Products Pricing Documentation

npm

Sign Up Sign In

Search packages Search

leaflet.utm DT

1.0.0 • Public • Published 2 years ago

Readme

Explore BETA

0 Dependencies

1 Dependents

6 Versions

Leaflet.UTM

Build Status npm v1.0.0 license BSD-3-Clause Leaflet 0.7 ✓ Leaflet 1.x ✓

Simple UTM (WGS84) methods for L.LatLng. Tested with Leaflet 0.7, 1.0.3, 1.1.0, 1.2.0, 1.3.1 and 1.6.0.

Based on javascript code from Chuck Taylor's Toolbox.

Simple example in action

Installation

Using npm for browserify npm install leaflet.utm (and require('leaflet.utm')), or just download L.LatLng.UTM.js and add a script tag for it in your html.

Usage

LatLng -> UTM

Call the method `utm()` in any `L.LatLng` object to get an UTM coordinates object. The method `toString` will convert it seamlessly to string. For instance, to create a popup in a marker:

```
marker.bindPopup('UTM: ' + marker.getLatLng().utm());
```

with the text `UTM: 467486.3, 4101149.3, 30S, WGS84`

You can use a personalized format like here:

```
var txt = map.getCenter().utm().toString({decimals: 0, format: '{x}
```

that produces `467486 4101149 30 North`

You can also use the values of that `object` directly (like `c.utm().x`). Here is a dump in the Console:

```
L.Utm {x: 467486.3402722592,
       y: 4101149.337496558,
       zone: 30,
       band: "S",
       southHemi: false}
```

UTM -> LatLng

Just create an object with `L.utm(options)`, and call the method `latLng` like here:

```
var item = L.utm({x: 467486.3, y: 4101149.3, zone: 30, band: 'S'});
var coord = item.latLng();
```

You can also specify the hemisphere if you don't know the band, with `southHemi` attribute: `{x: 467486, y: 4101149, zone: 30, southHemi: false}`

API

`L.LatLng.utm`

Extends the class `L.LatLng` with the method `utm([zone, [southHemi]])`. If zone is not provided, or 0, it is computed based on latitude and longitude (recommended). If southHemi is not provided or null, it is computed based on latitude. This method returns an object of class `L.Utm`.

`L.Utm`

Defines a class to deal with UTM coordinates. The available methods are:

`toString([options])`

Converts the UTM coordinates into a string. The available options are:

- decimals: number of decimals for x and y. Default 1.
- format: string defining the format to use. Default `'{x}{sep} {y}{sep} {zone}{band}{sep} {datum}'`, where:
  - `{x}` : easting
  - `{y}` : northing
  - `{zone}` : UTM zone, value between 1 and 60
  - `{band}` : Band letter, between C and X
  - `{datum}` : WGS84
  - `{hemi}` : Hemisphere, north or south (see options below)
  - `{sep}` : separator
- sep: separator used in the format. Default ','
- north: string used in the format for field `{hemi}` in the north hemisphere. Default 'North'.
- south: string used in the format for field `{hemi}` in the south hemisphere. Default 'South'

This method is automatically used by javascript when need to convert to string, for instance, when adding to another string.

`latLng()`

Creates an L.LatLng object, converting the UTM coordinates. If both `band` and `southHemi` attributes are defined, `band` has priority to determine the hemisphere, and therefore the latitude. If neither `band` nor `southHemi` are defined, an exception is thrown. Returns a `null` object if converted latitude is out of [-90, 90].

`normalize()`

Returns a new object, with the values on the proper zone, band, etc. Internally it converts to latLng and then to utm. Returns `null` if converted latitude is out of [-90, 90], or conversion is not possible.

`equals(other)`

Compares the object with `other`. It compares the lat and lng values, not the utm parameters (that means that both represent the same point in Earth). Returns false if any conversion is failing.

`clone()`

Creates a copy of itself.

Factory `L.utm(...)`

Creates an utm object. Accepts an object with attribures: x, y, zone, band, southHemi. This method does not check that input values make sense. You can set values far away from the proper zone, or wrong band. This may be obviously a problem when you call `latLng()` method. Values out of the zone but near work perfectly. Use the method `normalize()` to normalize it, or simply to check the input.

`L.Utm.setDefaultOptions(opts)`

This function changes the default options for the `toString` method. It is a global function (see that you must call it with the capital U), to modify globally the options for any object created. You can call it with an object or a function. To return to the 'factory values', just call it with `null`. Use it at the begining of your code if you prefer using another format for the UTM representation.

Running tests

Install dependencies and run tests:

```
npm install && npm test
```

or load `test/index.html` in your browser after installing the dependencies by running `npm install`.

Keywords

Leafleft map geo UTM

Install

```
> npm i leaflet.utm
```

Repository

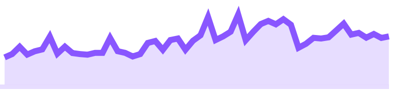
github.com/jjimenezshaw/Leaflet.UTM

Homepage

github.com/jjimenezshaw/Leaflet.UTM#readme

Weekly Downloads

449



Version	License
1.0.0	BSD-3-Clause
Unpacked Size	Total Files
48.4 kB	13
Issues	Pull Requests
1	3

Last publish

2 years ago

Collaborators

>_Try on RunKit

Report malware

Support

Help

Advisories

Status

Contact npm

Company

About

Blog

Press

Terms & Policies

Policies

Terms of Use

Code of Conduct

Privacy