----Layer----- cordova.js:5720

key: options

value: [object Object] cordova.js:5720

key: \_latlng

value: LatLng(53.34438, -6.26962) cordova.js:5720

key: \_initHooksCalled

value: true cordova.js:5720

key: \_leaflet\_id

value: 216 cordova.js:5720

key: \_\_parent

value: [object Object] cordova.js:5720

key: \_leaflet\_events

value: [object Object] cordova.js:5720

key: \_map

value: [object Object] cordova.js:5720

key: \_icon

value: [object HTMLImageElement] cordova.js:5720

key: dragging

value: [object Object] cordova.js:5720

key: \_shadow

value: [object HTMLImageElement] cordova.js:5720

key: \_zIndex

value: 343 cordova.js:5720

key: constructor

value: function (){this.initialize&&this.initialize.apply(this,arguments),this.\_initHooks&&this.callInitHooks()} cordova.js:5720

key: addEventListener

value: function (t,e,i){if(s.Util.invokeEach(t,this.addEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p=this[a]=this[a]||{},d=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)r={action:e,context:i||this},l=t[n],d?(h=l+"\_idx",u=h+"\_len",c=p[h]=p[h]||{},c[d]||(c[d]=[],p[u]=(p[u]||0)+1),c[d].push(r)):(p[l]=p[l]||[],p[l].push(r));return this} cordova.js:5720

key: hasEventListeners

value: function (t){var e=this[a];return!!e&&(t in e&&e[t].length>0||t+"\_idx"in e&&e[t+"\_idx\_len"]>0)} cordova.js:5720

key: removeEventListener

value: function (t,e,i){if(!this[a])return this;if(!t)return this.clearAllEventListeners();if(s.Util.invokeEach(t,this.removeEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p,d,m=this[a],f=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)if(r=t[n],u=r+"\_idx",c=u+"\_len",p=m[u],e){if(l=f&&p?p[f]:m[r]){for(h=l.length-1;h>=0;h--)l[h].action!==e||i&&l[h].context!==i||(d=l.splice(h,1),d[0].action=s.Util.falseFn);i&&p&&0===l.length&&(delete p[f],m[c]--)}}else delete m[r],delete m[u],delete m[c];return this} cordova.js:5720

key: clearAllEventListeners

value: function (){return delete this[a],this} cordova.js:5720

key: fireEvent

value: function (t,e){if(!this.hasEventListeners(t))return this;var i,n,o,r,l,h=s.Util.extend({},e,{type:t,target:this}),u=this[a];if(u[t])for(i=u[t].slice(),n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);r=u[t+"\_idx"];for(l in r)if(i=r[l].slice())for(n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);return this} cordova.js:5720

key: addOneTimeEventListener

value: function (t,e,i){if(s.Util.invokeEach(t,this.addOneTimeEventListener,this,e,i))return this;var n=s.bind(function(){this.removeEventListener(t,e,i).removeEventListener(t,n,i)},this);return this.addEventListener(t,e,i).addEventListener(t,n,i)} cordova.js:5720

key: on

value: function (t,e,i){if(s.Util.invokeEach(t,this.addEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p=this[a]=this[a]||{},d=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)r={action:e,context:i||this},l=t[n],d?(h=l+"\_idx",u=h+"\_len",c=p[h]=p[h]||{},c[d]||(c[d]=[],p[u]=(p[u]||0)+1),c[d].push(r)):(p[l]=p[l]||[],p[l].push(r));return this} cordova.js:5720

key: off

value: function (t,e,i){if(!this[a])return this;if(!t)return this.clearAllEventListeners();if(s.Util.invokeEach(t,this.removeEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p,d,m=this[a],f=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)if(r=t[n],u=r+"\_idx",c=u+"\_len",p=m[u],e){if(l=f&&p?p[f]:m[r]){for(h=l.length-1;h>=0;h--)l[h].action!==e||i&&l[h].context!==i||(d=l.splice(h,1),d[0].action=s.Util.falseFn);i&&p&&0===l.length&&(delete p[f],m[c]--)}}else delete m[r],delete m[u],delete m[c];return this} cordova.js:5720

key: once

value: function (t,e,i){if(s.Util.invokeEach(t,this.addOneTimeEventListener,this,e,i))return this;var n=s.bind(function(){this.removeEventListener(t,e,i).removeEventListener(t,n,i)},this);return this.addEventListener(t,e,i).addEventListener(t,n,i)} cordova.js:5720

key: fire

value: function (t,e){if(!this.hasEventListeners(t))return this;var i,n,o,r,l,h=s.Util.extend({},e,{type:t,target:this}),u=this[a];if(u[t])for(i=u[t].slice(),n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);r=u[t+"\_idx"];for(l in r)if(i=r[l].slice())for(n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);return this} cordova.js:5720

key: initialize

value: function (t,e){s.setOptions(this,e),this.\_latlng=s.latLng(t)} cordova.js:5720

key: onAdd

value: function (t){this.\_map=t,t.on("viewreset",this.update,this),this.\_initIcon(),this.update(),this.fire("add"),t.options.zoomAnimation&&t.options.markerZoomAnimation&&t.on("zoomanim",this.\_animateZoom,this)} cordova.js:5720

key: addTo

value: function (t){return t.addLayer(this),this} cordova.js:5720

key: onRemove

value: function (t){this.dragging&&this.dragging.disable(),this.\_removeIcon(),this.\_removeShadow(),this.fire("remove"),t.off({viewreset:this.update,zoomanim:this.\_animateZoom},this),this.\_map=null} cordova.js:5720

key: getLatLng

value: function (){return this.\_latlng} cordova.js:5720

key: setLatLng

value: function (t){return this.\_latlng=s.latLng(t),this.update(),this.fire("move",{latlng:this.\_latlng})} cordova.js:5720

key: setZIndexOffset

value: function (t){return this.options.zIndexOffset=t,this.update(),this} cordova.js:5720

key: setIcon

value: function (t){return this.options.icon=t,this.\_map&&(this.\_initIcon(),this.update()),this.\_popup&&this.bindPopup(this.\_popup),this} cordova.js:5720

key: update

value: function (){if(this.\_icon){var t=this.\_map.latLngToLayerPoint(this.\_latlng).round();this.\_setPos(t)}return this} cordova.js:5720

