عنوان: **آشنایی با Leaflet**
نویسنده: شاهین کیاست
تاریخ: ۱۳۹۳/۱۱/۱۳ ۱۹:۲۰
آدرس: www.dotnettips.info
گروه‌ها: JavaScript, GIS, leafletjs, Map

مقدمه

سیستم‌های جغرافیایی و GIS اهمیت زیادی در زندگی روزمره‌ی ما دارند. GIS به نرم افزار یا سخت افزاری اطلاق می‌شود که کاربر را قادر می‌سازد تا به ذخیره، بازیابی و تجزیه و تحلیل داده‌های جغرافیایی (Spatial) بپردازد. یکی از پایه‌های نرم افزارهای GIS، نقشه و نمایش اطلاعات بر روی نقشه می‌باشد. به طور حتم در وب سایت‌ها مشاهده کرده‌اید که آدرس یک شرکت بر روی نقشه نمایان می‌شود یا به عنوان مثالی دیگر سرویس دهنده‌های اینترنت از نقشه برای نمایش میزان و کیفیت آنتن دهی در محله‌های مختلف یک شهر استفاده می‌کنند.

برای نمایش نقشه در نرم افزارهای تحت وب کتابخانه‌های JavaScript ایی زیادی وجود دارند. این مطلب به معرفی کتاب خانه‌ی کدباز و رایگان leaflet می‌پردازد. leaflet یک کتابخانه‌ی مدرن JavaScript برای کار با نقشه می‌باشد. از خصوصیات بارز این کتابخانه پشتیبانی بسیار خوب آن از موبایل و دستگاههای لمسی است. Leaflet تنها 33 کیلوبایت حجم دارد و ویژگی‌های آن اغلب نیازهای‌های توسعه دهندگان را برای پیاده سازی نرم افزارهای مبتنی بر نقشه پوشش می‌دهد. از مزایای این کتابخانه می‌توان به مشارکت جامعه‌ی بزرگ توسعه دهندگان، سورس خوانا و تمیز، مستندات خوب و تعداد زیادی پلاگین برای آن اشاره کرد.

آماده سازی صفحه

برای استفاده از Leaflet ابتدا باید فایل Style و JavaScript کتابخانه را ارجاع داد:

```
<script src="http://cdn.leafletjs.com/leaflet-0.7.3/leaflet.js"></script>
 <link rel="stylesheet" href="http://cdn.leafletjs.com/leaflet-0.7.3/leaflet.css" />
```

سپس یک div با یک Id مشخص را به صفحه اضافه می‌کنیم. div مورد نظر باید از ارتفاع مشخصی برخوردار باشد که به سادگی با style زیر میسر می‌گردد:

```
#map { height: 600px; }
```

پس از انجام مقدمات اکنون می‌توان یک نقشه را با تنظیمات دلخواهی در div تعریف شده نمایش داد.

تنظیمات اولیه نقشه

با کد زیر ابتدا یک وهله از شیء map ایجاد می‌شود:

```
var map = L.map('map').setView([29.6760859,52.4950737], 13);
```

همانطور که مشاهده می‌شود شناسه‌ی div تعریف شده از طریق سازنده به map پاس داده شده است و سپس به کمک تابع setView به محل مختصات جغرافیایی مورد نظر با زوم پیشفرض 13 نمایش داده می‌شود. طراحی Leaflet به صورتی است که استفاده از متدهای زنجیروار (chainable) را میسر می‌سازد. به عنوان نمونه در کد بالا تابع setView یک شیء map را بر می‌گرداند و توسعه دهنده می‌تواند از توابع دیگر مقدار بازگشتی استفاده کند. این مورد از نظر طراحی شبیه به jQuery می‌باشد.

اگر Google Maps را مشاهده کنید، متوجه می‌شوید که یک نقشه، به صورت مستطیل مستطیل، بارگزاری می‌شود. به این مستطیل‌ها Tile گفته می‌شود. tileها همان فایل‌های png هستند و درواقع به ازای زوم‌های مختلف در محل‌های مختلف، tileهای متفاوتی با شناسه‌ی مشخصی وجود دارند. تصویر زیر نقشه‌ی Google می‌باشد؛ قبل از اینکه tileها بارگزاری شوند. اگر با دقت نگاه کنید مستطیل‌های بزرگ و کوچکی را مشاهده می‌کنید که قسمت‌های مختلف یک نقشه یا همان تایل می‌باشند.

