Leaflet交流群：331437754 由fengl整理

星期五，4月 25 2014，9:51 下午

This reference reflects **Leaflet 0.6**.
Docs for the previous stable version (0.5) are available in the source form (see instructions for running docs).

L.Map

API各种类中的核心部分，用来在页面中创建地图并操纵地图.

使用 example

```
// initialize the map on the "map" div with a given center and zoom
```

```
var map = L.map('map', {
        center: [51.505, -0.09],
        zoom: 13
});
```

构造器

构造器	使用	描述
`L.Map( <HTMLElement|String> id, <Map options> options? )`	`new L.Map(…)` `L.map(…)`	通过div元素和带有地图选项的描述的文字对象来实例化一个地图对象，其中文字对象是可选的。

Options

Map State Options地图状态选项

选项	类型	默认值	描述
center	LatLng	null	初始化地图的地理中心.
zoom	Number	null	初始化地图的缩放.
layers	ILayer[]	null	初始化后加载到地图上的图层.
minZoom	Number	null	地图的最小视图。可以重写地图图层的minZoom.
maxZoom	Number	null	地图的最大视图。可以重写地图图层的maxZoom.
maxBounds	LatLngBounds	null	当这个选项被设置后，地图被限制在给定的地理边界内，当用户平移将地图拖动到视图以外的范围时会出现弹回的效果， 并且也不允许缩小视图到给定范围以外的区域（这取决于地图的尺寸）。 使用setMaxBounds方法可以动态地设置这种约束.
crs	CRS	L.CRS.EPSG3857	使用的坐标系，当你不确定坐标系是什么时请不要更改.

Interaction Options交互操作

选项	类型	默认值	描述
dragging	Boolean	true	决定地图是否可被鼠标或触摸拖动.
touchZoom	Boolean	true	决定地图是否可被两只手指触摸拖拽缩放.
scrollWheelZoom	Boolean	true	决定地图是否被被鼠标滚轮滚动缩放.
doubleClickZoom	Boolean	true	决定地图是否可被双击缩放.
boxZoom	Boolean	true	决定地图是否可被缩放到鼠标拖拽出的矩形的视图，鼠标拖拽时需要同时按住shift键.
tap	Boolean	true	Enables mobile hacks for supporting instant taps (fixing 200ms click delay on iOS/Android) and touch holds (fired as

			contextmenu events).
tapTolerance	Number	15	The max number of pixels a user can shift his finger during touch for it to be considered a valid tap.
trackResize	Boolean	true	确定地图在窗口尺寸改变时是否可以自动处理浏览器以更新视图.
worldCopyJump	Boolean	false	当这个选项可用时，当你平移地图到其另一个领域时会被地图捕获到，并无缝地跳转到原始的领域以保证所有标注、矢量图层之类的覆盖物仍然可见.
closePopupOnClick	Boolean	true	当你不想用户点击地图关闭消息弹出框时，请将其设置为false .

Keyboard Navigation Options

选项	类型	默认值	描述
keyboard	Boolean	true	聚焦到地图且允许用户通过键盘的方向键和+/-键来漫游地图.
keyboardPanOffset	Number	80	确定按键盘方向键时地图平移的像素.
keyboardZoomOffset	Number	1	确定键盘+ or -键对于的缩放级数.

Panning Inertia Options

选项	类型	默认值	描述
inertia	Boolean	true	如果该选项可用，在拖动和在某一时间段内持续朝同一方向移动建有动力的地图时，会有惯性的效果.
inertiaDeceleration	Number	3000	确定惯性移动减速的速率，单位是像素每秒的二次方^2.
inertiaMaxSpeed	Number	1500	惯性移动的最大速度，单位是像素每秒.
inertiaThreshold	Number	depends	放开鼠标或是触摸来停止惯性移动与移动停止之间的毫秒数.

Control options

选项	类型	默认值	描述
zoomControl	Boolean	true	确定zoom control是否默认加载在地图上 .
attributionControl	Boolean	true	确定attribution control是否默认加载在地图上.

Animation options

选项	类型	默认值	描述
fadeAnimation	Boolean	depends	确定瓦片淡出动画是否可用。通常默认在所有浏览器中都支持CSS3转场，android例外.

zoomAnimation	Boolean	depends	确定瓦片缩放动画是否可用。通常默认在所有浏览器中都支持CSS3转场，android例外.
zoomAnimationThreshold	Number	4	Won't animate zoom if the zoom difference exceeds this value.
markerZoomAnimation	Boolean	depends	确定注记的缩放是否随地图缩放动画而播放，如果被禁用，注记在动画中拉长时会消失。通常默认在所有浏览器中都支持CSS3转场，android例外.

Events

You can subscribe to the following events using these methods.

Event	Data	描述
click	MouseEvent	用户点击或触摸地图时触发.
dblclick	MouseEvent	用户双击或连续两次触摸地图时触发.
mousedown	MouseEvent	用户按下鼠标按键时触发.
mouseup	MouseEvent	用户按下鼠标按键时触发.
mouseover	MouseEvent	鼠标进入地图时触发.
mouseout	MouseEvent	鼠标离开地图时触发.
mousemove	MouseEvent	鼠标在地图上移动时触发.
contextmenu	MouseEvent	当用户在地图上按下鼠标右键时触发，如果有监听器在监听这个时间，则浏览器默认的情景菜单被禁用.
focus	Event	当用户在地图上进行标引、点击或移动时进行聚焦.
blur	Event	当地图失去焦点时触发.
preclick	MouseEvent	当鼠标在地图上点击之前触发。有时会在点击鼠标时，并在已存在的点击事件开始处理之前想要某件事情发生时用得到.
load	Event	当地图初始化时触发。（当地图的中心点和缩放初次设置时）.
unload	Event	Fired when the map is destroyed with remove method.
viewreset	Event	当地图需要重绘内容时触发。（通常在地图缩放和载入时发生）这对于创建用户自定义的叠置图层非常有用.
movestart	Event	地图视图开始改变时触发。（比如用户开始拖动地图）.
move	Event	所有的地图视图移动时触发.
moveend	Event	当地图视图结束改变时触发。（比如用户停止拖动地图）.
dragstart	Event	用户开始拖动地图时触发.

drag	Event	用户拖动地图时不断重复地触发.
dragend	Event	用户停止拖动时触发.
zoomstart	Event	当地图缩放即将发生时触发。（比如缩放动作开始前）.
zoomend	Event	当地图缩放时触发.
zoomlevelschange	Event	Fired when the number of zoomlevels on the map is changed due to adding or removing a layer.
resize	ResizeEvent	Fired when the map is resized.
autopanstart	Event	打开弹出窗口时地图开始自动平移时触发.
layeradd	LayerEvent	当一个新的图层添加到地图上时触发.
layerremove	LayerEvent	当一些图层从地图上移除时触发.
baselayerchange	LayerEvent	当通过layer control改变基础图层时触发.
overlayadd	LayerEvent	Fired when an overlay is selected through the layer control.
overlayremove	LayerEvent	Fired when an overlay is deselected through the layer control.
locationfound	LocationEvent	当地理寻址成功时触发（使用locate方法）
locationerror	ErrorEvent	当地理寻址错误时触发（使用locate方法）
popupopen	PopupEvent	当弹出框打开时触发（使用openPopup方法）
popupclose	PopupEvent	当弹出框关闭时触发（使用closePopup方法）

地图状态修改

方法	返回值	描述
setView(<LatLng> center, <Number> zoom, <zoom/pan options> options?)	this	设定地图（设定其地理中心和缩放）.
setZoom(<Number> zoom, <zoom options> options?)	this	设定地图的缩放.
zoomIn(<Number> delta?, <zoom options> options?)	this	通过delta变量放大地图的级别，1是delta的默认值.
zoomOut(<Number> delta?, <zoom	this	通过delta变量缩小地图的级别，1是delta的默认值.

options> options?)		
setZoomAround(<LatLng> latlng, <Number> zoom, <zoom options> options?)	this	Zooms the map while keeping a specified point on the map stationary (e.g. used internally for scroll zoom and double-click zoom).
fitBounds(<LatLngBounds> bounds, <fitBounds options> options?)	this	将地图视图尽可能大地设定在给定的地理边界内.
fitWorld(<fitBounds options> options?)	this	将地图视图尽可能大地设定在包含全部地域的级别上.
panTo(<LatLng> latlng, <pan options> options?)	this	将地图平移到给定的中心。如果新的中心点在屏幕内与现有的中心点不同则产生平移动作.
panInsideBounds(<LatLngBounds> bounds)	this	平移地图到坐落于给定边界最接近的视图内.
panBy(<Point> point, <pan options> options?)	this	通过给定的像素值对地图进行平移.
invalidateSize(<Boolean> options?, <zoom/pan options> options?)	this	检查地图容器的大小是否改变并更新地图，如果是这样的话，在动态改变地图大小后调用，如果animate是true的话，对地图进行更新.
setMaxBounds(<LatLngBounds> bounds, <zoom/pan options> options?)	this	将地图限定在给定的边界内 (map maxBounds).
locate(<Locate options> options?)	this	用地理定位接口Geolocation API获取用户位置信息，在成功定位或定位出错产生locationerror后解除location-found事件与定位数据，且将地图视图设定到检测的确切的用户的位置（如果定位失败则回到地域视图）。在Locate options中有更多详细内容。
stopLocate()	this	Stops watching location previously initiated by `map.locate({watch: true})` and aborts resetting the map view if `map.locate` was called with `{setView: true}`.
remove()	this	Destroys the map and clears all related event listeners.

获取地图状态

方法	返回值	描述
getCenter()	LatLng	返回地图视图的地理中心.

getZoom()	Number	获取地图视图现在所处的缩放级别.
getMinZoom()	Number	返回地图最小的缩放级别.
getMaxZoom()	Number	返回地图最大的缩放级别.
getBounds()	LatLngBounds	返回地图视图的经纬度边界.
getBoundsZoom(<LatLngBounds> bounds, <Boolean> inside?)	Number	返回适应整个地图视图边界的最大缩放级别。如果inside的设置时true，这个方法返回适应整个地图视图边界的最小缩放级别.
getSize()	Point	返回现有地图容器的大小.
getPixelBounds()	Bounds	返回地图视图在像素投影坐标系下的边界。（很多时候对用户自定义图层和叠加很有用）.
getPixelOrigin()	Point	返回地图图层像素投影坐标系下的左上角的点。（很多时候对用户自定义图层和叠加很有用）.

图层控制/h3>

方法	返回值	描述
addLayer(<ILayer> layer, <Boolean> insertAtTheBottom?)	this	将图层添加到地图上。如果insertAtTheBottom的选项为true，图层添加时在所以图层之下。（在切换基底图时比较有用）.
removeLayer(<ILayer> layer)	this	将图层在地图上移除.
hasLayer(<ILayer> layer)	Boolean	如果添加的图层是当前图层则返回true.
openPopup(<Popup> popup)	this	当关闭前一个弹出框时弹出指定的对话框。（确定在同一时间只有一个打开并可用）.
openPopup(<String> html | <HTMLElement> el, <LatLng> latlng, <Popup options> options?)	this	Creates a popup with the specified options and opens it in the given point on a map.
closePopup(<Popup> popup?)	this	关闭openPopup打开的弹出框.
addControl(<IControl> control)	this	在地图上添加控制选项.
removeControl(<IControl> control)	this	在地图上移除控制选项.

