عنوان: **ساخت یک بلاگ ساده با Ember.js، قسمت دوم**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۷ ۸:۵
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, jQuery, SPA, EmberJS

پس از تهیه ساختار اولیه‌ی بلاگی مبتنی بر ember.js در قسمت قبل ، در ادامه قصد داریم امکانات تعاملی را به آن اضافه کنیم. بنابراین کار را با تعریف کنترلرها که تعیین کننده‌ی رفتار برنامه هستند، ادامه می‌دهیم.

اضافه کردن دکمه‌ی More info به صفحه‌ی About و مدیریت کلیک بر روی آن

فایل Scripts\Templates\about.hbs را گشوده و سپس محتوای فعلی آن را به نحو ذیل تکمیل کنید:

```
<h2>About Ember Blog</h2>

<p>Bla bla bla!</p>

<button class="btn btn-primary" {{action 'showRealName' }}>more info</button>
```

در ember.js اگر قصد مدیریت عملی را که قرار است توسط کلیک بر روی المانی رخ دهد، داشته باشیم، می‌توان از handlebar helper ایی به نام action استفاده کرد. سپس برای تهیه کدهای مرتبط با آن، این اکشن را باید در کنترلر متناظر با route جاری (مسیریابی about) اضافه کنیم.
به همین جهت فایل جدید Scripts\Controllers\about.js را در پوشه‌ی کنترلرهای سمت کاربر اضافه کنید (نام آن با نام مسیریابی یکی است)؛ با این محتوا:

```
Blogger.AboutController = Ember.Controller.extend({
  actions: {
   showRealName: function () {
    alert("You clicked at showRealName of AboutController.");
   }
  }
});
```

کنترلرها به صورت یک خاصیت جدید به شیء Application برنامه اضافه می‌شوند. مطابق اصول نامگذاری ember.js، نام خاصیت کنترلر با حروف بزرگ متناظر با route آن شروع می‌شود و به نام Controller ختم خواهد شد. به این ترتیب ember.js هرگاه قصد پردازش مسیریابی about را داشته باشد، می‌داند که باید از کدام شیء جهت پردازش اعمال کاربر استفاده کند.
در ادامه این خاصیت را با تهیه یک زیرکلاس از کلاس پایه Controller تهیه شده توسط ember.js مقدار دهی می‌کنیم. به این ترتیب به کلیه امکانات این کلاس پایه دسترسی خواهیم داشت؛ به علاوه می‌توان ویژگی‌های سفارشی را نیز به آن افزود. برای مثال در اینجا در قسمت actions آن، دقیقا مطابق نام اکشنی که در فایل about.hbs تعریف کرده‌ایم، یک متد جدید اضافه شده‌است.

پس از تعریف کنترلر about.js نیاز است مدخل متناظر با آن‌را به فایل index.html برنامه نیز در انتهای تعاریف موجود، اضافه کرد:

```
<script src="Scripts/Controllers/about.js" type="text/javascript"></script>
```

اکنون یکبار برنامه را اجرا کرده و در صفحه‌ی about بر روی دکمه‌ی more info کلیک کنید.

اضافه کردن دکمه‌ی ارسال پیام خصوصی به صفحه‌ی Contact و مدیریت کلیک بر روی آن

در ادامه به قالب فعلی Scripts\Templates\contact.hbs یک دکمه را جهت ارسال پیام خصوصی اضافه می‌کنیم.

```
<h1>Contact</h1>
<div class="row">
  <div class="col-md-6">
   <p>
    Want to get in touch?
    <ul>
      <li>{{#link-to 'phone'}}Phone{{/link-to}}</li>
      <li>{{#link-to 'email'}}Email{{/link-to}}</li>
    </ul>
   </p>

   <p>
    Or, click here to send a secret message:
   </p>
   <button class="btn btn-primary" {{action 'sendMessage' }}>Send message</button>
  </div>
  <div class="col-md-6">
   {{outlet}}
  </div>
</div>
```

سپس برای مدیریت اکشن جدید sendMessage نیاز است کنترلر آن‌را نیز تعریف کنیم. با توجه به نام مسیریابی جاری، نام این کنترلر نیز contact خواهد بود. برای این منظور ابتدا فایل جدید Scripts\Controllers\contact.js را اضافه نمائید؛ با این محتوا:

```
Blogger.ContactController = Ember.Controller.extend({
  actions: {
   sendMessage: function () {
    var message = prompt('Type your message here:');
   }
  }
});
```

