| **Option** | **Type** | **Description** | **Default** |
| --- | --- | --- | --- |
| childRow | Function | [See documentation](https://www.npmjs.com/package/vue-tables-2#child-rows) | false |
| childRowTogglerFirst | Boolean | Should the child row be positioned at the first column or the last one | true |
| clientMultiSorting | Boolean | Enable multiple columns sorting using Shift + Click on the client component | true |
| toggleGroups (client-side) | Boolean | When using the groupBy option, settings this to true will make group's visibility togglable, by turning the group name into a button | false |
| columnsClasses | Object | Add class(es) to the specified columns. Takes key-value pairs, where the key is the column name and the value is a string of space-separated classes | {} |
| columnsDisplay | Object | Responsive display for the specified columns.  Columns will only be shown when the window width is within the defined limits.   Accepts key-value pairs of column name and device.  Possible values are mobile (x < 480), mobileP (x < 320), mobileL (320 <= x < 480), tablet (480 <= x < 1024), tabletP (480 <= x < 768), tabletL (768 <= x < 1024), desktop (x >= 1024).  All options can be preceded by the logical operators min,max, and not followed by an underscore.  For example, a column which is set to not\_mobile will be shown when the width of the window is greater than or equal to 480px, while a column set to max\_tabletP will only be shown when the width is under 768px | {} |
| columnsDropdown | Boolean | See [documentation](https://www.npmjs.com/package/vue-tables-2#columns-visibility) | false |
| customFilters | Array | See [documentation](https://www.npmjs.com/package/vue-tables-2#custom-filters) | [] |
| customSorting (client-side) | Object | See [documentation](https://www.npmjs.com/package/vue-tables-2#custom-sorting) | {} |
| dateColumns | Array | Use daterangepicker as a filter for the specified columns (when filterByColumn is set to true).  Dates should be passed as moment objects, or as strings in conjunction with the toMomentFormat option | [] |
| dateFormat | String | Format to display date objects (client component). Using [momentjs](https://momentjs.com/). This will also affect the datepicker on both components | DD/MM/YYYY |
| dateFormatPerColumn | Object | override the default date format for specific columns | {} |
| datepickerOptions | Object | Options for the daterangepicker when using a date filter (see dateColumns) | { locale: { cancelLabel: 'Clear' } } |
| datepickerPerColumnOptions | Object | additional options for specific columns, to be merged with datepickerOptions. Expects an object where the key is a column name, and the value is an options object | {} |
| debounce | Number | Number of idle milliseconds (no key stroke) to wait before sending a request. Used to detect when the user finished his query in order to avoid redundant requests (server) or rendering (client) | 500 |
| descOrderColumns | Array | By default the initial sort direction is ascending. To override this for specific columns pass them into this option | [] |
| destroyEventBus | Boolean | Should the plugin destroy the event bus before the table component is destroyed? Since the bus is shared by all instances, only set this to true if you have a single instance in your page | false |
| filterable | Array / Boolean | Filterable columns true - All columns. | Set to false or [] to hide the filter(s). Single filter mode (filterByColumn:false) is also affected |
| footerHeadings | Boolean | Display headings at the bottom of the table too | false |
| headings | Object | Table headings. | Can be either a string or a function, if you wish to inject vue-compiled HTML. E.g: function(h) { return <h2>Title</h2>} Note that this example uses jsx, and not HTML. The this context inside the function refers to the direct parent of the table instance. Another option is to use a slot, named "h\_\_{column}" If no heading is provided the default rule is to extract it from the first row properties, where underscores become spaces and the first letter is capitalized |