转换方法

方法	返回值	描述
latLngToLayerPoint(<LatLng> latlng)	Point	返回地图图层上与地理坐标相一致的点。（在地图上进行位置叠加时比较有用）.
layerPointToLatLng(<Point> point)	LatLng	返回给定地图上点的地理坐标系.

containerPointToLayerPoint(<Point> point)	Point	将于地图容器相关的点转换为地图图层相关的点.
layerPointToContainerPoint(<Point> point)	Point	将地图图层相关的点转换为地图容器相关的点.
latLngToContainerPoint(<LatLng> latlng)	Point	返回与给定地理坐标系相符的地图容器的点.
containerPointToLatLng(<Point> point)	LatLng	返回给定地理容器点的地理坐标.
project(<LatLng> latlng, <Number> zoom?)	Point	将地理坐标投影到指定缩放级别的像素坐标系中.
unproject(<Point> point, <Number> zoom?)	LatLng	将像素坐标系投影到指定缩放级别的地理坐标系中。（默认为当前的缩放级别）.
mouseEventToContainerPoint(<MouseEvent> event)	Point	返回鼠标点击事件对象的像素坐标（与地图左上角相关）.
mouseEventToLayerPoint(<MouseEvent> event)	Point	返回鼠标点击事件对象的像素坐标（与地图图层相关）.
mouseEventToLatLng(<MouseEvent> event)	LatLng	返回鼠标点击事件对象的地理坐标.

其他方法

方法	返回值	描述
getContainer()	HTMLElement	返回地图容器对象.
getPanes()	MapPanes	返回不同地图对象的边框（叠加渲染).
whenReady(<Function> fn, <Object> context?)	this	当地图的位置和缩放初始化好或是时间发生之后，运行给定的回调方法，通常传递一个函数内容.

位置选项

选项	类型	默认值	描述
watch	Boolean	false	如果该值为真，则开始利用W3C的watchPosition方法监听位置变化情况（而不是指监听一次）。你可以通过map.stoplocate()方法来停止监听.
setView	Boolean	false	如果该值为真，则通过自动将地图视图定位到用户一定精度范围内的位置，如果地理定位失败则显示全部地图.
maxZoom	Number	Infinity	在使用setView选项时视图缩放的最大级别.
timeout	Number	10000	在触发locationerror事件之前等待地理定位的毫秒为单位的

			时间.
maximumAge	Number	0	位置监听的最大生命周期。如果比最后定位回复后毫秒用时短，则locate返回一个缓存位置.
enableHighAccuracy	Boolean	false	开启高精度，参加 W3C SPEC的描述.

Zoom/pan options

选项	类型	默认值	描述
reset	Boolean	false	If true, the map view will be completely reset (without any animations).地图视图将完全复位（没有任何动画）。
pan	pan options	-	Sets the options for the panning (without the zoom change) if it occurs.
zoom	zoom options	-	Sets the options for the zoom change if it occurs.
animate	Boolean	-	An equivalent of passing animate to both zoom and pan options (see below).

Pan options

选项	类型	默认值	描述
animate	Boolean	-	If true, panning will always be animated if possible. If false, it will not animate panning, either resetting the map view if panning more than a screen away, or just setting a new offset for the map pane (except for `panBy` which always does the latter).
duration	Number	0.25	Duration of animated panning.
easeLinearity	Number	0.25	The curvature factor of panning animation easing (third parameter of the Cubic Bezier curve). 1.0 means linear animation, the less the more bowed the curve.
noMoveStart	Boolean	false	If true, panning won't fire movestart event on start (used internally for panning inertia).

Zoom options

选项	类型	默认值	描述
animate	Boolean	-	If not specified, zoom animation will happen if the zoom origin is inside the current view. If true, the map will attempt animating zoom disregarding where zoom origin is. Setting false will make it always reset the view completely without animation.

fitBounds options

The same as zoom/pan options and additionally:

选项	类型	默认值	描述
paddingTopLeft	Point	[0, 0]	Sets the amount of padding in the top left corner of a map container that shouldn't be accounted for when setting the view to fit bounds. Useful if you have some control overlays on the map like a sidebar and you don't want them to obscure objects you're zooming to.
paddingBottomRight	Point	[0, 0]	The same for bottom right corner of the map.
padding	Point	[0, 0]	Equivalent of setting both top left and bottom right padding to the same value.

Properties

M地图属性包括互动操作，允许你在运行环境中互动地控制地图行为，比如通过拖拽和点击缩放级别显示和不显示某要素. Example:

```
map.doubleClickZoom.disable();
```

You can also access default map controls like attribution control through map properties:

```
map.attributionControl.addAttribution("Earthquake data &copy; GeoNames");
```

Property	类型	描述
dragging	IHandler	地图拖拽处理程序，可以通过鼠标和触摸的形式.
touchZoom	IHandler	触摸地图缩放处理程序.
doubleClickZoom	IHandler	双击缩放处理程序.
scrollWheelZoom	IHandler	滚动缩放处理程序.
boxZoom	IHandler	矩形框（利用鼠标拖动）缩放处理程序.
keyboard	IHandler	键盘导向处理程序.
tap	IHandler	Mobile touch hacks (quick tap and touch hold) handler.
zoomControl	Control.Zoom	缩放控制.
attributionControl	Control.Attribution	属性控制.

地图窗口

望文思义，这是一个包括可以用来放置自定义图层的不同的地图窗口的对象。最大的区别是图层的叠置.

Property	类型	描述
mapPane	HTMLElement	包含其他地图窗口的窗口.
tilePane	HTMLElement	切片图层的窗口.
objectsPane	HTMLElement	包含除切片窗口以外所有窗口的窗口.
shadowPane	HTMLElement	用来隐藏图层的窗口（如标注的隐藏）.
overlayPane	HTMLElement	这线段和多边形一类图层的窗口.
markerPane	HTMLElement	标注图标的窗口.
popupPane	HTMLElement	弹出的窗口.

L.Marker

用来在地图中放置注记

```
L.marker([50.5, 30.5]).addTo(map);
```

构造函数

构造函数	使用	描述
L.Marker(<LatLng> latlng, <Marker options> options?)	new L.Marker(…) L.marker(…)	Instantiates a Marker object given a geographical point and optionally an options object.

Options

选项	类型	默认值	描述
icon（图标）	L.Icon	*	图标类用来表达注记。参见Icon documentation以了解自定义注记图标的详细信息。默认设置为new L.Icon.Default()。
clickable（可点击）	Boolean	true	如果是false，注记则不产生鼠标事件并表现为底层地图的一部分。
draggable（可拖动）	Boolean	false	决定注记是否可被鼠标或触摸拖动。
keyboard	Boolean	true	Whether the marker can be tabbed to with a keyboard and clicked by pressing enter.

title（标题）	String	''	注记旁边显示浏览器提示的文本信息。
zIndexOffset	Number	0	默认情况下，注记图片的叠置顺序由纬度自动设置。如果你想将某一注记放置于其他之上可用这个选项，设置一个较大的值即可，比如1000（或是相反地设置一个较大的负值）。
opacity（不透明度）	Number	1.0	决定注记的不透明度。
riseOnHover（凸显）	Boolean	false	如果此值为true，则当把鼠标放置于注记之上时，注记会显示与其他注记之上。
riseOffset（凸显补偿）	Number	250	riseOnHover要素凸显时高度的补偿值。

Events（事件）

You can subscribe to the following events using these methods.

Event	Data	描述
click（点击）	MouseEvent	当鼠标点击注记时触发。
dblclick（双击）	MouseEvent	当鼠标双击注记时触发。
mousedown（鼠标按下）	MouseEvent	当鼠标按下鼠标键时触发。
mouseover（鼠标置于其上）	MouseEvent	当鼠标在注记上时触发。
mouseout（鼠标移出）	MouseEvent	当鼠标离开注记时触发。
contextmenu（文本菜单）	MouseEvent	当鼠标右击注记时触发。
dragstrat（拖动开始）	Event	当用户拖动注记时触发。
drag（拖动）	Event	当用户拖动注记时不断触发。
dragend（拖动结束）	Event	当用户停止拖动注记时触发。
move（移动）	Event	当注记通过定义经纬度而移动时触发。新的坐标包含事件参数。
remove（删除）	Event	当注记在地图上被删除时触发。
popupopen	PopupEvent	Fired when a popup bound to the marker is open.
popupclose	PopupEvent	Fired when a popup bound to the marker is closed.

Methods（方法）

方法	返回值	描述

addTo(<Map> map)	this	在地图上添加注记。
getLatLng()	LatLng	返回当前注记的地理位置。
setLatLng(<LatLng> latlng)	this	将注记位置更改到给定点。
setIcon(<Icon> icon)	this	更改注记的图标。
setZIndexOffset(<Number> offset)	this	更改注记 zIndex offset
setOpacity(<Number> opacity)	this	更改注记的透明度。
update()	this	更新注记的位置，在直接更改经纬度对象的坐标时比较有用。
bindPopup(<String> html | <HTMLElement> el | <Popup> popup, <Popup options> options?)	this	当点击一个注记时绑定一个特定的HTML内容的弹出窗口。你也可以用Marker中的openPopup方法打开绑定的弹出窗口。
unbindPopup()	this	将先前用bindPopup方法绑定的注记取消。
openPopup()	this	打开先前用bindPopup方法绑定的弹出框。
closePopup()	this	关闭已打开的注记的弹出框。
togglePopup()	this	Toggles the popup previously bound by the bindPopup method.
setPopupContent(<String> html | <HTMLElement> el, <Popup options> options?)	this	设置弹窗的文本内容Binds a popup with a particular HTML content to a click on this marker. You can also open the bound popup with the Marker openPopup method.
toGeoJSON()	Object	返回值 a GeoJSON representation of the marker (GeoJSON Point Feature).

Interaction handlers（互操作处理程序）

Interaction handlers are properties of a marker instance that allow you to control interaction behavior in runtime, enabling or disabling certain features such as dragging (see IHandler methods). Example:

```
marker.dragging.disable();
```

Property	类型	描述
dragging（拖动）	IHandler	注记拖动处理程序（包括鼠标和触摸）。

L.Popup

Used to open popups in certain places of the map. Use Map#openPopup to open popups while making sure that only one popup is open at one time (recommended for usability), or use Map#addLayer to open as many as you want.