key: \_initIcon

value: function (){var t=this.options,e=this.\_map,i=e.options.zoomAnimation&&e.options.markerZoomAnimation,n=i?"leaflet-zoom-animated":"leaflet-zoom-hide",o=t.icon.createIcon(this.\_icon),a=!1;o!==this.\_icon&&(this.\_icon&&this.\_removeIcon(),a=!0,t.title&&(o.title=t.title),t.alt&&(o.alt=t.alt)),s.DomUtil.addClass(o,n),t.keyboard&&(o.tabIndex="0"),this.\_icon=o,this.\_initInteraction(),t.riseOnHover&&s.DomEvent.on(o,"mouseover",this.\_bringToFront,this).on(o,"mouseout",this.\_resetZIndex,this);var r=t.icon.createShadow(this.\_shadow),l=!1;r!==this.\_shadow&&(this.\_removeShadow(),l=!0),r&&s.DomUtil.addClass(r,n),this.\_shadow=r,t.opacity<1&&this.\_updateOpacity();var h=this.\_map.\_panes;a&&h.markerPane.appendChild(this.\_icon),r&&l&&h.shadowPane.appendChild(this.\_shadow)} cordova.js:5720

key: \_removeIcon

value: function (){this.options.riseOnHover&&s.DomEvent.off(this.\_icon,"mouseover",this.\_bringToFront).off(this.\_icon,"mouseout",this.\_resetZIndex),this.\_map.\_panes.markerPane.removeChild(this.\_icon),this.\_icon=null} cordova.js:5720

key: \_removeShadow

value: function (){this.\_shadow&&this.\_map.\_panes.shadowPane.removeChild(this.\_shadow),this.\_shadow=null} cordova.js:5720

key: \_setPos

value: function (t){s.DomUtil.setPosition(this.\_icon,t),this.\_shadow&&s.DomUtil.setPosition(this.\_shadow,t),this.\_zIndex=t.y+this.options.zIndexOffset,this.\_resetZIndex()} cordova.js:5720

key: \_updateZIndex

value: function (t){this.\_icon.style.zIndex=this.\_zIndex+t} cordova.js:5720

key: \_animateZoom

value: function (t){var e=this.\_map.\_latLngToNewLayerPoint(this.\_latlng,t.zoom,t.center).round();this.\_setPos(e)} cordova.js:5720

key: \_initInteraction

value: function (){if(this.options.clickable){var t=this.\_icon,e=["dblclick","mousedown","mouseover","mouseout","contextmenu"];s.DomUtil.addClass(t,"leaflet-clickable"),s.DomEvent.on(t,"click",this.\_onMouseClick,this),s.DomEvent.on(t,"keypress",this.\_onKeyPress,this);for(var i=0;i<e.length;i++)s.DomEvent.on(t,e[i],this.\_fireMouseEvent,this);s.Handler.MarkerDrag&&(this.dragging=new s.Handler.MarkerDrag(this),this.options.draggable&&this.dragging.enable())}} cordova.js:5720

key: \_onMouseClick

value: function (t){var e=this.dragging&&this.dragging.moved();(this.hasEventListeners(t.type)||e)&&s.DomEvent.stopPropagation(t),e||(this.dragging&&this.dragging.\_enabled||!this.\_map.dragging||!this.\_map.dragging.moved())&&this.fire(t.type,{originalEvent:t,latlng:this.\_latlng})} cordova.js:5720

key: \_onKeyPress

value: function (t){13===t.keyCode&&this.fire("click",{originalEvent:t,latlng:this.\_latlng})} cordova.js:5720

key: \_fireMouseEvent

value: function (t){this.fire(t.type,{originalEvent:t,latlng:this.\_latlng}),"contextmenu"===t.type&&this.hasEventListeners(t.type)&&s.DomEvent.preventDefault(t),"mousedown"!==t.type?s.DomEvent.stopPropagation(t):s.DomEvent.preventDefault(t)} cordova.js:5720

key: setOpacity

value: function (t){return this.options.opacity=t,this.\_map&&this.\_updateOpacity(),this} cordova.js:5720

key: \_updateOpacity

value: function (){s.DomUtil.setOpacity(this.\_icon,this.options.opacity),this.\_shadow&&s.DomUtil.setOpacity(this.\_shadow,this.options.opacity)} cordova.js:5720

key: \_bringToFront

value: function (){this.\_updateZIndex(this.options.riseOffset)} cordova.js:5720

key: \_resetZIndex

value: function (){this.\_updateZIndex(0)} cordova.js:5720

key: \_initHooks

value: cordova.js:5720

key: callInitHooks

value: function (){if(!this.\_initHooksCalled){a.prototype.callInitHooks&&a.prototype.callInitHooks.call(this),this.\_initHooksCalled=!0;for(var t=0,e=n.\_initHooks.length;e>t;t++)n.\_initHooks[t].call(this)}} cordova.js:5720

key: openPopup

value: function (){return this.\_popup&&this.\_map&&!this.\_map.hasLayer(this.\_popup)&&(this.\_popup.setLatLng(this.\_latlng),this.\_map.openPopup(this.\_popup)),this} cordova.js:5720

key: closePopup

value: function (){return this.\_popup&&this.\_popup.\_close(),this} cordova.js:5720

key: togglePopup

value: function (){return this.\_popup&&(this.\_popup.\_isOpen?this.closePopup():this.openPopup()),this} cordova.js:5720

key: bindPopup

value: function (t,e){var i=s.point(this.options.icon.options.popupAnchor||[0,0]);return i=i.add(s.Popup.prototype.options.offset),e&&e.offset&&(i=i.add(e.offset)),e=s.extend({offset:i},e),this.\_popupHandlersAdded||(this.on("click",this.togglePopup,this).on("remove",this.closePopup,this).on("move",this.\_movePopup,this),this.\_popupHandlersAdded=!0),t instanceof s.Popup?(s.setOptions(t,e),this.\_popup=t):this.\_popup=new s.Popup(e,this).setContent(t),this} cordova.js:5720

key: setPopupContent

value: function (t){return this.\_popup&&this.\_popup.setContent(t),this} cordova.js:5720

key: unbindPopup

value: function (){return this.\_popup&&(this.\_popup=null,this.off("click",this.togglePopup,this).off("remove",this.closePopup,this).off("move",this.\_movePopup,this),this.\_popupHandlersAdded=!1),this} cordova.js:5720

key: getPopup

value: function (){return this.\_popup} cordova.js:5720

key: \_movePopup

value: function (t){this.\_popup.setLatLng(t.latlng)} cordova.js:5720

key: toGeoJSON

value: function (){return s.GeoJSON.getFeature(this,{type:"Point",coordinates:s.GeoJSON.latLngToCoords(this.getLatLng())})} cordova.js:5720

