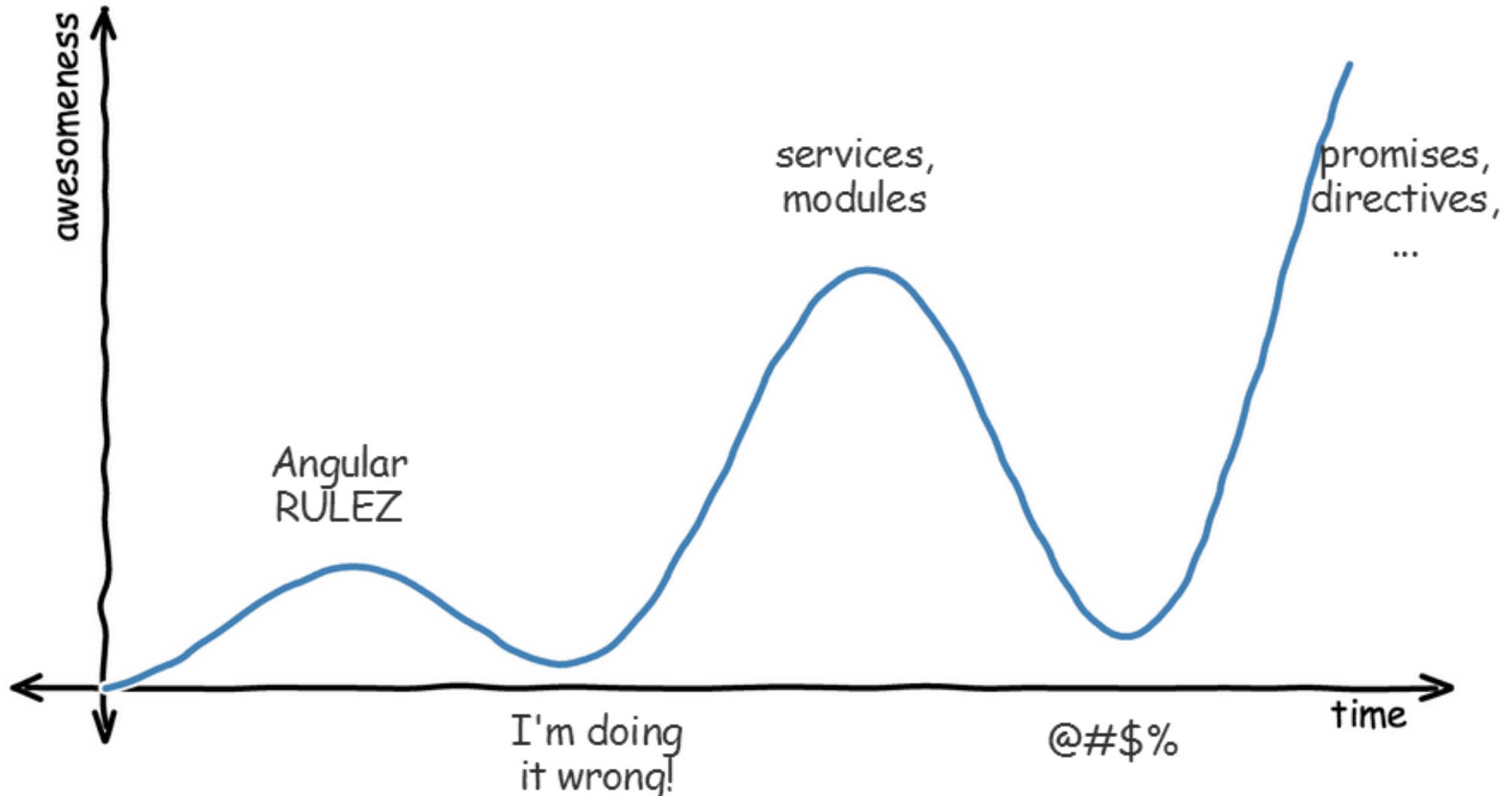
`started` | 16.7% | 1.000ms

`function () {var a=d.url(),b=f.$$replace;if(!o||a!=f.absUrl())o++,c.$evalAsync(function(){c.$broadcast("$locationChangeStart",f.absUrl(),a).defaultPrevented};f.$$parse(a):(d.url(f.absUrl(),b),i(a)));f.$$replace=!1;return o}` | 16.7% | 1.000ms

Kilka niezbędnych r d – debugowanie



ENJOYMENT OF ANGULAR JS



Demo

Dziękuję za uwagę !