使用 example

If you want to just bind a popup to marker click and then open it, it's really easy:

```
marker.bindPopup(popupContent).openPopup();
```

Path overlays like polylines also have a `bindPopup` method. Here's a more complicated way to open a popup on a map:

```
var popup = L.popup()
        .setLatLng(latlng)
        .setContent('<p>Hello world!<br />This is a nice popup.</p>')
        .openOn(map);
```

构造器

构造器	使用	描述
`L.Popup( <Popup options> options?, <ILayer> source? )`	`new L.Popup(…)` `L.popup(…)`	通过给定一些选项构造一个弹出框对象，对象用来描述出现形式和位置还有一个可选对象来根据指向的资源对象标注弹出框。

Options

选项	类型	默认值	描述
`maxWidth`	`Number`	`300`	弹出框的最大宽度。
`minWidth`	`Number`	`50`	弹出框的最小宽度。
`maxHeight`	`Number`	`null`	设置后，如果内容超过弹出窗口的给定高度则产生一个可以滚动的容器。
`autoPan`（自动平移）	`Boolean`	`true`	如果你不想地图自动平移来适应打开的弹出框，就设置其为 false。
`keepInView`	`Boolean`	`false`	Set it to `true` if you want to prevent users from panning the popup off of the screen while it is open.
`closeButton`（关闭按钮）	`Boolean`	`true`	控制弹出窗口中出现的关闭按钮。
`offset`（补偿值）	Point	`Point(0, 6)`	弹出窗口位置的补偿值。在同一图层中打开弹出窗口时对于控制锚点比较有用。
`autoPanPadding`（自动平移填补）	Point	`Point(5, 5)`	在地图视图自动平移产生后弹出窗口和地图视图之间的边缘。

zoomAnimation	Boolean	true	决定是否在所在级别上弹出窗口。如果你在弹出窗口中有flash内容的最好将其设置为不可用。
closeOnClick	Boolean	null	Set it to `false` if you want to override the default behavior of the popup closing when user clicks the map (set globally by the `Map closePopupOnClick` option).

Methods

方法	返回值	描述
addTo(<Map> map)	this	将弹出窗口添加到地图上。
openOn(<Map> map)	this	将弹出窗口添加到地图上并将之前的一个关闭。与map.oenPopup(popup)方法相同。
setLatLng(<LatLng> latlng)	this	设置弹出窗口打开的地理上的点位。
setContent(<String|HTMLElement> htmlContent)	this	设置弹出窗口的HTML内容。

L.TileLayer

用来在地图上载入和显示切片图层，用ILayer接口实现。

使用 example

```
L.tileLayer('http://{s}.tile.cloudmade.com/{key}/{styleId}/256/{z}/{x}/{y}.png', {
        key: 'API-key',
        styleId: 997
}).addTo(map);
```

构造器

构造器	使用	描述
L.TileLayer(<String> urlTemplate, <TileLayer options> options?)	new L.TileLayer(…) L.tileLayer(…)	L.TileLayer()：通过给定URL模板和具有选项的对象来实例化一个切片图层。

URL template

A string of the following form:

```
'http://{s}.somedomain.com/blabla/{z}/{x}/{y}.png'
```

`{s}` means one of the available subdomains (used sequentially to help with browser parallel requests per domain limitation; subdomain values are specified in options; `a`, `b` or `c` by default, can be omitted), `{z}` — zoom level, `{x}`

and {y} — tile coordinates.

You can use custom keys in the template, which will be evaluated from TileLayer options, like this:

```
L.tileLayer('http://{s}.somedomain.com/{foo}/{z}/{x}/{y}.png', {foo: 'bar'});
```

Options

选项	类型	默认值	描述
minZoom	Number	0	最小级别数
maxZoom	Number	18	最大级别数
tileSize	Number	256	切片尺寸（宽度和高度的像素值，假设切片是正方形的）
subdomains	String or String[]	'abc'	服务的子域。可以传递一个字符串（其中每一个字母都是一个子域名称）或是一个字符串数组。
errorTileUrl	String	''	图片的URL给出加载错误的位置。
attribution	String	''	e.g. "© CloudMade" — 用来进行属性控制的字符串，描述了图层数据。
tms	Boolean	false	如果此值为true，反转切片Y轴的编号（对于TMS服务需将此项打开）
continuousWorld	Boolean	false	如果设置为true，切片的坐标系不会被世界范围的宽度（-180度到180度）所覆盖，也不会被在世界范围的高度（-90度到90度）之内。你可以将此用在不反应真是世界的地图上（比如游戏、室内或照片的地图）。
noWrap	Boolean	false	如果设置此项为true，则切片不会用重复填充来表示世界范围（经度-180到180之间）之外的地方。
zoomOffset	Number	0	用此值来补偿URL中地图的缩放级别。
zoomReverse	Boolean	false	如果此项为true，URL中的缩放级别会被反转（用最大到最小缩放级别来替代缩放级别）。
opacity	Number	1.0	切片图层的透明度。
zIndex	Number	null	切片图层明确的叠置顺序，默认此项不会被设置。
unloadInvisibleTiles	Boolean	depends	如果此项为true，在平移后所有看不到的切片都会被移除（用以更好地显示），在移动设备的webkit中默认是true，其他的默认为false。
updateWhenIdle	Boolean	depends	如果此项为false，在平移过程中新的切片将会载入，其他的在其后载入（用以更好地显示），在移动设备webKit中默认是true，其他默认false。
detectRetina	Boolean	false	如果此项为true，并且用户是视网膜显示模式，会请求规定大小一般的四个切片和一个地区内一个更大的缩放级别来利用高分辨率。

reuseTiles	Boolean	false	如果此项为true，在平移后不可见的切片被放入一个队列中，在新的切片开始可见时他们会被取回（而不是动态地创建一个新的）。这理论上可以降低内存使用率并可以去除在需要新的切片时预留内存。

Events

You can subscribe to the following events using these methods.

Event	Data	描述
loading	Event	当切片图层开始加载切片时触发。
load	Event	当切片图层加载完可见切片后触发。
tileload	TileEvent	在加载切片时触发。
tileunload	TileEvent	在切片被移除时触发（比如打开了unloadInvisibleTiles）。

Methods

方法	返回值	描述
addTo(<Map> map)	this	将图层加到地图上。
bringToFront()	this	将此切片图层放到所有切片图层之上。
bringToBack()	this	将此切片图层放到所有切片图层之下。
setOpacity(<Number> opacity)	this	改变切片图层的透明度。
setZIndex(<Number> zIndex)	this	设置切片图层的叠放顺序。
redraw()	this	清除所有的切片并重新向服务端申请他们。
setUrl(<String> urlTemplate)	this	更新图层的URL模板并重绘他们。
getContainer()	HTMLElement	返回值 the HTML element that contains the tiles for this layer.

L.TileLayer.WMS

用来显示地图上切片图层的WMS服务，继承自TileLayer.

使用 example

```
var nexrad = L.tileLayer.wms("http://mesonet.agron.iastate.edu/cgi-bin/wms/nexrad/n0r.cgi", {
```

```
        layers: 'nexrad-n0r-900913',
        format: 'image/png',
        transparent: true,
        attribution: "Weather data © 2012 IEM Nexrad"
});
```

构造器

构造器	使用	描述
`L.TileLayer.WMS( <String> baseUrl, <TileLayer.WMS options> options )`	`new L.TileLayer.WMS(…)` `L.tileLayer.wms(…)`	通过给定一个基本的WMS服务的URL和WMS参数或选项对象来实例化一个WMS切片图层对象。

Options

Includes all TileLayer options and additionally:

选项	类型	默认值	描述
layers	String	''	**(required)**WMS图层以逗号分隔符隔开的列表。
styles	String	''	WMS样式以逗号分隔符隔开的列表。
format	String	'image/jpeg'	WMS图像格式（用“image/png”来显示透明图层）
transparent	Boolean	false	如果该项为true，WMS服务返回透明图片。
version	String	'1.1.1'	WMS服务的版本
crs	CRS	null	Coordinate Reference System to use for the WMS requests, defaults to map CRS. Don't change this if you're not sure what it means.

Methods

方法	返回值	描述
`setParams( <WMS parameters> params, <Boolean> noRedraw? )`	this	融合新的参数和在当前屏幕中重申请的切片（除非noRedraw设置为true）。

L.TileLayer.Canvas

用来创建浏览器端绘制的切片图层的底层画布。

使用 example

```
var canvasTiles = L.tileLayer.canvas();

canvasTiles.drawTile = function(canvas, tilePoint, zoom) {
        var ctx = canvas.getContext('2d');
        // draw something on the tile canvas
}
```

构造器

构造器	使用	描述
L.TileLayer.Canvas(<TileLayer options> options?)	new L.TileLayer.Canvas(…) L.tileLayer.canvas(…)	通过一个具有选项的对象来实例化一个切片图层画布对象。

Options

选项	类型	默认值	描述
async	Boolean	false	在实例化时可以异步地绘制切片。在全部绘制完后，tileDrawn方法需要在每个切片上使用。

Methods

方法	返回值	描述
drawTile(<HTMLCanvasElement> canvas, <Point> tilePoint, <Number> zoom)	this	在创建实例来绘制切片后你需要定义此方法；canvas是你可以绘制的实际上的切片画布，tilePoint反应了切片的数目，zoom是当前的缩放级别。
tileDrawn(<HTMLCanvasElement> canvas)	-	如果async选项被定义，在全部绘制完后，这个函数需要在每个切片上使用。canvas与画布对象相同，传递参数给drawTile。

L.ImageOverlay

用来在地图上规定范围内载入和显示单幅图像，继承自 ILayer

使用 example

```
var imageUrl = 'http://www.lib.utexas.edu/maps/historical/newark_nj_1922.jpg',
        imageBounds = [[40.712216, -74.22655], [40.773941, -74.12544]];

L.imageOverlay(imageUrl, imageBounds).addTo(map);
```

构造器

构造器	使用	描述
`L.ImageOverlay( <String> imageUrl, <LatLngBounds> bounds, <ImageOverlay options> options? )`	`new L.ImageOverlay(…)` `L.imageOverlay(…)`	通过给定图像的URL和相关的地理范围来实例化一个图像叠加层对象。

Options

选项	类型	默认值	描述
opacity	Number	1.0	图像叠加层的透明度。

Methods

方法	返回值	描述
`addTo( <Map> map )`	this	将图像叠加层添加到地图上。
`setOpacity( <Number> opacity )`	this	设置叠加层的透明度。
`bringToFront()`	this	将叠加层置于所有层的顶层。
`bringToBack()`	this	将叠加层置于所有层的底层。

L.Path

是包含选项和与适量叠加层共享常量的抽象类。不可以接使用。

Options

选项	类型	默认值	描述
stroke	Boolean	true	路径是否描边。设置为false时，多边形和圆的边界将不可见。
color	String	'#03f'	描边颜色。
weight	Number	5	描边的像素级别的宽度.
opacity	Number	0.5	描边透明度。
fill	Boolean	depends	路径是否填充颜色。设置为false时，多边形和圆的填充内容不可见。
fillColor	String	same as color	填充颜色。
fillOpacity	Number	0.2	填充透明度。
dashArray	String	null	定义描边线型的字符串。这在画布上不起作用。（比如android 2）

clickable	Boolean	true	如果此项为false，则矢量不产生鼠标事件并表现为底图的一部分。
pointerEvents	String	null	Sets the pointer-events attribute on the path if SVG backend is used.

Events

You can subscribe to the following events using these methods.

Event	Data	描述
click	MouseEvent	用户点击或点触对象时触发。
dblclick	MouseEvent	用户双击或连续两次点触对象时触发。
mousedown	MouseEvent	当用户在对象上按下鼠标时触发。
mouseover	MouseEvent	当鼠标置于对象上方时触发。
mouseout	MouseEvent	当鼠标离开对象时触发。
contextmenu	MouseEvent	当用户在对象上点击鼠标右键时触发，当此事件被监听时，会阻止弹出浏览器本身的右键菜单。
add	Event	当路径被添加在地图上时触发。
remove	Event	当路径在地图上移除时触发。
popupopen	PopupEvent	Fired when a popup bound to the path is open.
popupclose	PopupEvent	Fired when a popup bound to the path is closed.