----Target----- cordova.js:5720

key: options

value: [object Object] cordova.js:5720

key: \_featureGroup

value: [object Object] cordova.js:5720

key: \_leaflet\_id

value: 26 cordova.js:5720

key: \_nonPointGroup

value: [object Object] cordova.js:5720

key: \_inZoomAnimation

value: 0 cordova.js:5720

key: \_needsClustering

value: cordova.js:5720

key: \_needsRemoving

value: cordova.js:5720

key: \_currentShownBounds

value: [object Object] cordova.js:5720

key: \_queue

value: cordova.js:5720

key: \_initHooksCalled

value: true cordova.js:5720

key: \_leaflet\_events

value: [object Object] cordova.js:5720

key: \_map

value: [object Object] cordova.js:5720

key: \_maxZoom

value: 19 cordova.js:5720

key: \_gridClusters

value: [object Object] cordova.js:5720

key: \_gridUnclustered

value: [object Object] cordova.js:5720

key: \_topClusterLevel

value: [object Object] cordova.js:5720

key: \_zoom

value: 16 cordova.js:5720

key: \_queueTimeout

value: null cordova.js:5720

key: constructor

value: function (){this.initialize&&this.initialize.apply(this,arguments),this.\_initHooks&&this.callInitHooks()} cordova.js:5720

key: initialize

value: function (options) {

L.Util.setOptions(this, options);

if (!this.options.iconCreateFunction) {

this.options.iconCreateFunction = this.\_defaultIconCreateFunction;

}

this.\_featureGroup = L.featureGroup();

this.\_featureGroup.on(L.FeatureGroup.EVENTS, this.\_propagateEvent, this);

this.\_nonPointGroup = L.featureGroup();

this.\_nonPointGroup.on(L.FeatureGroup.EVENTS, this.\_propagateEvent, this);

this.\_inZoomAnimation = 0;

this.\_needsClustering = [];

this.\_needsRemoving = []; //Markers removed while we aren't on the map need to be kept track of

//The bounds of the currently shown area (from \_getExpandedVisibleBounds) Updated on zoom/move

this.\_currentShownBounds = null;

this.\_queue = [];

} cordova.js:5720

key: addLayer

value: function (layer) {

if (layer instanceof L.LayerGroup) {

var array = [];

for (var i in layer.\_layers) {

array.push(layer.\_layers[i]);

}

return this.addLayers(array);

}

//Don't cluster non point data

if (!layer.getLatLng) {

this.\_nonPointGroup.addLayer(layer);

return this;

}

if (!this.\_map) {

this.\_needsClustering.push(layer);

return this;

}

if (this.hasLayer(layer)) {

return this;

}

//If we have already clustered we'll need to add this one to a cluster

if (this.\_unspiderfy) {

this.\_unspiderfy();

}

this.\_addLayer(layer, this.\_maxZoom);

//Work out what is visible

var visibleLayer = layer,

currentZoom = this.\_map.getZoom();

if (layer.\_\_parent) {

while (visibleLayer.\_\_parent.\_zoom >= currentZoom) {

visibleLayer = visibleLayer.\_\_parent;

}

}

if (this.\_currentShownBounds.contains(visibleLayer.getLatLng())) {

if (this.options.animateAddingMarkers) {

this.\_animationAddLayer(layer, visibleLayer);

} else {

this.\_animationAddLayerNonAnimated(layer, visibleLayer);

}

}

return this;

} cordova.js:5720

key: removeLayer

value: function (layer) {

if (layer instanceof L.LayerGroup)

{

var array = [];

for (var i in layer.\_layers) {

array.push(layer.\_layers[i]);

}

return this.removeLayers(array);

}

//Non point layers

if (!layer.getLatLng) {

this.\_nonPointGroup.removeLayer(layer);

return this;

}

if (!this.\_map) {

if (!this.\_arraySplice(this.\_needsClustering, layer) && this.hasLayer(layer)) {

this.\_needsRemoving.push(layer);

}

return this;

}

if (!layer.\_\_parent) {

return this;

}

if (this.\_unspiderfy) {

this.\_unspiderfy();

this.\_unspiderfyLayer(layer);

}

//Remove the marker from clusters

this.\_removeLayer(layer, true);

if (this.\_featureGroup.hasLayer(layer)) {

this.\_featureGroup.removeLayer(layer);

if (layer.setOpacity) {

layer.setOpacity(1);

}

}

return this;

} cordova.js:5720

key: addLayers

value: function (layersArray) {

var fg = this.\_featureGroup,

npg = this.\_nonPointGroup,

chunked = this.options.chunkedLoading,

chunkInterval = this.options.chunkInterval,

chunkProgress = this.options.chunkProgress,

newMarkers, i, l, m;

if (this.\_map) {

var offset = 0,

started = (new Date()).getTime();

var process = L.bind(function () {

var start = (new Date()).getTime();

for (; offset < layersArray.length; offset++) {

if (chunked && offset % 200 === 0) {

// every couple hundred markers, instrument the time elapsed since processing started:

var elapsed = (new Date()).getTime() - start;

if (elapsed > chunkInterval) {

break; // been working too hard, time to take a break :-)

}

}

m = layersArray[offset];

//Not point data, can't be clustered

if (!m.getLatLng) {

npg.addLayer(m);

continue;

}

if (this.hasLayer(m)) {

continue;

}

this.\_addLayer(m, this.\_maxZoom);

//If we just made a cluster of size 2 then we need to remove the other marker from the map (if it is) or we never will

if (m.\_\_parent) {

if (m.\_\_parent.getChildCount() === 2) {

var markers = m.\_\_parent.getAllChildMarkers(),

otherMarker = markers[0] === m ? markers[1] : markers[0];

fg.removeLayer(otherMarker);

}

}

}

if (chunkProgress) {

// report progress and time elapsed:

chunkProgress(offset, layersArray.length, (new Date()).getTime() - started);

}

if (offset === layersArray.length) {

//Update the icons of all those visible clusters that were affected

this.\_featureGroup.eachLayer(function (c) {

if (c instanceof L.MarkerCluster && c.\_iconNeedsUpdate) {

c.\_updateIcon();

}

});

this.\_topClusterLevel.\_recursivelyAddChildrenToMap(null, this.\_zoom, this.\_currentShownBounds);

} else {

setTimeout(process, this.options.chunkDelay);

}

}, this);

process();

} else {

newMarkers = [];

for (i = 0, l = layersArray.length; i < l; i++) {

m = layersArray[i];

//Not point data, can't be clustered

if (!m.getLatLng) {

npg.addLayer(m);

continue;

}

if (this.hasLayer(m)) {

continue;

}

newMarkers.push(m);

}

this.\_needsClustering = this.\_needsClustering.concat(newMarkers);

}

return this;

} cordova.js:5720

key: removeLayers

value: function (layersArray) {

var i, l, m,

fg = this.\_featureGroup,

npg = this.\_nonPointGroup;

if (!this.\_map) {

for (i = 0, l = layersArray.length; i < l; i++) {

m = layersArray[i];

this.\_arraySplice(this.\_needsClustering, m);

npg.removeLayer(m);

}

return this;

