CalendarPicker

CalendarPicker is an extremely versatile and easy to implement Sencha Touch custom component that utilizes **TouchTreeGrid** to provide incredible flexibility for date selection and calendar display on Phones and Tablets. Calendars can be implemented with just a few configurations in overlay panels or traditional containers. Calendar is constructed using #months backward to #months forward configurations. Many features exist including: collapsible months for rapid single select, range select in single calendar popup, or multi-select random dates in single popup. Specific dates can be disabled from selection. Customizable styling exists for holidays, weekends, selected days, disabled days and custom days. Custom filters can be easily applied to only display specific months highlighting important upcoming dates. Day planner support is partially implemented and will be forthcoming. Provided examples work for Touch 2.2, 2.3-beta and for IE10/Windows phones.

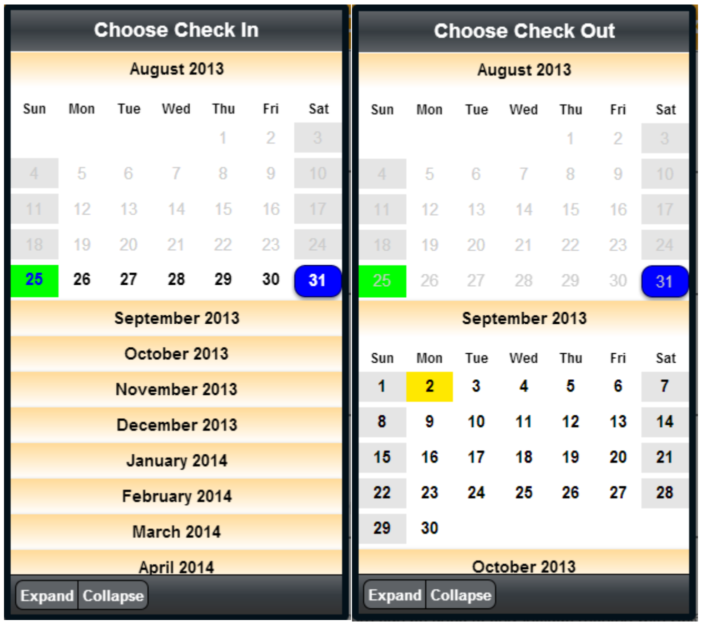
Refer to "**Features**" and "**Notes on Implementation**" sections for minimum configurations to get started using **CalendarPicker** along with explanation of provided examples. Basic flow is that developer will create a button to launch the calendar. The calendar will fire a 'closedCalendar' event which the developer will listen for via controller method to process the selection and destroy the calendar.

TouchTreeGrid and **CalendarPicker** were developed entirely within Sencha Architect (v2.2) designer product and can also be used without Architect. Architect components are provided for import into your toolbox (TouchTreeGrid.xdc and CalendarPicker.xdc). This software can be downloaded at https://github.com/swluken/TouchTreeGrid and is free to use (refer to associated MIT.LISCENSE) .

Links to live demo contained within READ.ME file on the github site.

Contents

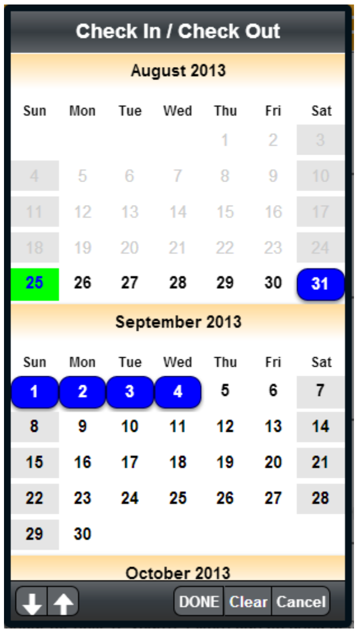
Kayak-style Check-in/Check-out date selection



- Similar to Kayak calendar popup on Android and IPhone apps (where next 12 months are displayed for selection) with enhancement that collapsible months are supported for rapid navigation.
- Dates prior to today are disabled from selection for Check-In date
- Dates prior to Check-In date are disabled for Check-Out date
- Upon navigation back, calendar auto-scrolls to Check-In Month.