Methods

方法	返回值	描述
addTo(<Map> map)	this	将图层添加到地图上。
bindPopup(<String> html | <HTMLElement> el | <Popup> popup, <Popup options> options?)	this	将具有特定HTML内容的弹出框与点击路径绑定起来。
bindPopup(<Popup> popup, <Popup options> options?)	this	Binds a given popup object to the path.
unbindPopup()	this	将之前的弹出框绑定解除。
openPopup(<LatLng> latlng?)	this	打开之前通过bindPopup方法与路径上指定点或未指定情况下某一点绑定的弹出框。
closePopup()	this	如果与路径绑定的弹出框是打开状

		态的，则将其关闭。
setStyle(<Path options> object)	this	更改给予对象选项对象的路径的表现形式。
getBounds()	LatLngBounds	返回路径的经纬度绑定信息。
bringToFront()	this	将此层移至所以路径层的最上层。
bringToBack()	this	将此层移至所以路径层的最底层。
redraw()	this	重绘图层。在更改了路径的坐标时比较有用。

Static properties（静态属性）

Constant	类型	Value	描述
SVG	Boolean	depends	如果用SVG来表达矢量，则此值为true（在当前大多数浏览器中是true）。
VML	Boolean	depends	如果VML用来表达矢量，则此值为true（在IE 6-8中适用）。
CANVAS	Boolean	depends	如果canvas用来表达矢量，则此值为true（在android 2中适用）。你也可以在页面中载入leaflet之前通过设置全局变量L_PREFER_CANVAS为true来强制使用此项——有时在表达上千上万相同的注记时会显著地提高性能，但目前由于漏洞导致移除图层非常慢。
CLIP_PADDING	Number	0.5 for SVG 0.02 for VML	决定地图视图周围裁剪区域延展的大小（与大小相关，比如0.5在每个方向上是屏幕的一半）。较小的值意味着在拖动地图时你会看到被裁剪路径的末端，较大值会降低绘制性能。

L.Polyline

绘制叠加在地图上线段的类。继承自Path。用Map#addLayer来添加到地图上。 Extends Path. Use Map#addLayer to add it to the map.

使用 example

```
// create a red polyline from an arrays of LatLng points
var polyline = L.polyline(latlngs, {color: 'red'}).addTo(map);

// zoom the map to the polyline
map.fitBounds(polyline.getBounds());
```

构造器

构造器	使用	描述

L.Polyline(<LatLng[]> latlngs, <Polyline options> options?)	new L.Polyline(…) L.polyline(…)	通过给定的地理点组成的数组和任意的选项对象实例化一个线段。

Options

You can use Path options and additionally the following options:

选项	类型	默认值	描述
smoothFactor	Number	1.0	决定每一个缩放级别上线段简化程度。如果大的话意味着更好的表现和看起来更光滑，小的话意味更准确地表示。
noClip	Boolean	false	不允许线段裁剪。

Methods

You can use Path methods and additionally the following methods:

方法	返回值	描述
addLatLng(<LatLng> latlng)	this	向线段添加一个点。
setLatLngs(<LatLng[]> latlngs)	this	用所给的地理点的数组替代线段上的点。
getLatLngs()	LatLng[]	返回路径上的点组成的数组。
spliceLatLngs(<Number> index, <Number> pointsToRemove, <LatLng> latlng?, …)	LatLng[]	允许添加、移除和更改线段上的点。同Array#splice的语法一致。返回移除点组成的数组。
getBounds()	LatLngBounds	返回线段的经纬度边界。
toGeoJSON()	Object	返回值 a GeoJSON representation of the polyline (GeoJSON LineString Feature).

L.MultiPolyline

是FeatureGroup的扩展，用来创建多线（在同一图层中由多个共享样式和弹出框的线段组成）。

构造器

构造器	使用	描述
L.MultiPolyline(<LatLng[][]> latlngs, <Polyline options> options?)	new L.MultiPolyline(…) L.multiPolyline(…)	通过给定的地理点的二维数组（其中每个一维数组表示一个线段）和选项对象来实例化一个多线对象。

Methods

MultiPolylines accept all Polyline methods but have different behavior around their coordinate contents since they can contain multiple line features:

方法	返回值	描述
`setLatLngs( <LatLng[][]> latlngs )`	`this`	Replace all lines and their paths with the given array of arrays of geographical points.
`getLatLngs()`	`<LatLng[][]> latlngs`	返回值 an array of arrays of geographical points in each line.
`toGeoJSON()`	`Object`	返回值 a GeoJSON representation of the multipolyline (GeoJSON MultiLineString Feature).

L.Polygon

在地图上绘制多边形的类。是Polyline的扩展。用Map#addLayer添加到地图上。

创建多边形时经过的点没有传统意义上的起点和终点——最好将这种点指出来。

构造器

构造器	使用	描述
`L.Polygon( <LatLng[]> latlngs, <Polyline options> options? )`	`new L.Polygon(…)` `L.polygon(…)`	通过给定地理点组成的数组和选项对象来实例化一个多边形（同线段构造方法相同）。你也可以通过传递经纬度的二维数组来创建一个带有洞的多边形，第一个经纬度数组表示外环，剩下的表示里面的洞。

Methods

Polygon has the same options and methods as Polyline, with the following differences:

方法	返回值	描述
`toGeoJSON()`	`Object`	返回值 a GeoJSON representation of the polygon (GeoJSON Polygon Feature).

L.MultiPolygon

是FeatureGroup的扩展，用来创建多多边形（在同一图层上由共享样式和弹出框的多个多边形组成）。

构造器

构造器	使用	描述
`L.MultiPolygon( <LatLng[][]> latlngs, <Polyline options> options? )`	`new L.MultiPolygon(…)` `L.multiPolygon(…)`	通过给定的经纬度的二维数组（每个一维数组表示一个多边形）和选项对象实例化多多边形（同多线相同）

Methods

MultiPolygons accept all Polyline methods but have different behavior around their coordinate contents since they can contain multiple polygon features:

方法	返回值	描述
`setLatLngs( <LatLng[][]> latlngs )`	`this`	Replace all polygons and their paths with the given array of arrays of geographical points.
`getLatLngs()`	`<LatLng[][]> latlngs`	返回值 an array of arrays of geographical points in each polygon.
`toGeoJSON()`	`Object`	返回值 a GeoJSON representation of the multipolygon (GeoJSON MultiPolygon Feature).

L.Rectangle

在地图上绘制矩形的类。是 Polygon的扩展。用Map#addLayer 添加到地图上。

使用 example

```
// define rectangle geographical bounds
var bounds = [[54.559322, -5.767822], [56.1210604, -3.021240]];

// create an orange rectangle
L.rectangle(bounds, {color: "#ff7800", weight: 1}).addTo(map);

// zoom the map to the rectangle bounds
map.fitBounds(bounds);
```

构造器

构造器	使用	描述
`L.Rectangle( <LatLngBounds> bounds, <Path options> options? )`	`new L.Rectangle(…)` `L.rectangle(…)`	通过给定的地理边界和选项对象来实例化一个矩形对象。

Methods

You can use Path methods and additionally the following methods:

方法	返回值	描述
setBounds(<LatLngBounds> bounds)	this	根据传递的边界重绘矩形。

L.Circle

在地图上绘制圆形叠加物的类。是Path的延伸。用Map#addLayer来添加到地图上。

```
L.circle([50.5, 30.5], 200).addTo(map);
```

构造器

构造器	使用	描述
L.Circle(<LatLng> latlng, <Number> radius, <Path options> options?)	new L.Circle(…) L.circle(…)	通过给定的地理点和以米为单位的半径和选项对象来实例化一个圆对象。

Methods

方法	返回值	描述
getLatLng()	LatLng	返回圆当前的地理位置。
getRadius()	Number	返回圆的半径，以米为单位。
setLatLng(<LatLng> latlng)	this	将圆放置到一个新的位置。
setRadius(<Number> radius)	this	设置圆的半径，以米为单位。
toGeoJSON()	Object	返回值 a GeoJSON representation of the circle (GeoJSON Point Feature).

L.CircleMarker

是一个特定半径的圆，半径单位是像素。是Circle的延伸。通过Map#addLayer添加到地图上。

构造器

构造器	使用	描述
L.CircleMarker(<LatLng> latlng, <Path options> options?)	new L.CircleMarker(…) L.circleMarker(…)	通过给定的地理点和选项对象来实例化一个圆注记。默认的半径是10像素，并且可以通过在路径选项中传递一个半径参数来修改半径。

Methods

方法	返回值	描述
setLatLng(<LatLng> latlng)	this	将圆注记放置于一个新的位置。
setRadius(<Number> radius)	this	设置圆注记的半径，以像素为单位。
toGeoJSON()	Object	返回值 a GeoJSON representation of the circle marker (GeoJSON Point Feature).

L.LayerGroup

用来将几个图层组成一个组并作为一个图层来处理。如果你将其添加到地图上，组中任何图层的添加或移除都将使其同样在地图添加或删除。继承自ILayer接口。

```
L.layerGroup([marker1, marker2])
        .addLayer(polyline)
        .addTo(map);
```

构造器

构造器	使用	描述
L.LayerGroup(<ILayer[]> layers?)	new L.LayerGroup(…) L.layerGroup(…)	L.LayerGroup()：创建一个组，视情况指定一组初始的图层。

Methods

方法	返回值	描述
addTo(<Map> map)	this	将图组添加到地图上。
addLayer(<ILayer> layer)	this	将给定的图层添加到组中。
removeLayer(<ILayer> layer)	this	将给定的图层从组中移除：
removeLayer(<String> id)	this	Removes a given layer of the given id from the group.
hasLayer(<ILayer> layer)	Boolean	返回值 true if the given layer is currently added to the group.
getLayer(<String> id)	Boolean	返回值 the layer with the given id.
getLayers()	Array	返回值 an array of all the layers added to the group.
clearLayers()	this	将组中的图层清空。

eachLayer(<Function> fn, <Object> context?)	this	遍历组中的图层，需选择一个符合情况的迭代函数。
		group.eachLayer(function (layer) {
		layer.bindPopup('Hello');

		});
toGeoJSON()	Object	返回值 a GeoJSON representation of the layer group (GeoJSON FeatureCollection).

L.FeatureGroup

是Extended LayerGroup 的扩展，但多了鼠标事件和共享的弹出框方法。继承自ILayer 接口

```
L.featureGroup([marker1, marker2, polyline])
        .bindPopup('Hello world!')
        .on('click', function() { alert('Clicked on a group!'); })
        .addTo(map);
```

构造器

构造器	使用	描述
L.FeatureGroup(<ILayer[]> layers?)	new L.FeatureGroup(…) L.featureGroup(…)	创建一个图组，视情况指定一组初始图层。

Methods

具有 LayerGroup 所有的方法，还有下面多出的方法：

方法	返回值	描述
bindPopup(<String> htmlContent, <Popup options> options?)	this	在组中任意具有bindPopup方法的图层上绑定一个具有具体HTML内容的弹出框。
getBounds()	LatLngBounds	返回要素组的经纬度边界（通过他子图层的边界和坐标获得）。
setStyle(<Path options> style)	this	设置组中具有setStyle方法的图层的路径选项。
bringToFront()	this	将图组置于顶层。
bringToBack()	this	将图组置于底层。

Events

You can subscribe to the following events using these methods.