همچنین مدخل متناظر با فایل contact.js نیز باید به صفحه‌ی index.html اضافه شود:

```
<script src="Scripts/Controllers/contact.js" type="text/javascript"></script>
```

نمایش تصویری تعاملی در صفحه‌ی about

تا اینجا با نحوه‌ی تعریف اکشن‌ها در قالب‌ها و مدیریت آن‌ها توسط کنترلرهای متناظر آشنا شدیم. در ادامه قصد داریم با اصول binding اطلاعات در ember.js آشنا شویم. برای مثال فرض کنید می‌خواهیم دکمه‌ای را در صفحه‌ی about قرار داده و با کلیک بر روی آن، لوگوی ember.js را که به صورت یک تصویر مخفی در صفحه قرار دارد، نمایان کنیم. برای اینکار نیاز است خاصیتی را در کنترلر متناظر، تعریف کرده و سپس آن‌را به template جاری bind کرد.
برای این منظور فایل Scripts\Templates\about.hbs را گشوده و تعاریف موجود آن‌را به نحو ذیل تکمیل کنید:

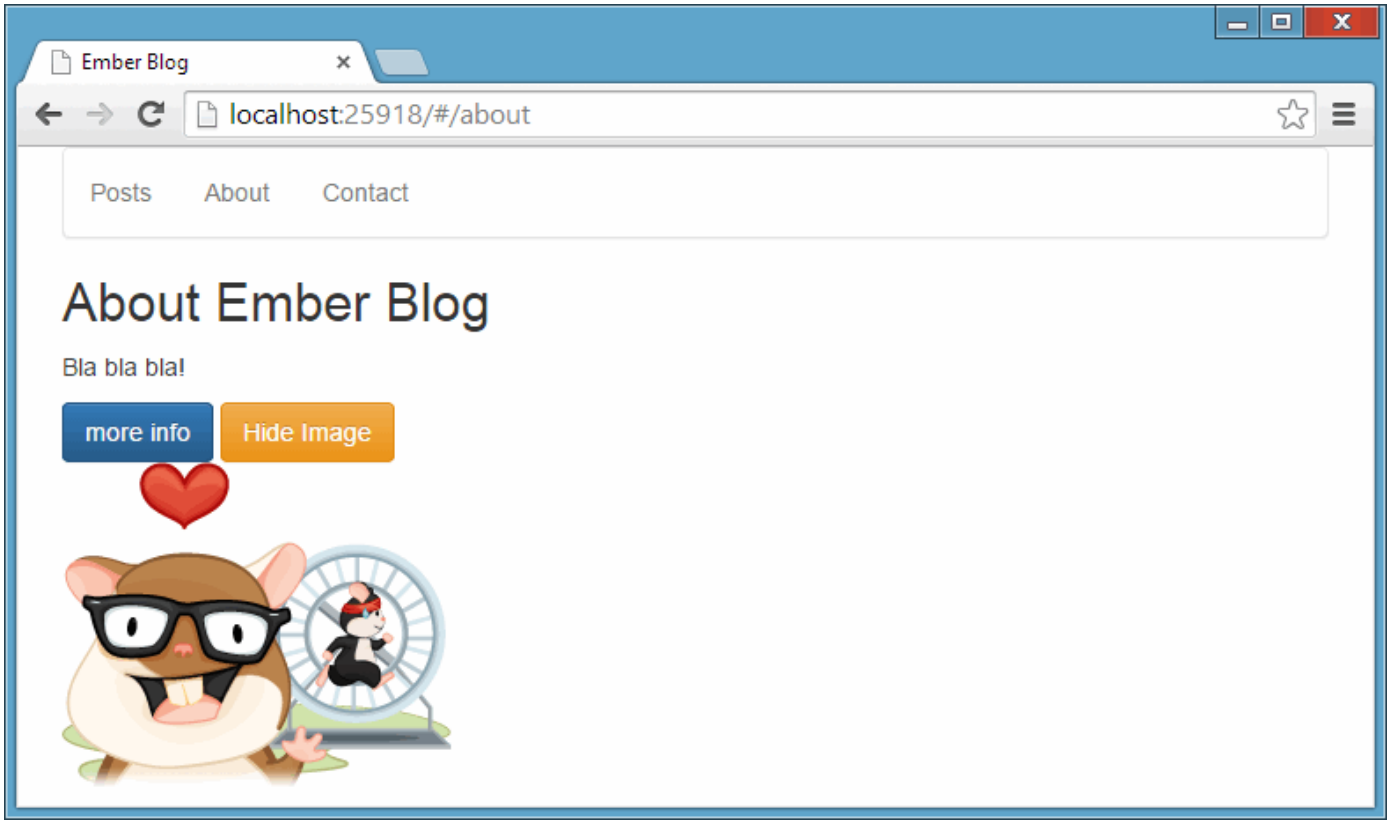
```
<h2>About Ember Blog</h2>
<p>Bla bla bla!</p>
<button class="btn btn-primary" {{action 'showRealName' }}>more info</button>

{{#if isAuthorShowing}}
<button class="btn btn-warning" {{action 'hideAuthor' }}>Hide Image</button>
<p><img src="Content/images/ember-productivity-sm.png"></p>
{{else}}
<button class="btn btn-info" {{action 'showAuthor' }}>Show Image</button>
{{/if}}
```

