عنوان: **قابلیت چند زبانه و Localization در AngularJs- بخش سوم: Best Practiceهای angular-translate**
نویسنده: مهرداد کاهه
تاریخ: ۱۳۹۴/۰۶/۳۱ ۱۶:۲۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, Localization, AngularJS

در این بخش قصد دارم تا در قالب یک پروژه، تمامی قابلیت‌هایی را که در angular-translate و ماژول‌های مرتبط با آن وجود دارند، به شما معرفی کنم. پروژه‌ی نمونه را از لینک زیر دریافت نمایید: AngularJs-Translate-BestPractices.zip

این پروژه در 12 بخش گوناگون تقسیم بندی شده‌است که هر کدام در قالب یک فایل HTML می‌باشد و تمامی اسکریپت‌های مورد نیاز به آن افزوده شده‌است. هر بخش به صورت مجزا به شرح یک ویژگی کاربردی در angular-translate می‌پردازد.

ex1_basic_usage

روند کار مثال اول خیلی ساده است. در ابتدا اسکریپت‌های زیر به صفحه اضافه شده‌اند:

```
<script src="Scripts/angular.js"></script>
    <script src="Scripts/angular-translate.js"></script>
```

در ابتدا وابستگی pascalprecht.translate به ماژول اضافه شده‌است و پس از آن زبان‌های مختلف به صورت JSON در بخش اسکریپت وارد شده‌اند.

```
angular.module('app', ['pascalprecht.translate'])
        .config([
        '$translateProvider', function ($translateProvider) {

            // Adding a translation table for the English language
            $translateProvider.translations('en_US', {
                "TITLE": "How to use",
                "HEADER": "You can translate texts by using a filter.",
                "SUBHEADER": "And if you don't like filters, you can use a directive.",
                "HTML_KEYS": "If you don't like an empty elements, you can write a key for the
translation as an inner HTML of the directive.",
                "DATA_TO_FILTER": "Your translations might also contain any static ({{staticValue}}) or
random ({{randomValue}}) values, which are taken directly from the model.",
                "DATA_TO_DIRECTIVE": "And it's no matter if you use filter or directive: static is
still {{staticValue}} and random is still {{randomValue}}.",
                "RAW_TO_FILTER": "In case you want to pass a {{type}} data to the filter, you have only
to pass it as a filter parameter.",
                "RAW_TO_DIRECTIVE": "This trick also works for {{type}} with a small mods.",
                "SERVICE": "Of course, you can translate your strings directly in the js code by using
a $translate service.",
                "SERVICE_PARAMS": "And you are still able to pass params to the texts. Static =
{{staticValue}}, random = {{randomValue}}."
            });

            // Adding a translation table for the Russian language
            $translateProvider.translations('ru_RU', {
                "TITLE": "Как пользоваться",
                "HEADER": "Вы можете переводить тексты при помощи фильтра.",
                "SUBHEADER": "А если Вам не нравятся фильтры, Вы можете воспользоваться директивой.",
                "HTML_KEYS": "Если вам не нравятся пустые элементы, Вы можете записать ключ для
перевода в как внутренний HTML директивы.",
                "DATA_TO_FILTER": "Ваши переводы также могут содержать любые статичные
({{staticValue}}) или случайные ({{randomValue}}) значения, которые берутся прямо из модели.",
                "DATA_TO_DIRECTIVE": "И совершенно не важно используете ли Вы фильтр или директиву:
статическое значение по прежнему {{staticValue}} и случайное - {{randomValue}}.",
                "RAW_TO_FILTER": "Если вы хотите передать \"сырые\" ({{type}}) данные фильтру, Вам
всего лишь нужно передать их фильтру в качестве параметров.",
                "RAW_TO_DIRECTIVE": "Это также работает и для директив ({{type}}) с небольшими
модификациями.",
                "SERVICE": "Конечно, Вы можете переводить ваши строки прямо в js коде при помощи
сервиса $translate.",
                "SERVICE_PARAMS": "И вы все еще можете передавать параметры в тексты. Статическое
значение = {{staticValue}}, случайное = {{randomValue}}."
            });

            // Tell the module what language to use by default
            $translateProvider.preferredLanguage('en_US');
```

```
        }])
```

در تکه کد فوق مشاهده می‌کنید که دو translate table زبان انگلیسی و روسی به صورت JSON وارد شده‌اند. شما قادرید تا چندین زبان را به همین صورت وارد نمایید. در خط آخر نیز زبان پیش فرض سیستم تعریف شده است.
حال به بررسی کدهای درون کنترلر می‌پردازیم:

```
.controller('ctrl', ['$scope', '$translate', function ($scope, $translate) {

        $scope.tlData = {
            staticValue: 42,
            randomValue: Math.floor(Math.random() * 1000)
        };

        $scope.jsTrSimple = $translate.instant('SERVICE');
        $scope.jsTrParams = $translate.instant('SERVICE_PARAMS', $scope.tlData);

        $scope.setLang = function (langKey) {
            // You can change the language during runtime
            $translate.use(langKey);

            // A data generated by the script have to be regenerated
            $scope.jsTrSimple = $translate.instant('SERVICE');
            $scope.jsTrParams = $translate.instant('SERVICE_PARAMS', $scope.tlData);
        };

    }]);
```