Event	**Data**	描述
click	MouseEvent	用户点击或触摸组是触发。
		用户双击或连续两次触摸组时触发。

dblclick	MouseEvent	
mouseover	MouseEvent	当鼠标置于组上方时触发。
mouseout	MouseEvent	当鼠标离开组时触发。
mousemove	MouseEvent	当鼠标经过组时触发。
contextmenu	MouseEvent	当用户右击图层时触发。
layeradd	LayerEvent	当图层被加入到组时触发。
layerremove	LayerEvent	当图层从组中移除时触发。

L.GeoJSON

展示一个GeoJSON 的图层.允许你在地图上解析并显示GeoJSON数据。是FeatureGroup的延伸。 由此创建的每个要素层获取要素与之关联的GeoJSON数据属性（因此你随后可以传递它的属性）。

```
L.geoJson(data, {
        style: function (feature) {
                return {color: feature.properties.color};
        },
        onEachFeature: function (feature, layer) {
                layer.bindPopup(feature.properties.description);
        }
}).addTo(map);
```

Each feature layer created by it gets a `feature` property that links to the GeoJSON feature data the layer was created from (so that you can access its properties later).

构造器

构造器	使用	描述
`L.GeoJSON( <Object> geojson?, <GeoJSON options> options? )`	`new L.GeoJSON(…)` `L.geoJson(…)`	创建一个GeoJSON图层。可以任意地接受GeoJSON格式的对象和选项对象并显示在地图上（随后可以选择用addData方法添加）。

Options

选项	描述
`pointToLayer( <GeoJSON> featureData, <LatLng> latlng )`	在创建GeoJSON点图层时所用到的函数（如果不特意说明，会创建简单的注记）。
`style( <GeoJSON> featureData )`	在获取用来创建GeoJSON要素的矢量图层的样式选项时可以用到。

onEachFeature(<GeoJSON> featureData, <ILayer> layer)	在每个创建的图层上都会调用此函数。对于向要素添加事件和弹出框比较有用。
filter(<GeoJSON> featureData, <ILayer> layer)	用来决定是否显示某要素的函数。
coordsToLatLng(<Array> coords)	Function that will be used for converting GeoJSON coordinates to LatLng points (if not specified, coords will be assumed to be WGS84 — standard [longitude, latitude] values in degrees).

Methods

方法	返回值	描述
addData(<GeoJSON> data)	Boolean	在图层中添加GeoJSON对象。
setStyle(<Function> style)	this	通过给定的样式函数改变GeoJSON矢量图层的样式。
resetStyle(<Path> layer)	this	将矢量图层样式重置为初始GeoJSON样式，对于hover事件之后的重置比较有用。

Static methods

方法	返回值	描述
geometryToLayer(<GeoJSON> featureData, <Function> pointToLayer?)	ILayer	通过给定的GeoJSON要素创建图层。
coordsToLatlng(<Array> coords, <Boolean> reverse?)	LatLng	通过在GeoJSON中表示点的两个数字组成（分别表示纬度和经度）的数组来创建经纬度对象。如果reverse设置为true，那么这两个数字被颠倒，表经度和纬度。
coordsToLatlngs(<Array> coords, <Number> levelsDeep?, <Boolean> reverse?)	Array	通过GeoJSON坐标坐标的数组创建多维数组。leversDeep指定具体的嵌套级别（0表示点的数组，1表示点数组的数组等等，0为默认值）。如果reverse设置为true，这些数组变为经度和纬度。

L.LatLng

表示通过某一经度和纬度确定的地理上的点。

```
var latlng = new L.LatLng(50.5, 30.5);
```

所有leaflet接受的经纬度对象也接受他们的单一数组的形式（除非在其他方面表明不可以）：

```
map.panTo([50, 30]);
map.panTo({lon: 30, lat: 50});
map.panTo({lat: 50, lng: 30});
map.panTo(new L.LatLng(50, 30));
```

构造器

构造器	使用	描述

`L.LatLng( <Number> latitude, <Number> longitude )`	`new L.LatLng(…)`
	`L.latLng(…)`
	`L.latLng([…])`
		通过给定的纬度和经度创建表示地理点的对象。

Properties

Property	类型	描述
`lat`	`Number`	以度数表示的纬度。
`lng`	`Number`	以度数表示的经度。

Methods

方法	返回值	描述
`distanceTo( <LatLng> otherLatlng )`	`Number`	返回到通过半正矢公式计算的经纬度的距离（用米表示）。
`equals( <LatLng> otherLatlng )`	`Boolean`	如果给定的经纬度在相同的位置（具有较小的容差）则返回 true。
`toString()`	`String`	返回点的描述信息（用来调试用）。
`wrap( <Number> left, <Number> right )`	LatLng	返回在经度上left和right边界覆盖范围内（默认为0180到180）的心的经纬度对象。

Constants（常量）

Constant	类型	**Value**	描述
DEG_TO_RAD	`Number`	`Math.PI / 180`	度数转换为弧度的乘子。
RAD_TO_DEG	`Number`	`180 / Math.PI`	弧度转换为度数的乘子。
MAX_MARGIN	`Number`	`1.0E-9`	判断相等的容差。

L.LatLngBounds

表示地图上一个矩形的区域。

```
var southWest = new L.LatLng(40.712, -74.227),
        northEast = new L.LatLng(40.774, -74.125),
        bounds = new L.LatLngBounds(southWest, northEast);
```

所有接受LatLngBounds对象的leaflet方法也接受他们简单数组的形式（除非另行说明）：

```
map.fitBounds([
        [40.712, -74.227],
        [40.774, -74.125]
]);
```

构造器

构造器	使用	描述
L.LatLngBounds(<LatLng> southWest, <LatLng> northEast)	new L.LatLngBounds(…) L.latLngBounds(…) L.latLngBounds([…])	通过定义矩形西南角点和东北角点来创建经纬度的矩形框。
L.LatLngBounds(<LatLng[]> latlngs)	new L.LatLngBounds(…) L.latLngBounds(…)	通过定义内在点来创建经纬度的矩形框。当用fitBounds把地图放到适合某些位置的缩放级别时是比较有用的。

Methods

方法	返回值	描述
extend(<LatLng|LatLngBounds> latlng)	this	将边界延伸到包含给定点和边界的范围。
getSouthWest()	LatLng	返回边界的西南角点。
getNorthEast()	LatLng	返回边界的东北角点。
getNorthWest()	LatLng	返回边界的西北角点。
getSouthEast()	LatLng	返回边界的东南角点。
getWest()	Number	返回边界的西点。
getSouth()	Number	返回边界的南角点。
getEast()	Number	返回边界的东角点。
getNorth()	Number	返回边界的北角点。
getCenter()	LatLng	返回边界的中心点。
contains(<LatLngBounds> otherBounds)	Boolean	如果矩形框包含给定的边界则返回true。
contains(<LatLng> latlng)	Boolean	如果矩形框包含给定的点则返回

		true。
intersects(<LatLngBounds> otherBounds)	Boolean	如果矩形框与给定的边界相交则返回true。
equals(<LatLngBounds> otherBounds)	Boolean	如果矩形框与给定的范围相等（在一定容差范围内）则返回true。
toBBoxString()	String	返回“西南经度，西南纬度，东北经度，东北纬度”形式的外接矩形的坐标。在向网络服务器提交请求返回地理数据时比较有用。
pad(<Number> bufferRatio)	LatLngBounds	返回当前范围扩大一定百分比后的边界。
isValid()	Boolean	如果边界可被初始化则返回true。

L.Point

显示以像素为单位的点的x，y坐标。

```
var point = new L.Point(200, 300);
```

所有接受点对象的leaflet方法和选项都也接受他们简单数组的形式:

```
map.panBy([200, 300]);
map.panBy(new L.Point(200, 300));
```

构造器

构造器	使用	描述
L.Point(<Number> x, <Number> y, <Boolean> round?)	new L.Point(…) L.point(…) L.point([…])	用给定点的x和y坐标来创建点对象。如果round设置为true，则将x和y的值转换为圆中。？

Properties

Property	类型	描述
x	Number	x坐标。
y	Number	y坐标.

Methods

方法	返回值	描述
add(<Point> otherPoint)	Point	返回当前点和给定点的和。

subtract(<Point> otherPoint)	Point	返回当前点和给定点的差。
multiplyBy(<Number> number)	Point	返回当前点和给定值的积。
divideBy(<Number> number, <Boolean> round?)	Point	返回当前点和给定值的商。如果round设置为true，则返回一个圆的结果。
distanceTo(<Point> otherPoint)	Number	返回当前点与给定的的距离。
clone()	Point	返回当前的副本。
round()	Point	返回当前的在圆上的坐标的副本。
equals(<Point> otherPoint)	Boolean	返如果点坐标相同则返回true。
toString()	String	在调试时显示点的字符串的形式。

L.Bounds

用像素坐标表示的矩形的区域。

```
var p1 = new L.Point(10, 10),
        p2 = new L.Point(40, 60),
        bounds = new L.Bounds(p1, p2);
```

所有接受边界对象的leaflet方法和选项都也接受他们简单数组的形式:

```
otherBounds.intersects([[10, 10], [40, 60]]);
```

构造器

构造器	使用	描述
L.Bounds(<Point> topLeft, <Point> bottomRight)	new L.Bounds(…) L.bounds(…) L.bounds([…])	用两个坐标（通常是左上角的点和右下角的点）来创建边界对象。
L.Bounds(<Point[]> points)	new L.Bounds(…) L.bounds(…)	用包含的点创建边界对象。

Properties

Property	类型	描述
min	Point	矩形左上角点。
max	Point	矩形右下角点。

Methods

方法	返回值	描述
extend(<Point> point)	-	将包含给定点的边界延伸。
getCenter()	Point	返回边界的中心点。
contains(<Bounds> otherBounds)	Boolean	如果矩形包含给定的边界则返回true。
contains(<Point> point)	Boolean	如果矩形包含给定点则返回true。
intersects(<Bounds> otherBounds)	Boolean	如果矩形与给定边界相交则返回true。
isValid()	Boolean	如果边界可以被初始化则返回true。
getSize()	Point	返回边界的大小。

L.Icon

创建注记时显示的图标。

```
var myIcon = L.icon({
        iconUrl: 'my-icon.png',
        iconRetinaUrl: 'my-icon@2x.png',
        iconSize: [38, 95],
        iconAnchor: [22, 94],
        popupAnchor: [-3, -76],
        shadowUrl: 'my-icon-shadow.png',
        shadowRetinaUrl: 'my-icon-shadow@2x.png',
        shadowSize: [68, 95],
        shadowAnchor: [22, 94]
});

L.marker([50.505, 30.57], {icon: myIcon}).addTo(map);
```

L.Icon.默认值 extends L.Icon and is the blue icon Leaflet uses for markers by default.