در اینجا بر اساس مقدار خاصیت isAuthorShowing تصمیم گیری خواهد شد که آیا تصویر لوگوی ember.js نمایش داده شود یا خیر. همچنین دو اکشن نمایش و مخفی کردن تصویر نیز اضافه شده‌اند که با کلیک بر روی هر کدام، سبب تغییر وضعیت خاصیت isAuthorShowing خواهیم شد.
کنترلر about (فایل Scripts\Controllers\about.js) جهت مدیریت این خاصیت جدید، به همراه دو اکشن تعریف شده، اینبار به نحو ذیل تغییر خواهد یافت:

```
Blogger.AboutController = Ember.Controller.extend({
  isAuthorShowing: false,
  actions: {
   showRealName: function () {
    alert("You clicked at showRealName of AboutController.");
   },
   showAuthor: function () {
    this.set('isAuthorShowing', true);
   },
   hideAuthor: function () {
    this.set('isAuthorShowing', false);
   }
  }
});
```

ابتدا خاصیت isAuthorShowing به کنترلر اضافه شده‌است. از این خاصیت بار اولی که مسیر http://localhost:25918/#/about توسط کاربر درخواست می‌شود، استفاده خواهد شد.
سپس در دو متد showAuthor و hideAuthor که به اکشن‌های دو دکمه‌ی جدید تعریف شده در قالب about متصل خواهند شد، نحوه‌ی تغییر مقدار خاصیت isAuthorShowing را توسط متد set ملاحظه می‌کنید.
این قسمت مهم‌ترین تفاوت ember.js با jQuery است. در jQuery مستقیما المان‌های صفحه در همانجا تغییر داده می‌شوند. در ember.js منطق مدیریت کننده‌ی رابط کاربری و کدهای قالب متناظر با آن از هم جدا شده‌اند تا بتوان یک برنامه‌ی بزرگ را بهتر مدیریت کرد. همچنین در اینجا مشخص است که هر قسمت و هر فایل، چه ارتباطی با سایر اجزای تعریف شده دارد و چگونه به هم متصل شده‌اند و اینبار شاهد انبوهی از کدهای جاوا اسکریپتی مخلوط بین المان‌های HTML صفحه نیستیم.