| groupBy (client-side) | String | Group rows by a common property. E.g, for a list of countries, group by the continent property | false |
| groupMeta (client-side) | Array | Meta data associated with each group value. To be used in conjunction with groupBy. See documentation for a full example | [] |
| headingsTooltips | Object | Table headings tooltips. | Can be either a string or a function, if you wish to inject vue-compiled HTML. Renders as title attribute of <th>.  E.g: function(h) { return 'Expanded Title'} The this context inside the function refers to the direct parent of the table instance. |
| highlightMatches | Boolean | Highlight matches | false |
| initFilters | Object | Set initial values for all filter types: generic, by column or custom.  Accepts an object of key-value pairs, where the key is one of the following:   a. "GENERIC" - for the generic filter b. column name - for by column filters. c. filter name - for custom filters.   In case of date filters the date range should be passed as an object comprised of start and end properties, each being a moment object. | {} |
| initialPage | Number | Set the initial page to be displayed when the table loads | 1 |
| listColumns | Object | See [documentation](https://www.npmjs.com/package/vue-tables-2#list-filters) | {} |
| multiSorting (client-side) | Object | See [documentation](https://www.npmjs.com/package/vue-tables-2#multiple-sotring) | {} |
| orderBy.ascending | Boolean | initial order direction | orderBy: { ascending:true } |
| orderBy.column | String | initial column to sort by | Original dataset order |
| pagination.chunk | Number | maximum pages in a chunk of pagination | pagination: { chunk:10 } |
| pagination.dropdown | Boolean | use a dropdown select pagination next to the records-per-page list, instead of links at the bottom of the table. | pagination: { dropdown:false } |
| pagination.nav | String | Which method to use when navigating outside of chunk boundries. Options are : scroll - the range of pages presented will incrementally change when navigating to a page outside the chunk (e.g 1-10 will become 2-11 once the user presses the next arrow to move to page 11). fixed - navigation will occur between fixed chunks (e.g 1-10, 11-20, 21-30 etc.). Double arrows will be added to allow navigation to the beginning of the previous or next chunk | pagination: { nav: 'fixed' } |
| pagination.edge | Boolean | Show 'First' and 'Last' buttons | pagination: { edge: false } |
| params (server-side) | Object | Additional parameters to send along with the request | {} |
| perPage | number | Initial records per page | 10 |
| perPageValues | Array | Records per page options | [10,25,50,100] |
| preserveState | Boolean | Preserve dynamically created vuex module when the table is destroyed | false |
| requestAdapter (server-side) | Function | Set a custom request format | function(data) { return data; } |
| requestFunction (server-side) | Function | Set a custom request function | See documentation |
| requestKeys (server-side) | Object | Set your own request keys | { query:'query', limit:'limit', orderBy:'orderBy', ascending:'ascending', page:'page', byColumn:'byColumn' } |
| responseAdapter (server-side) | Function | Transform the server response to match the format expected by the client. This is especially useful when calling a foreign API, where you cannot control the response on the server-side | function(resp) { var data = this.getResponseData(resp); return { data: data.data, count: data.count } } |
| rowClassCallback | Function | Add dynamic classes to table rows.  E.g function(row) { return row-${row.id}}   This can be useful for manipulating the appearance of rows based on the data they contain | false |
| saveState | Boolean | Constantly save table state and reload it each time the component mounts. When setting it to true, use the name prop to set an identifier for the table | false |