构造器

构造器	使用	描述
L.Icon(<Icon options> options)	new L.Icon(…) L.icon(…)	通过给定的选项创建图标实例。

Options

选项	类型	描述
iconUrl	String	(required) 请求图标图片的URL（脚本中的绝对或相对路径）。

iconRetinaUrl	String	图标图片视网膜视图下的尺寸的URL。用于视网膜屏幕的设备。
iconSize	Point	图标图片的像素大小。
iconAnchor	Point	T图标提示的坐标（在左上角）。图标是对其的，所以这个点是注记的地理位置。如果大小是指定的则位于中心处，也可以在CSS中设置负边界。
shadowUrl	String	图标阴影图的URL。如果没有指定，图标没有阴影。
shadowRetinaUrl	String	图标在视网膜视图下的尺寸的URL。如果没有指定，图标没有阴影。用于视网膜屏幕的设备。
shadowSize	Point	Size of the shadow image in pixels.
shadowAnchor	Point	阴影的提示坐标（在左上角）（如果没有指定则与iconAnchor相同）。
popupAnchor	Point	与图标锚相关的打开弹出框的点的坐标。
className	String	图标和阴影图片的自定义的类名。默认为空。

L.DivIcon

用div要素而非图片来轻量级地显示注记的图标。

```
var myIcon = L.divIcon({className: 'my-div-icon'});
// you can set .my-div-icon styles in CSS

L.marker([50.505, 30.57], {icon: myIcon}).addTo(map);
```

默认情况下，阴影会有一个小的白色的方形作为leaflet-div-icon类和样式。

构造器

构造器	使用	描述
L.DivIcon(<DivIcon options> options)	new L.DivIcon(…) L.divIcon(…)	用给定的选项实例化图标。

Options

选项	类型	描述
iconSize	Point	图标的像素大小。也可以通过CSS设置。
iconAnchor	Point	图标提示的坐标（在左上角）。图标是对其的，所以这个点是注记的地理位置。如果大小是指定的则位于中心处，也可以在CSS中设置负边界。
className	String	用于对其图标的自定义的类名，默认为leaflet-div-icon。
html	String	在div要素中自定义的HTML代码，默认为空。

L.Control

所有leaflet控制的基础类。继承自IControl接口。 You can add controls to the map like this:

```
control.addTo(map);
// the same as
map.addControl(control);
```

构造器

构造器	使用	描述
`L.Control( <Control options> options? )`	`new L.Control(…)` `L.control(…)`	通过给定的选项创建一个控制。

Options

选项	类型	默认值	描述
`position`	`String`	`'topright'`	控制初始的位置（在地图的某一角）。参见 control positions.

Methods

方法	返回值	描述
`setPosition( <String> position )`	`this`	设置控制的位置。参见 control positions.
`getPosition()`	`String`	返回控制的当前位置。
`addTo( <Map> map )`	`this`	将控制添加到地图上。
`removeFrom( <Map> map )`	`this`	将控制从地图上移除。
`getContainer()`	`HTMLElement`	返回 the HTML container of the control.

Control Positions（控制的位置）

Control positions (map corner to put a control to) are set using strings. Margins between controls and the map border are set with CSS, so that you can easily override them.

Position	描述
`'topleft'`	地图的左上角。
`'topright'`	地图的右上角。
`'bottomleft'`	地图的左下角。

'bottomright'	地图的右下角。

L.Control.Zoom

拥有两个按钮（放大和缩小）的级别的缩放控制。默认地图上是有的，除非设置zoomControl选项为false。Extends Control.

构造器

构造器	使用	描述
L.Control.Zoom(<Control.Zoom options> options?)	new L.Control.Zoom(…) L.control.zoom(…)	创建缩放控制。

Options

选项	类型	默认值	描述
position	String	'topleft'	控制的位置（在地图的某一角）。参见 control positions.

L.Control.Attribution

可以在地图上一个小的文本盒子中显示属性数据的属性控制。默认地图上是有的，除非设置attributionControl选项为false，并且它自动地通过getAttribution方法获取图层的属性文本。继承自Control。

构造器

构造器	使用	描述
L.Control.Attribution(<Control.Attribution options> options?)	new L.Control.Attribution(…) L.control.attribution(…)	创建属性控制。

Options

选项	类型	默认值	描述
position	String	'bottomright'	控制的位置（在地图的某一角）。参见 control positions.
prefix	String	'Powered by Leaflet'	在属性之前显示的HTML文本。传递false来使其不显示。

Methods

方法	返回值	描述
`setPrefix( <String> prefix )`	this	在属性之前设置文本
`addAttribution( <String> text )`	this	添加属性文本。 (e.g. 'Vector data © CloudMade').
`removeAttribution( <String> text )`	this	移除属性文本。

L.Control.Layers

图层控制使用户可以在不同的底图之间切换，并可以控制覆盖物的开关。继承自Control.

```
var baseLayers = {
        "CloudMade": cloudmade,
        "OpenStreetMap": osm
};

var overlays = {
        "Marker": marker,
        "Roads": roadsLayer
};

L.control.layers(baseLayers, overlays).addTo(map);
```

构造器

构造器	使用	描述
`L.Control.Layers( <Layer Config> baseLayers?, <Layer Config> overlays?, <Control.Layers options> options? )`	`new L.Control.Layers(…)` `L.control.layers(…)`	通过给定的图层创建数据控制。基础图层通过单选项进行切换，覆盖物通过复选框切换显示。

Methods

方法	返回值	描述
`addBaseLayer( <ILayer> layer, <String> name )`	this	通过给定的控制名称添加基础层（通过单选按钮实体）。
`addOverlay( <ILayer> layer, <String> name )`	this	凸显给定的控制名称添加覆盖物（通过复选框实体）。
`removeLayer( <ILayer> layer )`	this	将图层从控制中移除。

Options

选项	类型	默认值	描述
position	String	'topright'	控制的位置（在地图的某一角）。参见 control positions.
collapsed	Boolean	true	如果为true，控制可以收缩为一个图标，在鼠标置于上方或点触时展开。
autoZIndex	Boolean	true	如果为true，控制的图层升序地叠置对齐，在切换图层打开或关闭时，顺便不变.

Layer Config

An object literal with layer names as keys and layer objects as values:

```
{
        "<someName1>": layer1,
        "<someName2>": layer2
}
```

The layer names can contain HTML, which allows you to add additional styling to the items:

```
{"<img src='my-layer-icon' /> <span class='my-layer-item'>My Layer</span>": myLayer}
```

Events

You can subscribe to the following events on the Map object using these methods.

Event	Data	描述
baselayerchange	LayersControlEvent	Fired when the base layer is changed through the control.
overlayadd	LayersControlEvent	Fired when an overlay is selected through the control.
overlayremove	LayersControlEvent	Fired when an overlay is deselected through the control.

L.Control.Scale

显示在公制(m/km)或英制(mi/ft)的屏幕当前中心的比例的简单比例尺控制。继承自 IControl interface.

```
L.control.scale().addTo(map);
```

构造器

构造器	使用	描述
L.Control.Scale(<Control.Scale options> options?)	new L.Control.Scale(…) L.control.scale(…)	通过选项创建比例控制。

Options

选项	类型	默认值	描述
position	String	'bottomleft'	控制器的位置（在地图的某一角）。参见 control positions.
maxWidth	Number	100	控制器最大的像素宽度。宽度可以围绕几个值动态设置。(e.g. 100, 200, 500).
metric	Boolean	true	是否显示公制比例尺 (m/km).
imperial	Boolean	true	是否显示英比例尺(mi/ft).
updateWhenIdle	Boolean	false	如果设置为true，控制由moveend更新，否则它总是最新的（由move更新）。

Events methods

一系列事件驱动的类（比如map）之间共享的方法。通常，事件允许你在一个对象发生某些事情时执行一些函数。

Example

```
map.on('click', function(e) {
        alert(e.latlng);
});
```

leaflet通过引用来处理事件监听器，所以如果你想咬添加或移除一个监听器时，可以用函数的方法。:

```
function onClick(e) { ... }

map.on('click', onClick);
map.off('click', onClick);
```

Methods

方法	返回值	描述
addEventListener(<String> type, <Function> fn, <Object> context?)	this	向某一类型的事件中添加监听器函数。你可以选择性地指定监听器的内容（对象中this关键字会被使用）。你也可以传递几个空格间隔的类型（如"click dbclick"）。
addOneTimeEventListener(<String> type, <Function> fn, <Object> context?) (类型,函数,内容)	this	The same as above except the listener will only get fired once and then removed.
addEventListener(<Object> eventMap, <Object> context?) (发生事件的地图,内容)	this	添加一系列的类型/监听器对, e.g. {click: onClick, mousemove: onMouseMove}
removeEventListener(<String>	this	移除之前添加的监听器函数。如果没有指定具体的函数，则所以的都会被移除。

type, <Function> fn?, <Object> context?)		
removeEventListener(<Object> eventMap, <Object> context?)	this	移除一系列类型/监听器对.
removeEventListener()	this	Removes all listeners. An alias to clearAllEventListeners when you use it without arguments.
hasEventListeners(<String> type)	Boolean	如果某一事件类型有附属的监听器则返回true。
fireEvent(<String> type, <Object> data?)	this	触发指定类型的事件。你可以提供一个数据对象——监听器对象的第一个参数应该包含它的属性。
clearAllEventListeners()	this	Removes all listeners to all events on the object.
on(…)	this	addEventListener的别称。
once(…)	this	Alias to addOneTimeEventListener.
off(…)	this	removeEventListener的别称。
fire(…)	this	fireEvent()的别称。

Event objects

当一些事件触发时接受监听器函数参数的事件对象，它包含了事件一些有用的信息。 For example:

```
map.on('click', function(e) {
        alert(e.latlng); // e is an event object (MouseEvent in this case)
});
```

Event

The base event object. All other event objects contain these properties too.

属性	类型	描述
type	String	事件的类型。 (e.g. 'click').
target	Object	触发事件的对象.

MouseEvent（鼠标事件）

property	type	description
latlng	LatLng	鼠标事件发生的地理点。

layerPoint	Point	鼠标事件发生的与地图图层相关的点的像素坐标。
containerPoint	Point	鼠标事件发生的与地图容器相关的点的像素坐标。
originalEvent	DOMMouseEvent	由浏览器触发的原始的DOM鼠标事件。

LocationEvent

property	type	description
latlng	LatLng	监测到的用户的地理位置.
bounds	LatLngBounds	用户坐落的区域的地理边界（考虑位置精度问题）。
accuracy	Number	米为单位的位置的精度。
altitude	Number	Height of the position above the WGS84 ellipsoid in meters.
altitudeAccuracy	Number	Accuracy of altitude in meters.
heading	Number	The direction of travel in degrees counting clockwise from true North.
speed	Number	Current velocity in meters per second.
timestamp	Number	The time when the position was acquired.

ErrorEvent（错误事件）

property	type	description
message	String	错误信息。
code	Number	错误代码（若可用）.

LayerEvent

property	type	description
layer	ILayer	添加或移除的图层。

LayersControlEvent

property	type	description
layer	ILayer	The layer that was added or removed.
name	String	The name of the layer that was added or removed.

TileEvent

property	type	description
tile	HTMLElement	切片要素（图片）。
url	String	切片的url源。

ResizeEvent

property	type	description
oldSize	Point	The old size before resize event.
newSize	Point	The new size after the resize event.

GeoJSON event（GeoJSON事件）

property	type	description
layer	ILayer	将要添加到地图上的GeoJSON要素的图层。
properties	Object	要素的GeoJSON的属性。
geometryType	String	要素的GeoJSON的几何类型。
id	String	要素的GeoJSON的ID（如果出现）.