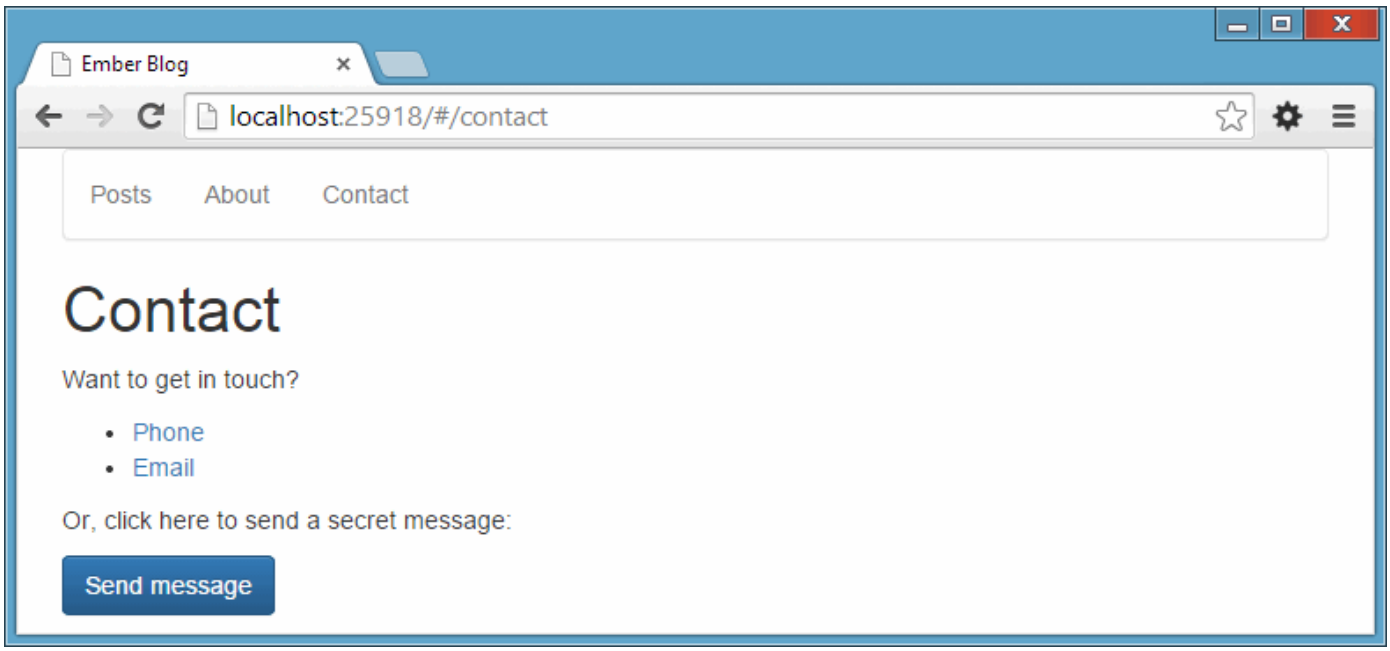


نمایش پیامی به کاربر پس از ارسال پیام خصوصی در صفحه‌ی تماس با ما

قصد داریم ویژگی مشابهی را به صفحه‌ی contact نیز اضافه کنیم. اگر کاربر بر روی دکمه‌ی ارسال پیام کلیک کرد، پیام تشکری به همراه عددی ویژه به او نمایش خواهیم داد.
برای این‌کار قالب Scripts\Templates\contact.hbs را به نحو ذیل تکمیل کنید:

```
<h1>Contact</h1>
<div class="row">
  <div class="col-md-6">
   <p>
    Want to get in touch?
    <ul>
      <li>{{#link-to 'phone'}}Phone{{/link-to}}</li>
      <li>{{#link-to 'email'}}Email{{/link-to}}</li>
    </ul>
   </p>
   {{#if messageSent}}
   <p>
    Thank you. Your message has been sent.
    Your confirmation number is {{confirmationNumber}}.
   </p>
   {{else}}
   <p>
    Or, click here to send a secret message:
   </p>
   <button class="btn btn-primary" {{action 'sendMessage' }}>Send message</button>
   {{/if}}
  </div>
  <div class="col-md-6">
   {{outlet}}
  </div>
</div>
```