می‌بینیم که وابستگی $translate تزریق شده است. پس از آن دو عدد رندم به پارامترهای تعریف شده در translate table ارسال می‌گردد. تغییر زبان نیز توسط متد setLang صورت می‌پذیرد.
در بخش نهایی می‌خواهیم روش‌های گوناگون استفاده از translate tables را درون HTML نمایش دهیم. به بخش HTML همین مثال توجه کنید:

```
<p>
        <a href="#" ng-click="setLang('en_US')">English</a>
        |
        <a href="#" ng-click="setLang('ru_RU')">Русский</a>
    </p>
    <!-- Translation by a filter -->
    <h1>{{'HEADER' | translate}}</h1>
    <!-- Translation by a directive -->
    <h2 translate="SUBHEADER">Subheader</h2>
    <!-- Using inner HTML as a key for translation -->
    <p translate>HTML_KEYS</p>
    <hr>
    <!-- Passing a data object to the translation by the filter -->
    <p>{{'DATA_TO_FILTER' | translate: tlData}}</p>
    <!-- Passing a data object to the translation by the directive -->
    <p translate="DATA_TO_DIRECTIVE" translate-values="{{tlData}}"></p>
    <hr>
    <!-- Passing a raw data to the filter -->
    <p>{{'RAW_TO_FILTER' | translate:'{ type: "raw" }' }}</p>
    <!-- Passing a raw data to the filter -->
    <p translate="RAW_TO_DIRECTIVE" translate-values="{ type: 'directives' }"></p>
    <hr>
    <!-- Using a $translate service -->
    <p>{{jsTrSimple}}</p>
    <!-- Passing a data to the $translate service -->
    <p>{{jsTrParams}}</p>
```

نحوه تعریف هر روش به صورت کامنت پیش از هر تگ نوشته شده است. شما به روش‌های مختلف و بر حسب استانداردهایی که خود از آن پیروی می‌کنید می‌توانید از یکی از روش‌های فوق استفاده نمایید. اما رایج‌ترین روش، دو روش اول یعنی استفاده از دایرکتیو و یا فیلتر است و دلیل آن هم سادگی و خوانا بودن این دو روش می‌باشد.

ex2_remember_language_cookies

در این مثال همانگونه که از اسم آن پیداست قصد داریم تا زبان مورد نظری را که کاربر در سیستم انتخاب نموده است، ذخیره

کنیم تا پس از بستن و بازکردن مجدد وب سایت با همان زبان پیشین به کاربر نمایش داده شود. این کار بسیار ساده است. کافیست که در ابتدا علاوه بر اسکریپت‌های مثال قبل، اسکریپت‌های زیر را نیز به صفحه اضافه کنید:

```
<script src="Scripts/angular-cookies.js"></script>
    <script src="Scripts/angular-translate-storage-cookie.js"></script>
```

با اضافه کردن خط زیر درون بدنه config، یک کوکی جدید برای شما ساخته می‌شود. این کوکی NG_TRANSLATE_LANG_KEY نام دارد که هر بار با id زبان کنونی که در translate table وارد نموده‌اید آپدیت می‌شود.

```
// Tell the module to store the language in the cookie
$translateProvider.useCookieStorage();
```

حال اگر صفحه را refresh کنید می‌بینید که زبان پیشینی که انتخاب نموده‌اید، مجددا بارگذاری می‌گردد.

ex3_remember_language_local_storage

این مثال همانند مثال قبل رفتار می‌کند، با این تفاوت که به جای اینکه کلید زبان کنونی را درون کوکی ذخیره کند، آن را درون Local Storage با نام NG_TRANSLATE_LANG_KEY قرار می‌دهد. برای اجرا کافیست اسکریپت‌ها و تکه کد زیر را با موارد مثال قبل جایگزین کنید.

```
<script src="Scripts/angular-translate-storage-local.js"></script>


// Tell the module to store the language in the local storage
$translateProvider.useLocalStorage();
```

مثال های ex4_set_a_storage_key و ex5_set_a_storage_prefix نام کلیدی که برای ذخیره سازی زبان کنونی در کوکی یا Local Storage قرار می‌گیرد را تغییر می‌دهد که به دلیل سادگی از شرح آن می‌گذریم.

ex6_namespace_support

translate table در angular-translate قابلیت مفید namespacing را نیز داراست. این قابلیت به ما کمک می‌کند که جهت کپسوله کردن بخش‌های مختلف، ترجمه آنها را با namespaceهای خاص خود نمایش دهیم. به مثال زیر توجه کنید:

```
$translateProvider.translations('en_US', {
                "TITLE": "How to use namespaces",
                "ns1": {
                    "HEADER": "A translations table supports namespaces.",
                    "SUBHEADER": "So you can to structurize your translation table well."
                },
                "ns2": {
                    "HEADER": "Do you want to have a structured translations table?",
                    "SUBHEADER": "You can to use namespaces now."
                }
            });
```

همانطور که توجه می‌کنید بخش ns1 خود شامل زیر مجموعه‌هایی است و ns2 نیز به همین صورت. هر کدام دارای کلید HEADER و SUBHEADER می‌باشند. فرض کنید هر کدام از این بخش‌ها می‌خواهند اطلاعات درون یک section را نمایش دهند. حال به نحوه‌ی فراخوانی این translate tableها دقت کنید:

```
<!-- section 1: Translate Table Called by ns1 namespace -->
<h1 translate>ns1.HEADER</h1>
<h2 translate>ns1.SUBHEADER</h2>

<!-- section 2: Translate Table Called by ns2 namespace -->
<h1 translate>ns2.HEADER</h1>
<h2 translate>ns2.SUBHEADER</h2>
```

به همین سادگی می‌توان تمامی بخش‌ها را با namespaceهای مختلف در translate table قرار داد.

در بخش بعدی (پایانی) شش قابلیت دیگر angular translate که شامل فراخوانی translate table از یک فایل JSON، فراخوانی فایل‌های translate table به صورت lazy load و تغییر زبان بخشی از صفحه به صورت پویا هستند، بررسی خواهند شد.

فایل پروژه: AngularJs-Translate-BestPractices.zip