}

for (i = 0, l = layersArray.length; i < l; i++) {

m = layersArray[i];

if (!m.\_\_parent) {

npg.removeLayer(m);

continue;

}

this.\_removeLayer(m, true, true);

if (fg.hasLayer(m)) {

fg.removeLayer(m);

if (m.setOpacity) {

m.setOpacity(1);

}

}

}

//Fix up the clusters and markers on the map

this.\_topClusterLevel.\_recursivelyAddChildrenToMap(null, this.\_zoom, this.\_currentShownBounds);

fg.eachLayer(function (c) {

if (c instanceof L.MarkerCluster) {

c.\_updateIcon();

}

});

return this;

} cordova.js:5720

key: clearLayers

value: function () {

//Need our own special implementation as the LayerGroup one doesn't work for us

//If we aren't on the map (yet), blow away the markers we know of

if (!this.\_map) {

this.\_needsClustering = [];

delete this.\_gridClusters;

delete this.\_gridUnclustered;

}

if (this.\_noanimationUnspiderfy) {

this.\_noanimationUnspiderfy();

}

//Remove all the visible layers

this.\_featureGroup.clearLayers();

this.\_nonPointGroup.clearLayers();

this.eachLayer(function (marker) {

delete marker.\_\_parent;

});

if (this.\_map) {

//Reset \_topClusterLevel and the DistanceGrids

this.\_generateInitialClusters();

}

return this;

} cordova.js:5720

key: getBounds

value: function () {

var bounds = new L.LatLngBounds();

if (this.\_topClusterLevel) {

bounds.extend(this.\_topClusterLevel.\_bounds);

}

for (var i = this.\_needsClustering.length - 1; i >= 0; i--) {

bounds.extend(this.\_needsClustering[i].getLatLng());

}

bounds.extend(this.\_nonPointGroup.getBounds());

return bounds;

} cordova.js:5720

key: eachLayer

value: function (method, context) {

var markers = this.\_needsClustering.slice(),

i;

if (this.\_topClusterLevel) {

this.\_topClusterLevel.getAllChildMarkers(markers);

}

for (i = markers.length - 1; i >= 0; i--) {

method.call(context, markers[i]);

}

this.\_nonPointGroup.eachLayer(method, context);

} cordova.js:5720

key: getLayers

value: function () {

var layers = [];

this.eachLayer(function (l) {

layers.push(l);

});

return layers;

} cordova.js:5720

key: getLayer

value: function (id) {

var result = null;

this.eachLayer(function (l) {

if (L.stamp(l) === id) {

result = l;

}

});

return result;

} cordova.js:5720

key: hasLayer

value: function (layer) {

if (!layer) {

return false;

}

var i, anArray = this.\_needsClustering;

for (i = anArray.length - 1; i >= 0; i--) {

if (anArray[i] === layer) {

return true;

}

}

anArray = this.\_needsRemoving;

for (i = anArray.length - 1; i >= 0; i--) {

if (anArray[i] === layer) {

return false;

}

}

return !!(layer.\_\_parent && layer.\_\_parent.\_group === this) || this.\_nonPointGroup.hasLayer(layer);

} cordova.js:5720

key: zoomToShowLayer

value: function (layer, callback) {

var showMarker = function () {

if ((layer.\_icon || layer.\_\_parent.\_icon) && !this.\_inZoomAnimation) {

this.\_map.off('moveend', showMarker, this);

this.off('animationend', showMarker, this);

if (layer.\_icon) {

callback();

} else if (layer.\_\_parent.\_icon) {

var afterSpiderfy = function () {

this.off('spiderfied', afterSpiderfy, this);

callback();

};

this.on('spiderfied', afterSpiderfy, this);

layer.\_\_parent.spiderfy();

}

}

};

if (layer.\_icon && this.\_map.getBounds().contains(layer.getLatLng())) {

callback();

} else if (layer.\_\_parent.\_zoom < this.\_map.getZoom()) {

//Layer should be visible now but isn't on screen, just pan over to it

this.\_map.on('moveend', showMarker, this);

this.\_map.panTo(layer.getLatLng());

} else {

this.\_map.on('moveend', showMarker, this);

this.on('animationend', showMarker, this);

this.\_map.setView(layer.getLatLng(), layer.\_\_parent.\_zoom + 1);

layer.\_\_parent.zoomToBounds();

}

} cordova.js:5720

key: onAdd

value: function (map) {

this.\_map = map;

var i, l, layer;

if (!isFinite(this.\_map.getMaxZoom())) {

throw "Map has no maxZoom specified";

}

this.\_featureGroup.onAdd(map);

this.\_nonPointGroup.onAdd(map);

if (!this.\_gridClusters) {

this.\_generateInitialClusters();

}

for (i = 0, l = this.\_needsRemoving.length; i < l; i++) {

layer = this.\_needsRemoving[i];

this.\_removeLayer(layer, true);

}

this.\_needsRemoving = [];

//Remember the current zoom level and bounds

this.\_zoom = this.\_map.getZoom();

this.\_currentShownBounds = this.\_getExpandedVisibleBounds();

this.\_map.on('zoomend', this.\_zoomEnd, this);

this.\_map.on('moveend', this.\_moveEnd, this);

if (this.\_spiderfierOnAdd) { //TODO FIXME: Not sure how to have spiderfier add something on here nicely

this.\_spiderfierOnAdd();

}

this.\_bindEvents();

//Actually add our markers to the map:

l = this.\_needsClustering;

this.\_needsClustering = [];

this.addLayers(l);

} cordova.js:5720

key: onRemove

value: function (map) {

map.off('zoomend', this.\_zoomEnd, this);

map.off('moveend', this.\_moveEnd, this);

this.\_unbindEvents();

//In case we are in a cluster animation

this.\_map.\_mapPane.className = this.\_map.\_mapPane.className.replace(' leaflet-cluster-anim', '');

if (this.\_spiderfierOnRemove) { //TODO FIXME: Not sure how to have spiderfier add something on here nicely

this.\_spiderfierOnRemove();

}

//Clean up all the layers we added to the map

this.\_hideCoverage();

this.\_featureGroup.onRemove(map);

this.\_nonPointGroup.onRemove(map);

this.\_featureGroup.clearLayers();

this.\_map = null;

} cordova.js:5720

key: getVisibleParent

value: function (marker) {

var vMarker = marker;

while (vMarker && !vMarker.\_icon) {

vMarker = vMarker.\_\_parent;

}

return vMarker || null;

} cordova.js:5720

key: \_arraySplice

value: function (anArray, obj) {

for (var i = anArray.length - 1; i >= 0; i--) {

if (anArray[i] === obj) {

anArray.splice(i, 1);

return true;