Popup event（弹出框事件）

property	type	description
popup	Popup	打开或关闭的弹出框。

L.Class

L.Class强化了leaflet的面向对象的设备并被用于创建几乎所有这里提到的leaflet类。

除了执行一个简单的类接口模型，它还引入了方便代码组织的一下特殊的属性 — options, includes 和 statics.

```
var MyClass = L.Class.extend({
        initialize: function (greeter) {
                this.greeter = greeter;
                // class constructor
        },

        greet: function (name) {
                alert(this.greeter + ', ' + name)
        }
});
```

```
// create instance of MyClass, passing "Hello" to the constructor
var a = new MyClass("Hello");

// call greet method, alerting "Hello, World"
a.greet("World");
```

Inheritance（继承）

可以用L.Class.extend来定义新的类，但可以在任何一个类上用同样的方法来继承它:

```
var MyChildClass = MyClass.extend({
        // ... new properties and methods
});
```

这会创建一个继承父类所有方法和属性的类（由规范所约束），添加或重构你用来扩展的类。这也对instanceof做出反应:

```
var a = new MyChildClass();
a instanceof MyChildClass; // true
a instanceof MyClass; // true
```

你可以通过父类的规范和javascript的call与apply来调用父类的方法来响应子类的方法（就像你在其他语言中调用超类）:

```
var MyChildClass = MyClass.extend({
        initialize: function () {
                MyClass.prototype.initialize.call("Yo");
        },

        greet: function (name) {
                MyClass.prototype.greet.call(this, 'bro ' + name + '!');
        }
});

var a = new MyChildClass();
a.greet('Jason'); // alerts "Yo, bro Jason!"
```

Options（选项）

options 是一个与其他对象不同的特殊的属性，其他你用来扩展的对象会被父类合并而非完全重构，这使管理对象的结构和默认值更加方便:

```
var MyClass = L.Class.extend({
        options: {
                myOption1: 'foo',
                myOption2: 'bar'
        }
});

var MyChildClass = L.Class.extend({
        options: {
```

```
                myOption1: 'baz',
                myOption3: 5
        }
});

var a = new MyChildClass();
a.options.myOption1; // 'baz'
a.options.myOption2; // 'bar'
a.options.myOption3; // 5
```

选项中还有 `L.Util.setOptions`方法，可以方便地合并传递给函数构造器的选项和类中默认的定义:

```
var MyClass = L.Class.extend({
        options: {
                foo: 'bar',
                bla: 5
        },

        initialize: function (options) {
                L.Util.setOptions(this, options);
                ...
        }
});

var a = new MyClass({bla: 10});
a.options; // {foo: 'bar', bla: 10}
```

Includes（包含）

`includes` 是一个特殊的类，它将所有对象合并到一个类中。一个较好的例子是L.Mixin.Event，它是具有on、off和fire这些鱼事件相关的方法的类。

```
 var MyMixin = {
        foo: function () { ... },
        bar: 5
};

var MyClass = L.Class.extend({
        includes: MyMixin
});

var a = new MyClass();
a.foo();
```

You can also do such includes in runtime with the `include` method:

```
MyClass.include(MyMixin);
```

Statics（静态）

`statics`是一种方便的属性，将类中指定对象的属性变为静态属性，对于定义常量比较有用:

```
var MyClass = L.Class.extend({
	statics: {
		FOO: 'bar',
		BLA: 5
	}
});

MyClass.FOO; // 'bar'
```

Class Factories（类工厂）

你可以用个两种方式来创建leaflet的实例——用new关键字和用小写的factory方法:

```
new L.Map('map');
L.map('map');
```

The second way is implemented very easily, and you can do this for your own classes:

```
L.map = function (id, options) {
	return new L.Map(id, options);
};
```

Constructor Hooks（构造函数钩子）

如果你是一个插件开发者，你通常需要在现有的类中加入附件的初始化代码（比如因L.Polyline而编辑钩子）。leaflet可以用addInitHook方法来简化它：

```
MyClass.addInitHook(function () {
	// ... do something in constructor additionally
	// e.g. add event listeners, set custom properties etc.
});
```

You can also use the following shortcut when you just need to make one additional method call:

```
MyClass.addInitHook('methodName', arg1, arg2, …);
```

L.Browser

leaflet内部监测浏览器或要素的带有属性的命名空间。

```
if (L.Browser.ie6) {
	alert('Upgrade your browser, dude!');
}
```

property	type	description
ie	Boolean	如果是IE浏览器则返回true。
ie6	Boolean	如果是IE6浏览器则返回true。

ie7	Boolean	如果是IE7浏览器则返回true。
webkit	Boolean	如果是类似chrome和safari的基于webkit的浏览器（包括移动版）则返回true。
webkit3d	Boolean	如果基于webkit的浏览器支持CSS的3D转换则返回true。
android	Boolean	如果是安卓移动版的浏览器则返回true。
android23	Boolean	如果是安卓2或3的浏览器则返回true。
mobile	Boolean	如果是流行的移动版的浏览器（包括iOS下的safari和其他各种安卓浏览器）则返回true。
mobileWebkit	Boolean	如果是移动版的基于webkit的浏览器则返回true。
mobileOpera	Boolean	如果是移动版的opera浏览器则返回true。
touch	Boolean	对于所有触摸设备上的浏览器返回true。
msTouch	Boolean	对于微软的触摸模式的浏览器（比如IE10）返回true。
retina	Boolean	如果是视网膜屏幕的设备则返回true。

L.Util

在leaflet内部使用的多种实用的函数。

Methods

方法	返回值	描述
extend(<Object> dest, <Object> src?..)	Object	将src对象（或多个对象）的属性合并到dest对象中并将其返回。具有一个L.extend的快捷方式。
bind(<Function> fn, <Object> obj)	Function	返回由给定范围的obj执行fn函数的函数（所以关键字this可以表示函数代码里的obj）。具有一个L.bind快捷方式。
stamp(<Object> obj)	String	在对象上应用一个主键并返回这个键。具有L.stamp快捷方式。
limitExecByInterval(<Function> fn, <Number> time, <Object> context?)	Function	返回调用尽量快的但不会比间隔时间还要频繁的fn函数的包装器（对于拖动地图时检验和请求信的切片比较有用），可以通过context选择函数调用的范围。
falseFn()	Function	返回总是返回false的函数。
formatNum(<Number> num, <Number> digits)	Number	返回digits位数的num的数目。
splitWords(<String> str)	String[]	根据空格和空白来截取分割字符串并返回数组。
setOptions(<Object> obj,	Object	将所给的属性合并到obj的options中，返回最终的选项。参

<Object> options)		加Class options。具有L.setOptions快捷方式。
getParamString(<Object> obj)	String	将对象转换为带有参数的URL字符串,比如 {a: "foo", b: "bar"}转换为 '?a=foo&b=bar'.
template(<String> str, <Object> data)	String	是一个简单的模板，用通过将{a:'foo',b:'bar',...}形式的data对象应用到'Hello{a},{b}'形式的模板字符串来创建字符串——在上述示例中可以得到'Hello foo,bar'。
isArray(<Object> obj)	Boolean	如果对象为数组则返回true。
trim(<String> str)	String	Trims the whitespace from both ends of the string and returns the result.

Properties

Property	类型	描述
emptyImageUrl	String	包含64位编码的空的GIF图像的数据URL字符串。在webkit驱动的移动设备的上，用来作为清空没用图像的存储的钩子。

L.Transformation

表示仿射变换：用一系列a、b、c、d的系数来将（x，y）形式转换为（ax+b，cy+d）的形式并进行反转。在leaflet的投影代码中可以用得到。

```
var transformation = new L.Transformation(2, 5, -1, 10),
        p = new L.Point(1, 2),
        p2 = transformation.transform(p), // new L.Point(7, 8)
        p3 = transformation.untransform(p2); // new L.Point(1, 2)
```

构造器

构造器	使用	描述
L.Transformation(<Number> a, <Number> b, <Number> c, <Number> d)	new L.Transformation(…)	通过给定的系数创建转换对象.

Methods

方法	返回值	描述
transform(<Point> point, <Number> scale?)	Point	返回转换后的点，可以选择扩大一定的倍数。只接受真实的L.Point实例，而不是数组。
untransform(<Point> point, <Number> scale?)	Point	返回反转变换后的点，可以选择搜索一定倍数。只接受真实的L.Point实例，而不是数组。

L.LineUtil

一些处理线段点的应用函数，在leaflet内部用来使线段显示更快。

Methods

方法	返回值	描述
simplify(<Point[]> points, <Number> tolerance)	Point[]	在保持形状的同时动态地减少线上点的数目并返回简化后点的数组。在每一缩放级别处理和显示leaflet线段时可以大幅提升效率并可以减少视觉噪声。 tolerance影像简化的量（较小的值意味着更高的质量，但效率会地因为有更多的点）。这也是微型类库Simplify.js中的一部分.
pointToSegmentDistance(<Point> p, <Point> p1, <Point> p2)	Number	返回点p到p1和p2组成的线段之间的距离。
closestPointOnSegment(<Point> p, <Point> p1, <Point> p2)	Point	返回p1和p2线段上与p点最接近的点。
clipSegment(<Point> a, <Point> b, <Bounds> bounds)	-	用矩形边界裁剪点a到点b之间的折线段（直接修改折线段上的点）。在leaflet中用来显示屏幕内或边缘的线段上的点，可以因此而提高效率。

L.PolyUtil

多边形几何体的一些应用函数。

Methods

方法	返回值	描述
clipPolygon(<Point[]> points, <Bounds> bounds)	Point[]	通过矩形边界来裁剪给定点定义的多边形几何体。在leaflet中用来显示屏幕内或边缘的线段上的多边形上的点，可以因此而提高效率。多边形点需要不同的算法来裁剪折线段，因此这个方法也有不同的分支。

L.DomEvent

在leaflet内部用来处理DOM事件的应用函数

Methods

方法	返回值	描述
		向指定类型的DOM事件元素添加监听器fn。监听器中的this关

addListener(<HTMLElement> el, <String> type, <Function> fn, <Object> context?)	this	键字指向context，或是在没有说明的情况下指向要素。
removeListener(<HTMLElement> el, <String> type, <Function> fn)	this	在元素中移除事件监听器。

stopPropagation(<DOMEvent> e)	this	停止事件向父元素传播。Used inside the listener functions:
		`L.DomEvent.addListener(div, 'click', function (e) {`
		`  L.DomEvent.stopPropagation(e);`
		`});`

preventDefault(<DOMEvent> e)	this	阻止事件默认的动作发生（比如追踪元素href中的链接，或是当form提交时页面重载的POST请求）。
stop(<DOMEvent> e)	this	在同一时刻发起stopPropagation和preventDefault。
disableClickPropagation(<HTMLElement> el)	this	将stopPropagation添加到元素的'click','doubleclick','mousedown'和'touchstart'事件中。
getMousePosition(<DOMEvent> e, <HTMLElement> container?)	Point	如果没有特意说明则获取与容器或整个页面相关的DOM事件的标准的鼠标位置。
getWheelDelta(<DOMEvent> e)	Number	从mousewheel的DOM事件中获取标准的滚轮区域

L.DomUtil

在leaflet内部用来处理DOM树的应用函数。

Methods

方法	返回值	描述
get(<String or HTMLElement> id)	HTMLElement	如果传递字符串则返回一个带有指定id的元素，或是只是返回这个元素
getStyle(<HTMLElement> el, <String> style)	String	返回元素中特定样式属性的值，包括计算后的值和CSS中设置的值。
getViewportOffset(<HTMLElement> el)	Point	返回请求元素视图的偏移量。
create(<String> tagName, <String> className, <HTMLElement> container?)	HTMLElement	通过tagName创建元素，设置className并选择性地将其附加到container元素中。

disableTextSelection()	-	使文本不能被选择，比如拖动的时候。
enableTextSelection()	-	使文本选择重新可用。
hasClass(<HTMLElement> el, <String> name)	Boolean	如果元素类属性包含name则返回true。
addClass(<HTMLElement> el, <String> name)	-	将name添加到元素类的属性中。
removeClass(<HTMLElement> el, <String> name)	-	在元素类属性中移除name。
setOpacity(<HTMLElement> el, <Number> value)	-	设置元素的透明度（包括老的IE也支持）。值应当处于0到1之间。
testProp(<String[]> props)	String or false	检索样式名称的数组并返回第一个元素可用样式的名称。如果没有找到，那么返回false。
getTranslateString(<Point> point)	String	返回CSS转换字符串来通过给定点提供的偏移量来移动元素。
getScaleString(<Number> scale, <Point> origin)	String	返回CSS转换字符串来缩放元素（通过给定的比例原点）。
setPosition(<HTMLElement> el, <Point> point, <Boolean> disable3D?)	-	用CSS转换或屏幕左上角位置设置给定点的坐标系下的元素位置（leaflet内在地定位图层）。如果disable3D设置为true那么强制为左上角位置。
getPosition(<HTMLElement> el)	Point	返回之前用setPosition定位的元素的坐标。.

