adaptive-speech v0.3.0 

This module allows you to control web app using voice commands. It's based on Chrome's speech recognition API.

Demo

Check out http://angular-adaptive.github.io/adaptive-speech/demo/

References

We recomend you to read: - A More Awesome Web presentation from Google IO 2013 by Eric Bidelman - Voice Driven Web Apps article from HTML5 ROCKS by Glen Shires

Requirements

- AngularJS v ~1.2.x

Usage

We use bower for dependency management. Add

```
dependencies: {
    "angular-adaptive-speech": "latest"
}
```

To your `bower.json` file. Then run

```
bower install
```

This will copy the speech recognition files into your `bower_components` folder, along with its dependencies. Load the script files in your application:

```
<script type="text/javascript" src="bower_components/angular/angular.js"></script>
<script type="text/javascript"
    src="bower_components/angular-adaptive-speech/angular-adaptive-speech.min.js">
</script>
```

Add the adaptive.speech module as a dependency to your application module:

```
var myAppModule = angular.module('MyApp', ['adaptive.speech']);
```

and include $speechRecognition, $speechSynthetis, $speechCorrection service as a dependency to your controller:

```
angular.module('MyApp').controller('MainCtrl',
    function ['$scope', '$speechRecognition, $speechSynthetis',
    ($scope, $speechRecognition, $speechSynthetis) {

}]);
```

To start speech recognition run from controller:

```
$speechRecognition.onstart(function(){
  $speechSynthetis.speak('Yes? How can I help you?', 'en-UK');
});
$speechRecognition.setLang('en-UK'); // Default value is en-US
$speechRecognition.listen();
```

Apply the directive to your elements where *reference* is keyword reference:

```
<ul>
    <li
      ng-repeat="todo in todos | filter:statusFilter track by $index"
      speechrecognition="{'tasks': recognition['en-US']['listTasks'], 'reference': todo}"
        {{todo}}
    </li>
</ul>
```

Or run recognition directly from controller:

```
$speechRecognition.listenUtterance($scope.recognition['en-US']['addToList']);
```

Options

All the speechRecognition options can be set up in your controller.

```
myAppModule.controller('MyController', function($scope) {
    $scope.recognition = {};
    $scope.recognition['en-US'] = {
        'addToList': {
            'regex': /^to do .+/gi,
            'lang': 'en-US',
            'call': function(e){
                $scope.addToList(e);
            }
        },
        'listTasks': [{
            'regex': /^complete .+/gi,
            'lang': 'en-US',
```

```
            'call': function(e){
                $scope.completeTask(e);
            }
        },{
            'regex': /^remove .+/gi,
            'lang': 'en-US',
            'call': function(e){
                $scope.removeTask(e);
            }
        }]
    };
});
```

API

$speechRecognition

onstart(fn)

On start event.

```
$speechRecognition.onstart(function(e){
    // onstart
});
```

onerror(fn)

On error event.

```
$speechRecognition.error(function(e){
    // onerror
});
```

onUtterance(cb)

On recognised utterance callback.

```
$speechRecognition.onUtterance(function(utterance){
    console.log(utterance); // buy a milk
});
```

setLang(lang)

Set recognition language.

```
$speechRecognition.setLang('en-US');
```

To change language when recognition is already running you need to also restart recognizer:

```
$speechRecognition.stopListening();
$speechRecognition.listen();
```

getLang()

Get recognition language.

```
$speechRecognition.getLang(); // 'en-US'
```

payAttention()

Continue speech recognition after pause caused by `ignore()`. You don't need user permision again.

```
$speechRecognition.payAttention();
```

ignore()

Pause speech recognition.

```
$speechRecognition.ignore();
```

listen()

Start speech recognition. User permission is required.

```
$speechRecognition.listen();
```

stopListening()

Stop speech recognition.

```
$speechRecognition.stopListening();
```

command(utterance)

Call uterance manually.

```
$speechRecognition.command('do something');
```

listenUtterance(tasks)

Add listener for task

```
var task = {
    'regex': /^do .+/gi,
    'lang': 'en-US',
    'call': function(utterance){
      // do something with utterance 'do something'
    }
};
$speechRecognition.listenUtterance(task);
```

$speechSynthetis

speak(text, lang)

Speak utterance.

```
$speechSynthetis.speak('Hello there!', 'en-US');
```

justSpoke()

Return true after `speak()` has been called.

```
$speechSynthetis.justSpoke(); // true or false
```

recognised()

Manualy mark speechSynthetis voice as recognised. justSpoke will be `true`.

```
$speechSynthetis.recognised();
```

$speechCorrection

Correct speech recognition. After incorrect recognition utterance will be corrected.

addUtterance(utterance, correction, lang)

Create a key - value pair with incorret recognition, correction and language.

```
$speechCorrection.addUtterance('to something', 'do something', 'en-US');
```

removeUtterance(utterance, lang)

Remove utterance correction.

```
$speechCorrection.removeUtterance('to something', 'en-US');
```

addLangMap(lang, map)

Add complete correction map for a language.

```
var map = {
    'to something': 'do something',
    'pseudo make me a sandwich': 'sudo make me a sandwich'
};
$speechCorrection.addUtterance('en-US', map);
```

clearLangMap(lang)

Remove language map.

```
$speechCorrection.clearLangMap('en-US');
```

getCorrectionMap()

Get correction map for all languages.

```
$speechCorrection.getCorrectionMap();
//  {
//      'en-US: {
//          'to something': 'do something',
//          'pseudo make me a sandwich': 'sudo make me a sandwich'
//      }
//  }
```

getLangMap(lang)

Get correction map for a language.

```
$speechCorrection.getCorrectionMap('en-US');
//  {
//      'to something': 'do something',
//      'pseudo make me a sandwich': 'sudo make me a sandwich'
//  }
```

getCorrection(utterance, lang)

Get a single utterance correction.

```
$speechCorrection.getCorrection('pseudo make me a sandwich', 'en-US');
// 'sudo make me a sandwich'
```

speechrecognition directive

Add listener to html element. - tasks: configuration object (*remove something*) - reference: element reference name (*something*)

```
<li
  ng-repeat="todo in todos | filter:statusFilter track by $index"
  speechrecognition="{'tasks': recognition['en-US']['listTasks'], 'reference': todo}">
    {{todo}}
</li>
```

Testing

We use karma and jshint to ensure the quality of the code. The easiest way to run these checks is to use grunt:

```
npm install -g grunt-cli
npm install
bower install
grunt
```

The karma task will try to open Chrome as a browser in which to run the tests. Make sure this is available or change the configuration in `test/test.config.js`

Contributing

Pull requests are welcome.

Make a PR against canary branch and don't bump any versions.

Please respect the code style in place.