}

}

} cordova.js:5720

key: \_removeLayer

value: function (marker, removeFromDistanceGrid, dontUpdateMap) {

var gridClusters = this.\_gridClusters,

gridUnclustered = this.\_gridUnclustered,

fg = this.\_featureGroup,

map = this.\_map;

//Remove the marker from distance clusters it might be in

if (removeFromDistanceGrid) {

for (var z = this.\_maxZoom; z >= 0; z--) {

if (!gridUnclustered[z].removeObject(marker, map.project(marker.getLatLng(), z))) {

break;

}

}

}

//Work our way up the clusters removing them as we go if required

var cluster = marker.\_\_parent,

markers = cluster.\_markers,

otherMarker;

//Remove the marker from the immediate parents marker list

this.\_arraySplice(markers, marker);

while (cluster) {

cluster.\_childCount--;

if (cluster.\_zoom < 0) {

//Top level, do nothing

break;

} else if (removeFromDistanceGrid && cluster.\_childCount <= 1) { //Cluster no longer required

//We need to push the other marker up to the parent

otherMarker = cluster.\_markers[0] === marker ? cluster.\_markers[1] : cluster.\_markers[0];

//Update distance grid

gridClusters[cluster.\_zoom].removeObject(cluster, map.project(cluster.\_cLatLng, cluster.\_zoom));

gridUnclustered[cluster.\_zoom].addObject(otherMarker, map.project(otherMarker.getLatLng(), cluster.\_zoom));

//Move otherMarker up to parent

this.\_arraySplice(cluster.\_\_parent.\_childClusters, cluster);

cluster.\_\_parent.\_markers.push(otherMarker);

otherMarker.\_\_parent = cluster.\_\_parent;

if (cluster.\_icon) {

//Cluster is currently on the map, need to put the marker on the map instead

fg.removeLayer(cluster);

if (!dontUpdateMap) {

fg.addLayer(otherMarker);

}

}

} else {

cluster.\_recalculateBounds();

if (!dontUpdateMap || !cluster.\_icon) {

cluster.\_updateIcon();

}

}

cluster = cluster.\_\_parent;

}

delete marker.\_\_parent;

} cordova.js:5720

key: \_isOrIsParent

value: function (el, oel) {

while (oel) {

if (el === oel) {

return true;

}

oel = oel.parentNode;

}

return false;

} cordova.js:5720

key: \_propagateEvent

value: function (e) {

if (e.layer instanceof L.MarkerCluster) {

//Prevent multiple clustermouseover/off events if the icon is made up of stacked divs (Doesn't work in ie <= 8, no relatedTarget)

if (e.originalEvent && this.\_isOrIsParent(e.layer.\_icon, e.originalEvent.relatedTarget)) {

return;

}

e.type = 'cluster' + e.type;

}

this.fire(e.type, e);

} cordova.js:5720

key: \_defaultIconCreateFunction

value: function (cluster) {

var childCount = cluster.getChildCount();

var c = ' marker-cluster-';

if (childCount < 10) {

c += 'small';

} else if (childCount < 100) {

c += 'medium';

} else {

c += 'large';

}

return new L.DivIcon({ html: '<div><span>' + childCount + '</span></div>', className: 'marker-cluster' + c, iconSize: new L.Point(40, 40) });

} cordova.js:5720

key: \_bindEvents

value: function () {

var map = this.\_map,

spiderfyOnMaxZoom = this.options.spiderfyOnMaxZoom,

showCoverageOnHover = this.options.showCoverageOnHover,

zoomToBoundsOnClick = this.options.zoomToBoundsOnClick;

//Zoom on cluster click or spiderfy if we are at the lowest level

if (spiderfyOnMaxZoom || zoomToBoundsOnClick) {

this.on('clusterclick', this.\_zoomOrSpiderfy, this);

}

//Show convex hull (boundary) polygon on mouse over

if (showCoverageOnHover) {

this.on('clustermouseover', this.\_showCoverage, this);

this.on('clustermouseout', this.\_hideCoverage, this);

map.on('zoomend', this.\_hideCoverage, this);

}

} cordova.js:5720

key: \_zoomOrSpiderfy

value: function (e) {

var map = this.\_map;

if (map.getMaxZoom() === map.getZoom()) {

if (this.options.spiderfyOnMaxZoom) {

e.layer.spiderfy();

}

} else if (this.options.zoomToBoundsOnClick) {

e.layer.zoomToBounds();

}

// Focus the map again for keyboard users.

if (e.originalEvent && e.originalEvent.keyCode === 13) {

map.\_container.focus();

}

} cordova.js:5720

key: \_showCoverage

value: function (e) {

var map = this.\_map;

if (this.\_inZoomAnimation) {

return;

}

if (this.\_shownPolygon) {

map.removeLayer(this.\_shownPolygon);

}

if (e.layer.getChildCount() > 2 && e.layer !== this.\_spiderfied) {

this.\_shownPolygon = new L.Polygon(e.layer.getConvexHull(), this.options.polygonOptions);

map.addLayer(this.\_shownPolygon);

}

} cordova.js:5720

key: \_hideCoverage

value: function () {

if (this.\_shownPolygon) {

this.\_map.removeLayer(this.\_shownPolygon);

this.\_shownPolygon = null;

}

} cordova.js:5720

key: \_unbindEvents

value: function () {

var spiderfyOnMaxZoom = this.options.spiderfyOnMaxZoom,

showCoverageOnHover = this.options.showCoverageOnHover,

zoomToBoundsOnClick = this.options.zoomToBoundsOnClick,

map = this.\_map;

if (spiderfyOnMaxZoom || zoomToBoundsOnClick) {

this.off('clusterclick', this.\_zoomOrSpiderfy, this);

}

if (showCoverageOnHover) {

this.off('clustermouseover', this.\_showCoverage, this);

this.off('clustermouseout', this.\_hideCoverage, this);

map.off('zoomend', this.\_hideCoverage, this);

}

} cordova.js:5720

key: \_zoomEnd

value: function () {

if (!this.\_map) { //May have been removed from the map by a zoomEnd handler

return;

}

this.\_mergeSplitClusters();

this.\_zoom = this.\_map.\_zoom;

this.\_currentShownBounds = this.\_getExpandedVisibleBounds();

} cordova.js:5720

key: \_moveEnd

value: function () {

if (this.\_inZoomAnimation) {

return;

}

var newBounds = this.\_getExpandedVisibleBounds();

this.\_topClusterLevel.\_recursivelyRemoveChildrenFromMap(this.\_currentShownBounds, this.\_zoom, newBounds);

this.\_topClusterLevel.\_recursivelyAddChildrenToMap(null, this.\_map.\_zoom, newBounds);

this.\_currentShownBounds = newBounds;

return;

} cordova.js:5720

key: \_generateInitialClusters

value: function () {

var maxZoom = this.\_map.getMaxZoom(),

radius = this.options.maxClusterRadius,

radiusFn = radius;