Properties（属性）

Property	类型	描述
TRANSITION	String	带前缀的转换样式名称（如'webkitTransition'用来表示WebKit）。
TRANSFORM	String	带前缀的变换样式名称。

L.PosAnimation

在内部用来平移动画镜头，利用CSS3转换在现代浏览器中实现，在IE6到9中用时间降速的功能实现。

```
var fx = new L.PosAnimation();
fx.run(el, [300, 500], 0.5);
```

构造器

构造器	使用	描述

L.PosAnimation()	new L.PosAnimation()	创建动画对象。

Methods

方法	返回值	描述
run(<HTMLElement> element, <Point> newPos, <Number> duration?, <Number> easeLinearity?)	this	在新的位置运行指定元素，可以选择性地设置持续的秒数（默认是0.25秒）和线性效果（通过cubic bezier curve的第三个参数，默认是0.5）。

Events

You can subscribe to the following events using these methods.

Event	Data	描述
start	Event	当动画开始时触发。
step	Event	在动画过程中持续触发。
end	Event	动画结束时触发。

L.Draggable

使DOM元素可以拖动的类。在内部被用来拖动地图和注记。

```
var draggable = new L.Draggable(elementToDrag);
draggable.enable();
```

构造器

构造器	使用	描述
L.Draggable(<HTMLElement> element, <HTMLElement> dragHandle?)	new L.Draggable(…)	创建可拖动对象，这样在你开始移动dragHandle元素时就可以移动给定元素了（默认同元素自身是同一个）。

Events

You can subscribe to the following events using these methods.

Event	Data	描述
dragstart	Event	拖动开始时触发。
predrag	Event	在拖动过程中相应元素位置更新之前持续触发。

drag	Event	拖动过程中持续触发。
dragend	Event	拖动结束后触发。

Methods

方法	返回值	描述
enable()	-	使拖动功能可用。
disable()	-	使拖动功能不可用。

IHandler

继承自interaction handlers接口.

方法	返回值	描述
enable()	-	使处理程序可用。
disable()	-	使处理程序不可用。
enabled()	Boolean	返如果处理程序可用则返回true。

ILayer

显示地图上附属于某一位置（或一系列位置）的对象。被 tile layers, markers, popups, image overlays, vector layers and layer groups所继承.

Methods

方法	返回值	描述
onAdd(<Map> map)	-	需要包含创建覆盖物的DOM元素的代码，将他们加入到所属的 map panes中并在相关地图时间中放入监听器。调用map.addLayer(map)。
onRemove(<Map> map)	-	包含从DOM移除覆盖物元素和移除之前onAdd方法添加的监听器的所有的清除代码。调用map.removeLayer(layer)。

Implementing Custom Layers（实例化自定义图层）

何时实例化自定义图层最重要的是地图的viewreset 事件和 latLngToLayerPoint 方法。viewset在地图需要重新定位图层时（比如缩放时）触发，latLngToLayerPoint在获取图层新的位置时使用。

在实例化图层时还有一个经常用到的事件是 moveend，在地图移动之后触发（比如平移和缩放等）。

还有一个需要注意的事情是你需要经常向你在图层中创建的DOM元素中添加leaflet-zoom-hide类，它会在缩放动画中隐藏。实例化自定义图层的 缩放动画师一个复杂的话题，在以后的章节中会讲到，但你可以在leaflet的图层代码（比如ImageOverlay）中看一下它是如何工作的。

Custom Layer Example

Here's how a custom layer implementation usually looks:

```
var MyCustomLayer = L.Class.extend({

	initialize: function (latlng) {
		// save position of the layer or any options from the constructor
		this._latlng = latlng;
	},

	onAdd: function (map) {
		this._map = map;

		// create a DOM element and put it into one of the map panes
		this._el = L.DomUtil.create('div', 'my-custom-layer leaflet-zoom-hide');
		map.getPanes().overlayPane.appendChild(this._el);

		// add a viewreset event listener for updating layer's position, do the latter
		map.on('viewreset', this._reset, this);
		this._reset();
	},

	onRemove: function (map) {
		// remove layer's DOM elements and listeners
		map.getPanes().overlayPane.removeChild(this._el);
		map.off('viewreset', this._reset, this);
	},

	_reset: function () {
		// update layer's position
		var pos = this._map.latLngToLayerPoint(this._latlng);
		L.DomUtil.setPosition(this._el, pos);
	}
});

map.addLayer(new MyCustomLayer(latlng));
```

IControl

在地图的某个角上显示UI元素。被 zoom, attribution, scale and layers controls所继承.

Methods

Every control in Leaflet should extend from Control class and additionally have the following methods:

方法	返回值	描述
onAdd(<Map> map)	HTMLElement	包含所有用于在相关地图事件上控制、添加监听器的创建必要DOM要素的代码，并返回包含控制的元素。调用map.addControl(control)或control.addTo(map)。
onRemove(<Map> map)	-	包含所有清除代码（比如0移除控制事件监听器）。调用map.removeControl(control)或control.removeFrom(map)。控制的DOM容器自动移除。

Custom Control Example

```
var MyControl = L.Control.extend({
        options: {
                position: 'topright'
        },

        onAdd: function (map) {
                // create the control container with a particular class name
                var container = L.DomUtil.create('div', 'my-custom-control');

                // ... initialize other DOM elements, add listeners, etc.

                return container;
        }
});

map.addControl(new MyControl());
```

If specify your own constructor for the control, you'll also probably want to process options properly:

```
var MyControl = L.Control.extend({
        initialize: function (foo, options) {
                // ...
                L.Util.setOptions(this, options);
        },
        // ...
});
```

This will allow you to pass options like `position` when creating the control instances:

```
map.addControl(new MyControl('bar', {position: 'bottomleft'}));
```

IProjection

具有将地理坐标投影到平面（和后方）的方法的对象。参见 Map projection.

Methods

方法	返回值	描述
project(<LatLng> latlng)	Point	将地理坐标投影为二维点。
unproject(<Point> point)	LatLng	将二维的点反投影为地理位置。

Defined Projections

Leaflet comes with a set of already defined projections out of the box:

Projection	描述
L.Projection.SphericalMercator	球面墨卡托投影——网上地图最常用的投影，几乎所有的免费和商业的切片提供者都会使用。假设地球是一个规则球体。被EPSG:3857坐标参考系统使用。
L.Projection.Mercator	椭圆墨卡托投影——比球面墨卡托投影更为复杂，这个投影考虑到地球是椭球而非规则球体。在EPSG:3395坐标参考系统中使用。
L.Projection.LonLat	正交矩形或圆柱投影——最简单的投影，几乎只被GIS专家使用。将地图的x方向作为经度，y方向作为纬度。对于平面的世界也适用，比如游戏地图。在EPSG:3395和Simple坐标参考系统中使用。

ICRS

为将地理点投影到像素坐标或屏幕坐标和反向投影（投影到用于WMS服务的其他单位的坐标）而定义坐标参考系统。参见 Spatial reference system.

Methods

方法	返回值	描述
latLngToPoint(<LatLng> latlng, <Number> zoom)	Point	将给定缩放级别的地理坐标投影为像素坐标。
pointToLatLng(<Point> point, <Number> zoom)	LatLng	是latLngToPoint的反转。将给定缩放级别的像素坐标投影为地理坐标。
project(<LatLng> latlng)	Point	将地理坐标投影为CRS可接受单位的坐标（比如EPSG:3857中的米，传递给WMS服务）。
scale(<Number> zoom)	Number	返回转换投影坐标为特定级别的像素坐标所用到的缩放级别。比如，在基于墨卡托投影的CRS中返回256*2^zoom。

Properties

Property	类型	描述
projection	IProjection	CRS使用的投影。

transformation	Transformation	CRS使用的用来将投影坐标转换为特定切片服务的屏幕坐标的转换方式。
code	String	向WMS服务传递的标准CRS的标准代码名称（比如'EPSG:3857'）。

Defined CRS（定义的坐标参考系统）

Leaflet comes with a set of already defined CRS to use out of the box:

Projection	描述
L.CRS.EPSG3857	在线地图最常用的CRS，几乎所有的免费喝商业切片服务都会使用。使用球面墨卡托投影。是地图的crs选项的开始默认值。
L.CRS.EPSG4326	在GIS专家中常见的CRS。使用简单的圆柱投影。
L.CRS.EPSG3395	较少地被商业切片服务使用。使用椭圆墨卡托投影。
L.CRS.Simple	直接将经纬和纬度映射为x和y的简单的CRS。可能会在平面地图中用到（比如游戏地图）。y轴始终是反向的（由下而上）。

如果你想用其他此处未列出的不常用的CRS，请查询 Proj4Leaflet 插件.

Global Switches

用来在少数情况并且基本都是即使某个特别的浏览器要素存在，也使leaflet不监测的全局设置开关。需要在leaflet包含于页面之前将全局变量开关设置为true, like this:

```
<script>L_PREFER_CANVAS = true;</script>
<script src="leaflet.js"></script>
```

Switch	描述
L_PREFER_CANVAS	对于矢量图层，强制leaflet在后台使用画布而非SVG。这在某些情况下可以适当提高性能（比如在地图上有成千上万的圆注记时）。
L_NO_TOUCH	即使监测到触摸事件，也强制leaflet不去使用触摸事件。
L_DISABLE_3D	即使可用，也强制leaflet在定位时不使用硬件加速来进行CSS 3D变换（在少数情况下会发生偶然故障）。

L.noConflict()

这个方法用来将L全局变量恢复到leaflet包含的初始值，并返回leaflet真实的命名空间，所以你可以将它放到任何地方， like this:

```
// L points to some other library
...
// you include Leaflet, it replaces the L variable to Leaflet namespace
```

```
var Leaflet = L.noConflict();
// now L points to that other library again, and you can use Leaflet.Map etc.
```

L.version

显示当前使用的leaflet版本的常量。

```
L.version // returns "0.5" (or whatever version is currently in use)
```