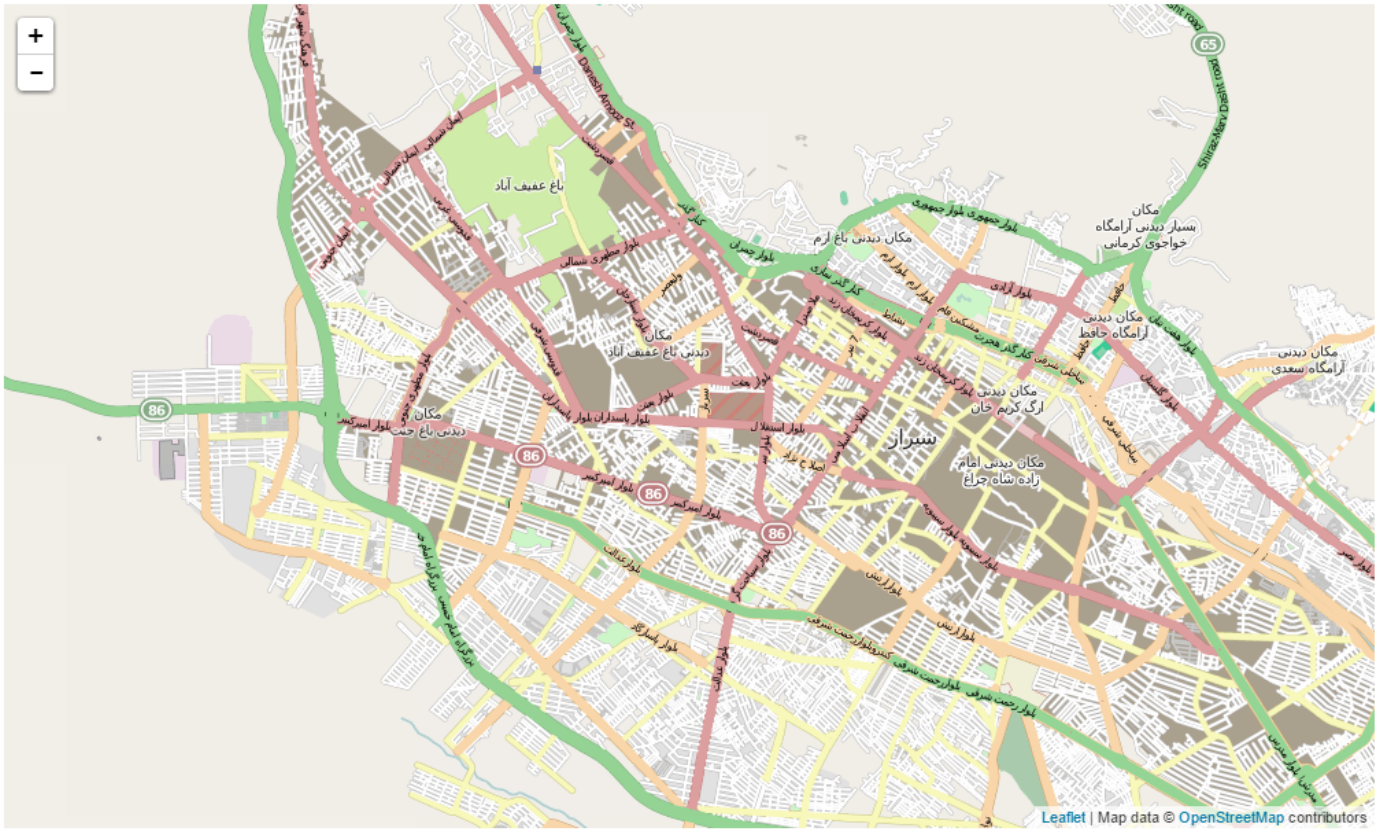
پس برای نمایش یک نقشه نیاز است tileها را از یک منبع، در اختیار نقشه قرار داد. این منبع می‌تواند یک وب سرویس باشد.

پس از تعریف اولیه، نیاز است یک Tile Layer ایجاد کرده و آن را به نقشه اضافه کرد:

```
var osmUrl='http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png';
var osmAttrib='Map data © <a href="http://openstreetmap.org">OpenStreetMap</a> contributors';
var osm = new L.TileLayer(osmUrl, { maxZoom: 18, attribution: osmAttrib}).addTo(map);
```

در کد بالا ابتدا آدرس tile server تعریف شده است. در این مثال از سرویس OpenStreetMap برای تهیه‌ی Tileها استفاده شده است. سپس لینک سرویس دهنده، به همراه متن attribution(نوشته‌ای که در زیر نقشه قرار می‌گیرد) به شیء TileLayer پاس داده شد و شیء ایجاد شده از طریق متد addTo به شیء map اضافه شده است.