//If we just set maxClusterRadius to a single number, we need to create

//a simple function to return that number. Otherwise, we just have to

//use the function we've passed in.

if (typeof radius !== "function") {

radiusFn = function () { return radius; };

}

if (this.options.disableClusteringAtZoom) {

maxZoom = this.options.disableClusteringAtZoom - 1;

}

this.\_maxZoom = maxZoom;

this.\_gridClusters = {};

this.\_gridUnclustered = {};

//Set up DistanceGrids for each zoom

for (var zoom = maxZoom; zoom >= 0; zoom--) {

this.\_gridClusters[zoom] = new L.DistanceGrid(radiusFn(zoom));

this.\_gridUnclustered[zoom] = new L.DistanceGrid(radiusFn(zoom));

}

this.\_topClusterLevel = new L.MarkerCluster(this, -1);

} cordova.js:5720

key: \_addLayer

value: function (layer, zoom) {

var gridClusters = this.\_gridClusters,

gridUnclustered = this.\_gridUnclustered,

markerPoint, z;

if (this.options.singleMarkerMode) {

layer.options.icon = this.options.iconCreateFunction({

getChildCount: function () {

return 1;

},

getAllChildMarkers: function () {

return [layer];

}

});

}

//Find the lowest zoom level to slot this one in

for (; zoom >= 0; zoom--) {

markerPoint = this.\_map.project(layer.getLatLng(), zoom); // calculate pixel position

//Try find a cluster close by

var closest = gridClusters[zoom].getNearObject(markerPoint);

if (closest) {

closest.\_addChild(layer);

layer.\_\_parent = closest;

return;

}

//Try find a marker close by to form a new cluster with

closest = gridUnclustered[zoom].getNearObject(markerPoint);

if (closest) {

var parent = closest.\_\_parent;

if (parent) {

this.\_removeLayer(closest, false);

}

//Create new cluster with these 2 in it

var newCluster = new L.MarkerCluster(this, zoom, closest, layer);

gridClusters[zoom].addObject(newCluster, this.\_map.project(newCluster.\_cLatLng, zoom));

closest.\_\_parent = newCluster;

layer.\_\_parent = newCluster;

//First create any new intermediate parent clusters that don't exist

var lastParent = newCluster;

for (z = zoom - 1; z > parent.\_zoom; z--) {

lastParent = new L.MarkerCluster(this, z, lastParent);

gridClusters[z].addObject(lastParent, this.\_map.project(closest.getLatLng(), z));

}

parent.\_addChild(lastParent);

//Remove closest from this zoom level and any above that it is in, replace with newCluster

for (z = zoom; z >= 0; z--) {

if (!gridUnclustered[z].removeObject(closest, this.\_map.project(closest.getLatLng(), z))) {

break;

}

}

return;

}

//Didn't manage to cluster in at this zoom, record us as a marker here and continue upwards

gridUnclustered[zoom].addObject(layer, markerPoint);

}

//Didn't get in anything, add us to the top

this.\_topClusterLevel.\_addChild(layer);

layer.\_\_parent = this.\_topClusterLevel;

return;

} cordova.js:5720

key: \_enqueue

value: function (fn) {

this.\_queue.push(fn);

if (!this.\_queueTimeout) {

this.\_queueTimeout = setTimeout(L.bind(this.\_processQueue, this), 300);

}

} cordova.js:5720

key: \_processQueue

value: function () {

for (var i = 0; i < this.\_queue.length; i++) {

this.\_queue[i].call(this);

}

this.\_queue.length = 0;

clearTimeout(this.\_queueTimeout);

this.\_queueTimeout = null;

} cordova.js:5720

key: \_mergeSplitClusters

value: function () {

//Incase we are starting to split before the animation finished

this.\_processQueue();

if (this.\_zoom < this.\_map.\_zoom && this.\_currentShownBounds.contains(this.\_getExpandedVisibleBounds())) { //Zoom in, split

this.\_animationStart();

//Remove clusters now off screen

this.\_topClusterLevel.\_recursivelyRemoveChildrenFromMap(this.\_currentShownBounds, this.\_zoom, this.\_getExpandedVisibleBounds());

this.\_animationZoomIn(this.\_zoom, this.\_map.\_zoom);

} else if (this.\_zoom > this.\_map.\_zoom) { //Zoom out, merge

this.\_animationStart();

this.\_animationZoomOut(this.\_zoom, this.\_map.\_zoom);

} else {

this.\_moveEnd();

}

} cordova.js:5720

key: \_getExpandedVisibleBounds

value: function () {

if (!this.options.removeOutsideVisibleBounds) {

return this.getBounds();

}

var map = this.\_map,

bounds = map.getBounds(),

sw = bounds.\_southWest,

ne = bounds.\_northEast,

latDiff = L.Browser.mobile ? 0 : Math.abs(sw.lat - ne.lat),

lngDiff = L.Browser.mobile ? 0 : Math.abs(sw.lng - ne.lng);

return new L.LatLngBounds(

new L.LatLng(sw.lat - latDiff, sw.lng - lngDiff, true),

new L.LatLng(ne.lat + latDiff, ne.lng + lngDiff, true));

} cordova.js:5720

key: \_animationAddLayerNonAnimated

value: function (layer, newCluster) {

if (newCluster === layer) {

this.\_featureGroup.addLayer(layer);

} else if (newCluster.\_childCount === 2) {

newCluster.\_addToMap();

var markers = newCluster.getAllChildMarkers();

this.\_featureGroup.removeLayer(markers[0]);

this.\_featureGroup.removeLayer(markers[1]);

} else {

newCluster.\_updateIcon();

}

} cordova.js:5720

key: \_initHooks

value: cordova.js:5720

key: callInitHooks

value: function (){if(!this.\_initHooksCalled){a.prototype.callInitHooks&&a.prototype.callInitHooks.call(this),this.\_initHooksCalled=!0;for(var t=0,e=n.\_initHooks.length;e>t;t++)n.\_initHooks[t].call(this)}} cordova.js:5720

key: \_animationStart

value: function () {

this.\_map.\_mapPane.className += ' leaflet-cluster-anim';

this.\_inZoomAnimation++;

} cordova.js:5720

key: \_animationEnd

value: function () {

if (this.\_map) {

this.\_map.\_mapPane.className = this.\_map.\_mapPane.className.replace(' leaflet-cluster-anim', '');

}

this.\_inZoomAnimation--;

this.fire('animationend');

} cordova.js:5720

key: \_animationZoomIn

value: function (previousZoomLevel, newZoomLevel) {

var bounds = this.\_getExpandedVisibleBounds(),

fg = this.\_featureGroup,

i;