| sendEmptyFilters (server-side) | Boolean | When filtering by column send all request keys, including empty ones | false |
| serverMultiSorting | Boolean | Enable multiple columns sorting using Shift + Click on the server component | false |
| skin | String | Space separated table styling classes | table-striped table-bordered table-hover |
| sortIcon | String | Sort icon classes | { base:'glyphicon', up:'glyphicon-chevron-up', down:'glyphicon-chevron-down', is:'glyphicon-sort' } |
| sortable | Array | Sortable columns | All columns |
| sortingAlgorithm | Function | define your own sorting algorithm | function (data, column) { return data.sort(this.getSortFn(column));} |
| storage | String | Which persistance mechanism should be used when saveState is set to true: local - localStorage. session - sessionStorage | local |
| templates | Object | See [documentation](https://www.npmjs.com/package/vue-tables-2#templates) | {} |
| texts | Object | see the texts object in [defaults.js](https://github.com/matfish2/vue-tables-2/blob/master/lib/config/defaults.js) |  |
| toMomentFormat (client-side) | String | transform date columns string values to momentjs objects using this format. If this option is not used the consumer is expected to pass momentjs objects himself | false |
| uniqueKey | String | The key of a unique identifier in your dataset, used to track the child rows, and return the original row in row click event |  |

DEFAULT.JS:

|  |
| --- |
| module.exports = function() { |
|  | return { |
|  | dateColumns:[], |
|  | listColumns:{}, |
|  | datepickerOptions:{ |
|  | locale: { |
|  | cancelLabel:'Clear' |
|  | } |
|  | }, |
|  | datepickerPerColumnOptions:{}, |
|  | initialPage:1, |
|  | perPage:10, |
|  | perPageValues:[10,25,50,100], |
|  | groupBy:false, |
|  | collapseGroups:false, |
|  | destroyEventBus:false, |
|  | sendEmptyFilters:false, |
|  | params:{}, |
|  | sortable:true, |
|  | filterable:true, |
|  | groupMeta:[], |
|  | initFilters:{}, |
|  | customFilters:[], |
|  | templates:{}, |
|  | debounce:250, |
|  | dateFormat:"DD/MM/YYYY", |
|  | dateFormatPerColumn:{}, |
|  | toMomentFormat:false, |
|  | skin:false, |
|  | columnsDisplay: {}, |
|  | columnsDropdown: false, |
|  | texts:{ |
|  | count:"Showing {from} to {to} of {count} records|{count} records|One record", |
|  | first:'First', |
|  | last:'Last', |
|  | filter:"Filter:", |
|  | filterPlaceholder:"Search query", |
|  | limit:"Records:", |
|  | page:"Page:", |
|  | noResults:"No matching records", |
|  | filterBy:"Filter by {column}", |
|  | loading:'Loading...', |
|  | defaultOption:'Select {column}', |
|  | columns:'Columns' |
|  | }, |
|  | sortIcon:{ |
|  | is:'glyphicon-sort', |
|  | base:'glyphicon', |
|  | up: 'glyphicon-chevron-up', |
|  | down: 'glyphicon-chevron-down' |
|  | }, |
|  | sortingAlgorithm(data, column) { |
|  | return data.sort(this.getSortFn(column)); |
|  | }, |
|  | customSorting:{}, |
|  | multiSorting:{}, |
|  | clientMultiSorting:true, |
|  | serverMultiSorting:false, |
|  | filterByColumn:false, |
|  | highlightMatches:false, |
|  | orderBy:false, |
|  | descOrderColumns:[], |
|  | footerHeadings:false, |
|  | headings:{}, |
|  | headingsTooltips:{}, |
|  | pagination: { |
|  | dropdown:false, |
|  | chunk:10, |
|  | edge:false, |
|  | align:'center', |
|  | nav:'fixed' |
|  | }, |
|  | childRow: false, |
|  | childRowTogglerFirst:true, |
|  | uniqueKey:'id', |
|  | requestFunction:false, |
|  | requestAdapter: function(data) { |
|  | return data; |
|  | }, |
|  | responseAdapter: function(resp) { |
|  |  |
|  | var data = this.getResponseData(resp); |
|  |  |
|  | return { |
|  | data: data.data, |
|  | count: data.count |
|  | } |
|  | }, |
|  | requestKeys:{ |
|  | query:'query', |
|  | limit:'limit', |
|  | orderBy:'orderBy', |
|  | ascending:'ascending', |
|  | page:'page', |
|  | byColumn:'byColumn' |
|  | }, |
|  | rowClassCallback: false, |
|  | preserveState: false, |
|  | saveState: false, |
|  | storage:'local', |
|  | columnsClasses:{} |
|  | } |
|  | } |