* [Date Range Picker](http://docs.google.com/)
* [Improvely](https://www.improvely.com)
* [W3Counter](https://www.w3counter.com)
* [Website Goodies](https://www.websitegoodies.com)

Date Range Picker

A JavaScript component for choosing date ranges, dates and times.

[View on GitHub](https://github.com/dangrossman/daterangepicker)   [Download ZIP](https://github.com/dangrossman/daterangepicker/archive/master.zip)

[Getting Started](#gjdgxs) [Examples](#30j0zll) [Options, Methods & Events](#1fob9te) [Configuration Generator](#3znysh7) [License & Comments](#2et92p0)

Originally created for reports at [Improvely](https://www.improvely.com), the Date Range Picker can be attached to any webpage element to pop up two calendars for selecting dates, times, or predefined ranges like "Last 30 Days".

# [**Getting Started**](#gjdgxs)

To get started, include jQuery, Moment.js and Date Range Picker's files in your webpage:

Then attach a date range picker to whatever you want to trigger it:

Code:

Produces:

You can customize Date Range Picker with options, and get notified when the user chooses new dates by providing a callback function.

# [**Examples**](#30j0zll)

## [**Simple Date Range Picker With a Callback**](#tyjcwt)

Code:

Produces:

## [**Date Range Picker With Times**](#3dy6vkm)

Code:

Produces:

## [**Single Date Picker**](#1t3h5sf)

Code:

Produces:

## [**Predefined Date Ranges**](#4d34og8)

Code:

Produces:

## [**Input Initially Empty**](#2s8eyo1)

Code:

Produces:

# [**Options**](#1fob9te)

* startDate (Date or string) The beginning date of the initially selected date range. If you provide a string, it must match the date format string set in your locale setting.
* endDate: (Date or string) The end date of the initially selected date range.
* minDate: (Date or string) The earliest date a user may select.
* maxDate: (Date or string) The latest date a user may select.
* maxSpan: (object) The maximum span between the selected start and end dates. Check off maxSpan in the configuration generator for an example of how to use this. You can provide any object the moment library would let you add to a date.
* showDropdowns: (true/false) Show year and month select boxes above calendars to jump to a specific month and year.
* minYear: (number) The minimum year shown in the dropdowns when showDropdowns is set to true.
* maxYear: (number) The maximum year shown in the dropdowns when showDropdowns is set to true.
* showWeekNumbers: (true/false) Show localized week numbers at the start of each week on the calendars.
* showISOWeekNumbers: (true/false) Show ISO week numbers at the start of each week on the calendars.
* timePicker: (true/false) Adds select boxes to choose times in addition to dates.
* timePickerIncrement: (number) Increment of the minutes selection list for times (i.e. 30 to allow only selection of times ending in 0 or 30).
* timePicker24Hour: (true/false) Use 24-hour instead of 12-hour times, removing the AM/PM selection.
* timePickerSeconds: (true/false) Show seconds in the timePicker.
* ranges: (object) Set predefined date ranges the user can select from. Each key is the label for the range, and its value an array with two dates representing the bounds of the range. Click ranges in the configuration generator for examples.
* showCustomRangeLabel: (true/false) Displays "Custom Range" at the end of the list of predefined ranges, when the ranges option is used. This option will be highlighted whenever the current date range selection does not match one of the predefined ranges. Clicking it will display the calendars to select a new range.
* alwaysShowCalendars: (true/false) Normally, if you use the ranges option to specify pre-defined date ranges, calendars for choosing a custom date range are not shown until the user clicks "Custom Range". When this option is set to true, the calendars for choosing a custom date range are always shown instead.
* opens: ('left'/'right'/'center') Whether the picker appears aligned to the left, to the right, or centered under the HTML element it's attached to.
* drops: ('down'/'up') Whether the picker appears below (default) or above the HTML element it's attached to.
* buttonClasses: (string) CSS class names that will be added to both the apply and cancel buttons.
* applyButtonClasses: (string) CSS class names that will be added only to the apply button.
* cancelButtonClasses: (string) CSS class names that will be added only to the cancel button.
* locale: (object) Allows you to provide localized strings for buttons and labels, customize the date format, and change the first day of week for the calendars. Check off locale in the configuration generator to see how to customize these options.
* singleDatePicker: (true/false) Show only a single calendar to choose one date, instead of a range picker with two calendars. The start and end dates provided to your callback will be the same single date chosen.
* autoApply: (true/false) Hide the apply and cancel buttons, and automatically apply a new date range as soon as two dates are clicked.
* linkedCalendars: (true/false) When enabled, the two calendars displayed will always be for two sequential months (i.e. January and February), and both will be advanced when clicking the left or right arrows above the calendars. When disabled, the two calendars can be individually advanced and display any month/year.
* isInvalidDate: (function) A function that is passed each date in the two calendars before they are displayed, and may return true or false to indicate whether that date should be available for selection or not.
* isCustomDate: (function) A function that is passed each date in the two calendars before they are displayed, and may return a string or array of CSS class names to apply to that date's calendar cell.
* autoUpdateInput: (true/false) Indicates whether the date range picker should automatically update the value of the <input> element it's attached to at initialization and when the selected dates change.
* parentEl: (string) jQuery selector of the parent element that the date range picker will be added to, if not provided this will be 'body'

# [**Methods**](#17dp8vu)

You can programmatically update the startDate and endDate in the picker using the setStartDate and setEndDate methods. You can access the Date Range Picker object and its functions and properties through data properties of the element you attached it to.

* setStartDate(Date or string): Sets the date range picker's currently selected start date to the provided date
* setEndDate(Date or string): Sets the date range picker's currently selected end date to the provided date

**Example usage:**

# [**Events**](#3rdcrjn)

Several events are triggered on the element you attach the picker to, which you can listen for.

* show.daterangepicker: Triggered when the picker is shown
* hide.daterangepicker: Triggered when the picker is hidden
* showCalendar.daterangepicker: Triggered when the calendar(s) are shown
* hideCalendar.daterangepicker: Triggered when the calendar(s) are hidden
* apply.daterangepicker: Triggered when the apply button is clicked, or when a predefined range is clicked
* cancel.daterangepicker: Triggered when the cancel button is clicked

Some applications need a "clear" instead of a "cancel" functionality, which can be achieved by changing the button label and watching for the cancel event:

While passing in a callback to the constructor is the easiest way to listen for changes in the selected date range, you can also do something every time the apply button is clicked even if the selection hasn't changed:

# [**Configuration Generator**](#3znysh7)

parentEl

startDate

endDate

minDate

maxDate

opens right left center

drops down up

showDropdowns

minYear

maxYear

showWeekNumbers

showISOWeekNumbers

singleDatePicker

timePicker

timePicker24Hour

timePickerIncrement (in minutes)

timePickerSeconds

maxSpan (example value)

locale (example settings)

autoApply

buttonClasses

applyButtonClasses

cancelButtonClasses

ranges (with example predefined ranges)

alwaysShowCalendars

showCustomRangeLabel

linkedCalendars

autoUpdateInput

Your Date Range Picker

Your Configuration to Copy

# [**License**](#2et92p0)

The MIT License (MIT)

Copyright (c) 2012-2019 [Dan Grossman](http://www.dangrossman.info)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

# [**Comments**](#26in1rg)

Please enable JavaScript to view the [comments powered by Disqus.](https://disqus.com/?ref_noscript)

Copyright © 2012-2018 [Dan Grossman](http://www.dangrossman.info/).