//Add all children of current clusters to map and remove those clusters from map

this.\_topClusterLevel.\_recursively(bounds, previousZoomLevel, 0, function (c) {

var startPos = c.\_latlng,

markers = c.\_markers,

m;

if (!bounds.contains(startPos)) {

startPos = null;

}

if (c.\_isSingleParent() && previousZoomLevel + 1 === newZoomLevel) { //Immediately add the new child and remove us

fg.removeLayer(c);

c.\_recursivelyAddChildrenToMap(null, newZoomLevel, bounds);

} else {

//Fade out old cluster

c.setOpacity(0);

c.\_recursivelyAddChildrenToMap(startPos, newZoomLevel, bounds);

}

//Remove all markers that aren't visible any more

//TODO: Do we actually need to do this on the higher levels too?

for (i = markers.length - 1; i >= 0; i--) {

m = markers[i];

if (!bounds.contains(m.\_latlng)) {

fg.removeLayer(m);

}

}

});

this.\_forceLayout();

//Update opacities

this.\_topClusterLevel.\_recursivelyBecomeVisible(bounds, newZoomLevel);

//TODO Maybe? Update markers in \_recursivelyBecomeVisible

fg.eachLayer(function (n) {

if (!(n instanceof L.MarkerCluster) && n.\_icon) {

n.setOpacity(1);

}

});

//update the positions of the just added clusters/markers

this.\_topClusterLevel.\_recursively(bounds, previousZoomLevel, newZoomLevel, function (c) {

c.\_recursivelyRestoreChildPositions(newZoomLevel);

});

//Remove the old clusters and close the zoom animation

this.\_enqueue(function () {

//update the positions of the just added clusters/markers

this.\_topClusterLevel.\_recursively(bounds, previousZoomLevel, 0, function (c) {

fg.removeLayer(c);

c.setOpacity(1);

});

this.\_animationEnd();

});

} cordova.js:5720

key: \_animationZoomOut

value: function (previousZoomLevel, newZoomLevel) {

this.\_animationZoomOutSingle(this.\_topClusterLevel, previousZoomLevel - 1, newZoomLevel);

//Need to add markers for those that weren't on the map before but are now

this.\_topClusterLevel.\_recursivelyAddChildrenToMap(null, newZoomLevel, this.\_getExpandedVisibleBounds());

//Remove markers that were on the map before but won't be now

this.\_topClusterLevel.\_recursivelyRemoveChildrenFromMap(this.\_currentShownBounds, previousZoomLevel, this.\_getExpandedVisibleBounds());

} cordova.js:5720

key: \_animationZoomOutSingle

value: function (cluster, previousZoomLevel, newZoomLevel) {

var bounds = this.\_getExpandedVisibleBounds();

//Animate all of the markers in the clusters to move to their cluster center point

cluster.\_recursivelyAnimateChildrenInAndAddSelfToMap(bounds, previousZoomLevel + 1, newZoomLevel);

var me = this;

//Update the opacity (If we immediately set it they won't animate)

this.\_forceLayout();

cluster.\_recursivelyBecomeVisible(bounds, newZoomLevel);

//TODO: Maybe use the transition timing stuff to make this more reliable

//When the animations are done, tidy up

this.\_enqueue(function () {

//This cluster stopped being a cluster before the timeout fired

if (cluster.\_childCount === 1) {

var m = cluster.\_markers[0];

//If we were in a cluster animation at the time then the opacity and position of our child could be wrong now, so fix it

m.setLatLng(m.getLatLng());

if (m.setOpacity) {

m.setOpacity(1);

}

} else {

cluster.\_recursively(bounds, newZoomLevel, 0, function (c) {

c.\_recursivelyRemoveChildrenFromMap(bounds, previousZoomLevel + 1);

});

}

me.\_animationEnd();

});

} cordova.js:5720

key: \_animationAddLayer

value: function (layer, newCluster) {

var me = this,

fg = this.\_featureGroup;

fg.addLayer(layer);

if (newCluster !== layer) {

if (newCluster.\_childCount > 2) { //Was already a cluster

newCluster.\_updateIcon();

this.\_forceLayout();

this.\_animationStart();

layer.\_setPos(this.\_map.latLngToLayerPoint(newCluster.getLatLng()));

layer.setOpacity(0);

this.\_enqueue(function () {

fg.removeLayer(layer);

layer.setOpacity(1);

me.\_animationEnd();

});

} else { //Just became a cluster

this.\_forceLayout();

me.\_animationStart();

me.\_animationZoomOutSingle(newCluster, this.\_map.getMaxZoom(), this.\_map.getZoom());

}

}

} cordova.js:5720

key: \_forceLayout

value: function () {

//In my testing this works, infact offsetWidth of any element seems to work.

//Could loop all this.\_layers and do this for each \_icon if it stops working

L.Util.falseFn(document.body.offsetWidth);

} cordova.js:5720

key: \_spiderfied

value: null cordova.js:5720

key: \_spiderfierOnAdd

value: function () {

this.\_map.on('click', this.\_unspiderfyWrapper, this);

if (this.\_map.options.zoomAnimation) {

this.\_map.on('zoomstart', this.\_unspiderfyZoomStart, this);

}

//Browsers without zoomAnimation or a big zoom don't fire zoomstart

this.\_map.on('zoomend', this.\_noanimationUnspiderfy, this);

if (L.Path.SVG && !L.Browser.touch) {

this.\_map.\_initPathRoot();

//Needs to happen in the pageload, not after, or animations don't work in webkit

// http://stackoverflow.com/questions/8455200/svg-animate-with-dynamically-added-elements

//Disable on touch browsers as the animation messes up on a touch zoom and isn't very noticable

}

} cordova.js:5720

key: \_spiderfierOnRemove

value: function () {

this.\_map.off('click', this.\_unspiderfyWrapper, this);

this.\_map.off('zoomstart', this.\_unspiderfyZoomStart, this);

this.\_map.off('zoomanim', this.\_unspiderfyZoomAnim, this);

this.\_unspiderfy(); //Ensure that markers are back where they should be

} cordova.js:5720

key: \_unspiderfyZoomStart

value: function () {

if (!this.\_map) { //May have been removed from the map by a zoomEnd handler

return;

}

this.\_map.on('zoomanim', this.\_unspiderfyZoomAnim, this);

} cordova.js:5720

key: \_unspiderfyZoomAnim

value: function (zoomDetails) {

//Wait until the first zoomanim after the user has finished touch-zooming before running the animation

if (L.DomUtil.hasClass(this.\_map.\_mapPane, 'leaflet-touching')) {

return;

}

this.\_map.off('zoomanim', this.\_unspiderfyZoomAnim, this);

this.\_unspiderfy(zoomDetails);