نتیجه‌ی کار در مرورگر اینگونه خواهد بود:

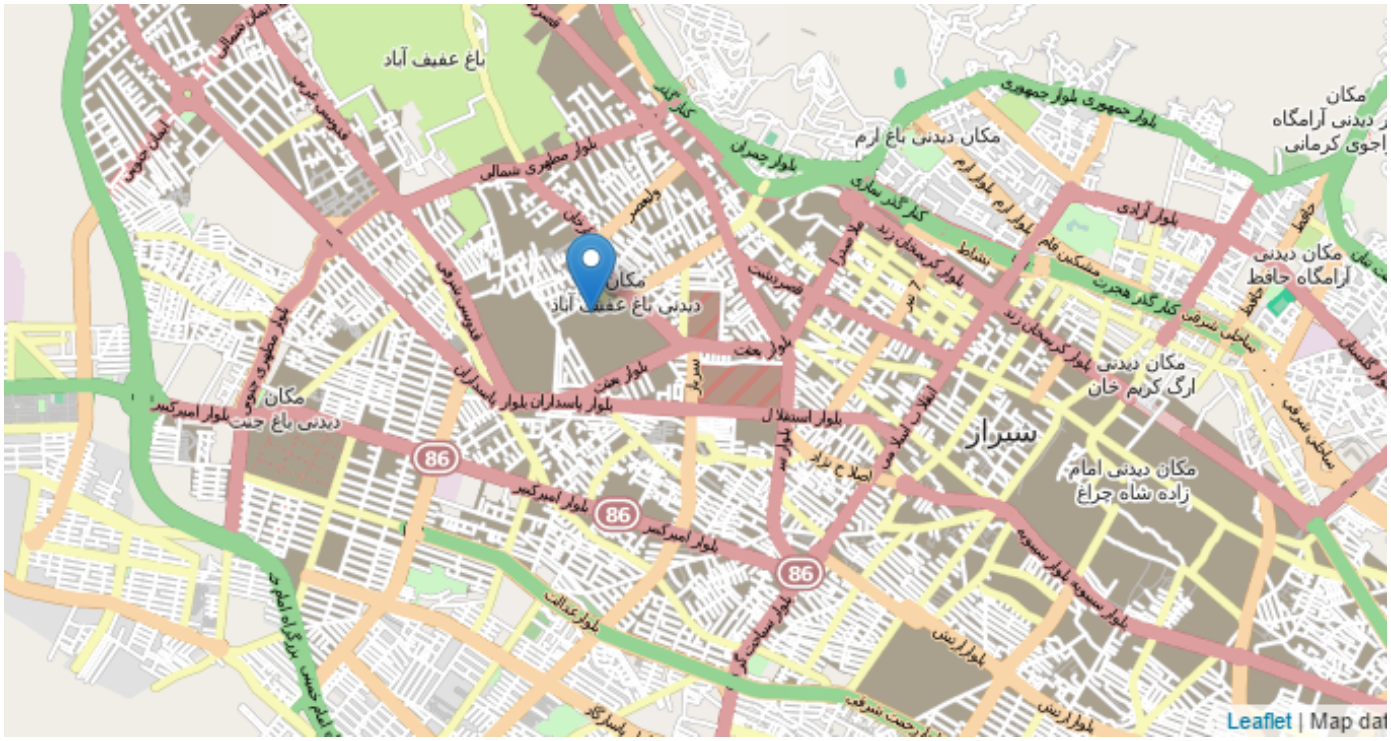


Marker، دایره و چندضلعی

در کنار نمایش Tileها می‌توان اشکال گرافیکی نیز به نقشه اضافه کرد؛ مثل مارکر(نقطه)، مستطیل، دایره و یا یک Popup. اضافه کردن یک Marker به سادگی، با کد زیر صورت می‌پذیرد:

```
var marker = L.marker([29.623116,52.497856]).addTo(map);
```

محل مورد نظر به شیء مارکر پاس داده شده و مقدار بازگشتی به map اضافه شده است.