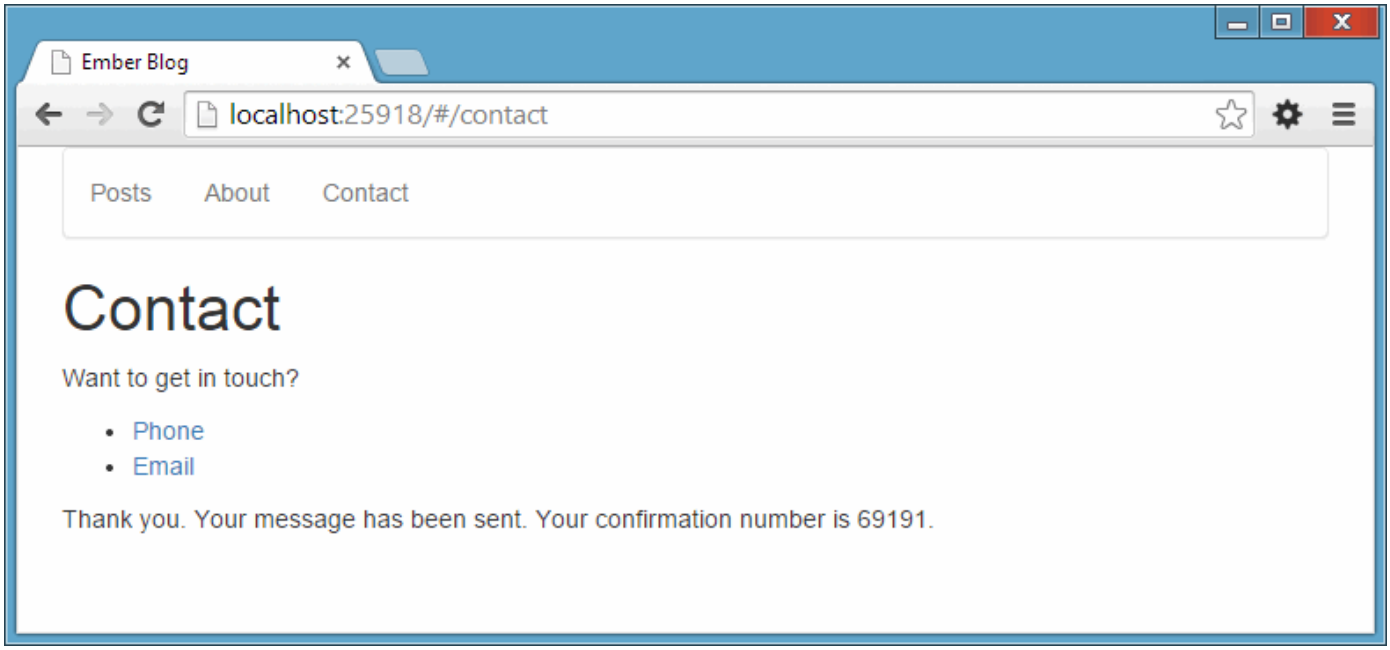
در آن شرط بررسی if messageSent اضافه شده‌است؛ به همراه نمایش confirmationNumber در انتهای پیام تشکر.



برای تعریف منطق مرتبط با این خواص، به کنترلر contact واقع در فایل Scripts\Controllers\contact.js مراجعه کرده و آن‌را به نحو ذیل تغییر می‌دهیم:

```
Blogger.ContactController = Ember.Controller.extend({
  messageSent: false,
  actions: {
   sendMessage: function () {
    var message = prompt('Type your message here:');
    if (message) {
      this.set('confirmationNumber', Math.round(Math.random() * 100000));
      this.set('messageSent', true);
    }
   }
  }
});
```

همانطور که مشاهده می‌کنید، مقدار اولیه خاصیت messageSent مساوی false است. بنابراین در قالب contact.hbs قسمت else شرط نمایش داده می‌شود. اگر کاربر پیامی را وارد کند، خاصیت confirmationNumber به یک عدد اتفاقی و خاصیت messageSent به true تنظیم خواهد شد. به این ترتیب اینبار به صورت خودکار پیام تشکر به همراه عددی اتفاقی، به کاربر نمایش داده می‌شود.



بنابراین به صورت خلاصه، کار کنترلر، مدیریت منطق نمایشی برنامه است و برای اینکار حداقل دو مکانیزم را ارائه می‌دهد: اکشن‌ها و خواص. اکشن‌ها بیانگر نوعی رفتار هستند؛ برای مثال نمایش یک popup و یا تغییر مقدار یک خاصیت. مقدار خواص را می‌توان مستقیما در صفحه نمایش داد و یا از آن‌ها جهت پردازش عبارات شرطی و نمایش قسمت خاصی از قالب جاری نیز می‌توان کمک گرفت.

کدهای کامل این قسمت را از اینجا می‌توانید دریافت کنید:

EmberJS03_02.zip