} cordova.js:5720

key: \_unspiderfyWrapper

value: function () {

/// <summary>\_unspiderfy but passes no arguments</summary>

this.\_unspiderfy();

} cordova.js:5720

key: \_unspiderfy

value: function (zoomDetails) {

if (this.\_spiderfied) {

this.\_spiderfied.unspiderfy(zoomDetails);

}

} cordova.js:5720

key: \_noanimationUnspiderfy

value: function () {

if (this.\_spiderfied) {

this.\_spiderfied.\_noanimationUnspiderfy();

}

} cordova.js:5720

key: \_unspiderfyLayer

value: function (layer) {

if (layer.\_spiderLeg) {

this.\_featureGroup.removeLayer(layer);

layer.setOpacity(1);

//Position will be fixed up immediately in \_animationUnspiderfy

layer.setZIndexOffset(0);

this.\_map.removeLayer(layer.\_spiderLeg);

delete layer.\_spiderLeg;

}

} cordova.js:5720

key: addEventListener

value: function (t,e,i){if(s.Util.invokeEach(t,this.addEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p=this[a]=this[a]||{},d=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)r={action:e,context:i||this},l=t[n],d?(h=l+"\_idx",u=h+"\_len",c=p[h]=p[h]||{},c[d]||(c[d]=[],p[u]=(p[u]||0)+1),c[d].push(r)):(p[l]=p[l]||[],p[l].push(r));return this} cordova.js:5720

key: hasEventListeners

value: function (t){var e=this[a];return!!e&&(t in e&&e[t].length>0||t+"\_idx"in e&&e[t+"\_idx\_len"]>0)} cordova.js:5720

key: removeEventListener

value: function (t,e,i){if(!this[a])return this;if(!t)return this.clearAllEventListeners();if(s.Util.invokeEach(t,this.removeEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p,d,m=this[a],f=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)if(r=t[n],u=r+"\_idx",c=u+"\_len",p=m[u],e){if(l=f&&p?p[f]:m[r]){for(h=l.length-1;h>=0;h--)l[h].action!==e||i&&l[h].context!==i||(d=l.splice(h,1),d[0].action=s.Util.falseFn);i&&p&&0===l.length&&(delete p[f],m[c]--)}}else delete m[r],delete m[u],delete m[c];return this} cordova.js:5720

key: clearAllEventListeners

value: function (){return delete this[a],this} cordova.js:5720

key: fireEvent

value: function (t,e){if(!this.hasEventListeners(t))return this;var i,n,o,r,l,h=s.Util.extend({},e,{type:t,target:this}),u=this[a];if(u[t])for(i=u[t].slice(),n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);r=u[t+"\_idx"];for(l in r)if(i=r[l].slice())for(n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);return this} cordova.js:5720

key: addOneTimeEventListener

value: function (t,e,i){if(s.Util.invokeEach(t,this.addOneTimeEventListener,this,e,i))return this;var n=s.bind(function(){this.removeEventListener(t,e,i).removeEventListener(t,n,i)},this);return this.addEventListener(t,e,i).addEventListener(t,n,i)} cordova.js:5720

key: on

value: function (t,e,i){if(s.Util.invokeEach(t,this.addEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p=this[a]=this[a]||{},d=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)r={action:e,context:i||this},l=t[n],d?(h=l+"\_idx",u=h+"\_len",c=p[h]=p[h]||{},c[d]||(c[d]=[],p[u]=(p[u]||0)+1),c[d].push(r)):(p[l]=p[l]||[],p[l].push(r));return this} cordova.js:5720

key: off

value: function (t,e,i){if(!this[a])return this;if(!t)return this.clearAllEventListeners();if(s.Util.invokeEach(t,this.removeEventListener,this,e,i))return this;var n,o,r,l,h,u,c,p,d,m=this[a],f=i&&i!==this&&s.stamp(i);for(t=s.Util.splitWords(t),n=0,o=t.length;o>n;n++)if(r=t[n],u=r+"\_idx",c=u+"\_len",p=m[u],e){if(l=f&&p?p[f]:m[r]){for(h=l.length-1;h>=0;h--)l[h].action!==e||i&&l[h].context!==i||(d=l.splice(h,1),d[0].action=s.Util.falseFn);i&&p&&0===l.length&&(delete p[f],m[c]--)}}else delete m[r],delete m[u],delete m[c];return this} cordova.js:5720

key: once

value: function (t,e,i){if(s.Util.invokeEach(t,this.addOneTimeEventListener,this,e,i))return this;var n=s.bind(function(){this.removeEventListener(t,e,i).removeEventListener(t,n,i)},this);return this.addEventListener(t,e,i).addEventListener(t,n,i)} cordova.js:5720

key: fire

value: function (t,e){if(!this.hasEventListeners(t))return this;var i,n,o,r,l,h=s.Util.extend({},e,{type:t,target:this}),u=this[a];if(u[t])for(i=u[t].slice(),n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);r=u[t+"\_idx"];for(l in r)if(i=r[l].slice())for(n=0,o=i.length;o>n;n++)i[n].action.call(i[n].context,h);return this} cordova.js:5720

key: bindPopup

value: function (t,e){return this.\_popupContent=t,this.\_popupOptions=e,this.invoke("bindPopup",t,e)} cordova.js:5720

key: openPopup

value: function (t){for(var e in this.\_layers){this.\_layers[e].openPopup(t);break}return this} cordova.js:5720

key: setStyle

value: function (t){return this.invoke("setStyle",t)} cordova.js:5720

key: bringToFront

value: function (){return this.invoke("bringToFront")} cordova.js:5720

key: bringToBack

value: function (){return this.invoke("bringToBack")} cordova.js:5720

key: invoke

value: function (t){var e,i,n=Array.prototype.slice.call(arguments,1);for(e in this.\_layers)i=this.\_layers[e],i[t]&&i[t].apply(i,n);return this} cordova.js:5720

key: addTo

value: function (t){return t.addLayer(this),this} cordova.js:5720

key: setZIndex

value: function (t){return this.invoke("setZIndex",t)} cordova.js:5720

key: getLayerId

value: function (t){return s.stamp(t)} cordova.js:5720

key: toGeoJSON

value: function (){var e,i=this.feature&&this.feature.geometry,n=[];if(i&&"MultiPoint"===i.type)return t("MultiPoint").call(this);var o=i&&"GeometryCollection"===i.type;return this.eachLayer(function(t){t.toGeoJSON&&(e=t.toGeoJSON(),n.push(o?e.geometry:s.GeoJSON.asFeature(e)))}),o?s.GeoJSON.getFeature(this,{geometries:n,type:"GeometryCollection"}):{type:"FeatureCollection",features:n}} cordova.js:5720

Marker Clicked cordova.js:5720

LatLng(53.34438, -6.26962) cordova.js:5720