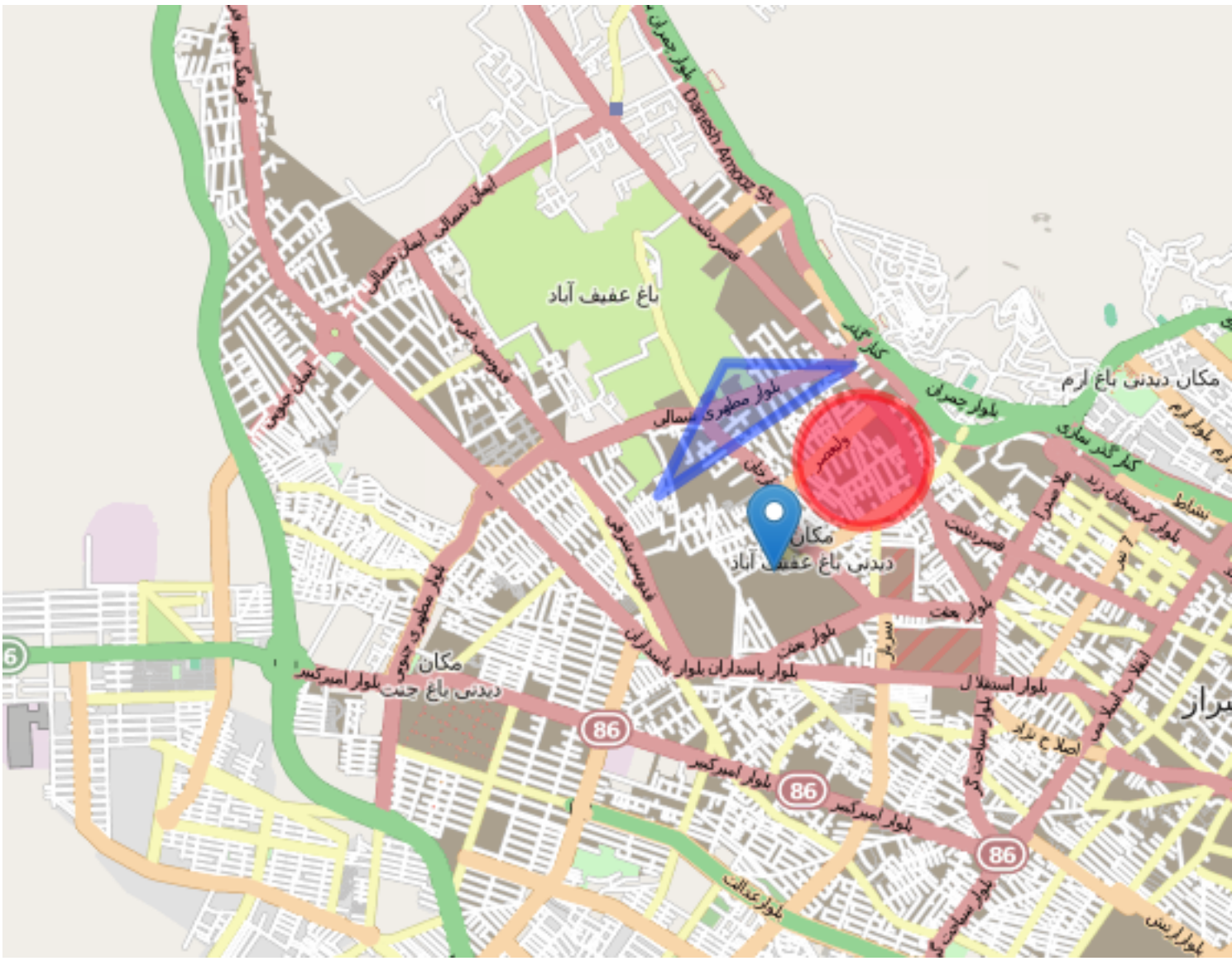
نمایش چند ضلعی و دایره هم کار ساده ای است. برای دایره باید ابتدا مختصات مرکز دایره و شعاع به متر را به L.circle پاس داد:

```
var circle = L.circle([29.6308217,52.5048021], 500, {
    color: 'red',
    fillColor: '#f03',
    fillOpacity: 0.5
}).addTo(map);
```

در کد بالا علاوه بر محل و اندازه دایره، رنگ محیط، رنگ داخل و شفافیت (opacity) نیز مشخص شده‌اند.

برای چند ضلعی‌ها می‌توان به این صورت عمل کرد:

```
var polygon = L.polygon([
[29.628453, 52.488838],
[29.637368, 52.493987],
[29.637168, 52.503987]
]).addTo(map);
```



کار کردن با Popup ها

از Popup می‌توان برای نمایش اطلاعات اضافه‌ای بر روی یک محل خاص یا یک عنوان به مانند Marker استفاده کرد. برای مثال می‌توان اطلاعاتی درباره‌ی محل یک Marker یا دایره نمایش داد. در هنگام ایجاد marker، دایره و چندضلعی مقادیر بازگشتی در متغیرهای جدایی ذخیره شدند. اکنون می‌توان به آن اشیاء یک popup اضافه کرد:

```
marker.bindPopup("باغ عفیف آباد").openPopup();
circle.bindPopup("I am a circle.");
polygon.bindPopup("I am a polygon.");
```

به علت اینکه openPopup برای Marker صدا زده شده، به صورت پیشفرض popup را نمایش می‌دهد. اما برای بقیه، نمایش با کلیک خواهد بود. البته الزاما نیازی نیست که popup روی یک شیء نمایش داده شود، می‌توان popupهای مستقلی نیز ایجاد کرد:

```
var popup = L.popup()
    .setLatLng([51.5, -0.09])
    .setContent("I am a standalone popup.")
    .openOn(map);
```