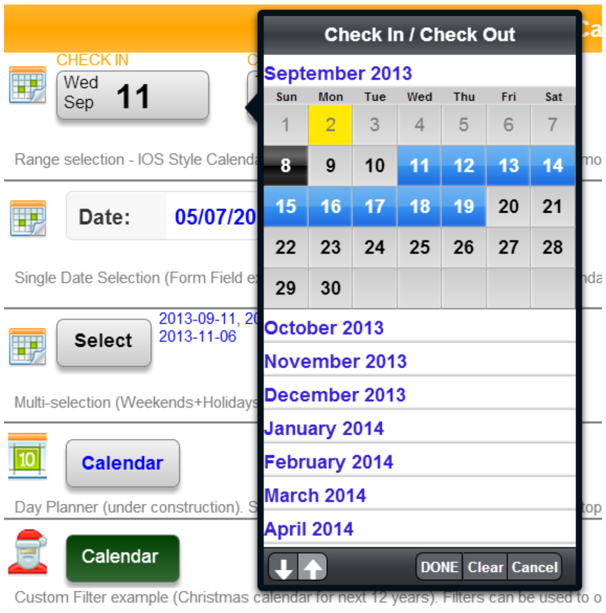
CalendarPicker

Range Selection in Single Popup



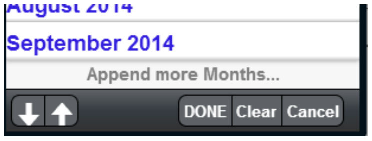
- Same as Kayak Check-In/Check-Out example except range of dates selectable within single popup.
- "DONE" accepts the selected dates and closes the calendar
- "Clear" clears the calendar selections and awaits further selections
- "Cancel" reverts any changes made to the calendar and closes it

IOS-Style Calendar Example

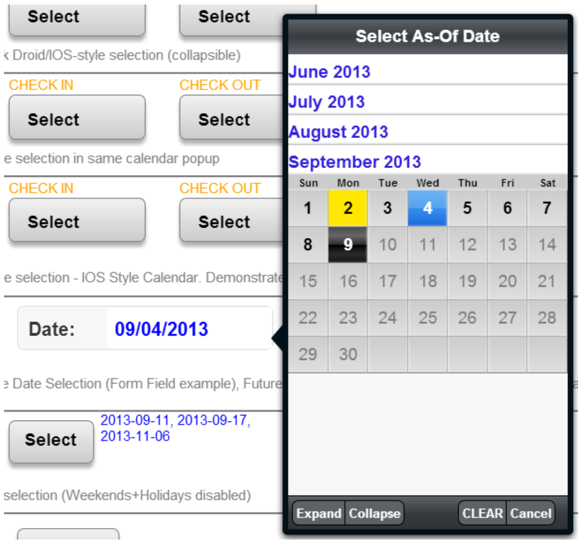

CHECK IN
Wed Sep 11
Range selection - IOS Style Calend
Date: 05/07/20
Single Date Selection (Form Field e
Select
2013-09-11, 20
2013-11-06
Multi-selection (Weekends+Holidays
Calendar
Day Planner (under construction). S
Calendar
Custom Filter example (Christmas calendar for next 12 years). Filters can be used to o
Check In / Check Out
September 2013
October 2013
November 2013
December 2013
January 2014
February 2014
March 2014
April 2014
DONE
Clear
Cancel

CalendarPicker

- I-Phone style calendar for range selection in single popup.
- Background transparency changed to white instead of default gray behind modal panel.
- Includes option at end of scrolled list to allow user to append more months to calendar (defaults to 3 months at a time, but configurable)

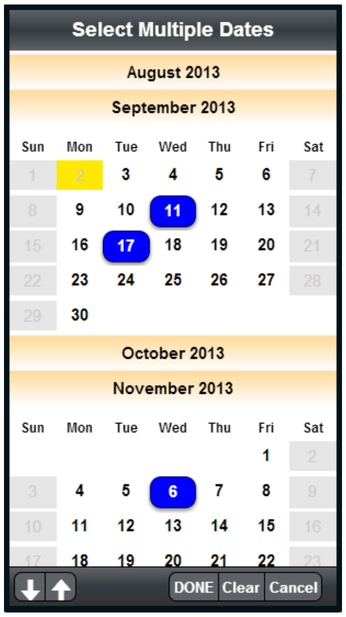


Single Popup selection Example from TextField


Select
Select
Droid/IOS-style selection (collapsible)
CHECK IN
CHECK OUT
Select
Select
e selection in same calendar popup
CHECK IN
CHECK OUT
Select
Select
e selection - IOS Style Calendar. Demonstrate
Date:
09/04/2013
Date Selection (Form Field example), Future
Select
2013-09-11, 2013-09-17, 2013-11-06
selection (Weekends+Holidays disabled)
Select As-Of Date
June 2013
July 2013
August 2013
September 2013
Sun Mon Tue Wed Thu Fri Sat
Expand
Collapse
CLEAR
Cancel

- I-Phone style single date selection example
- Tap listener added to read-only textfield which fires custom ‘sel_date’ event processed by controller method to launch calendar.
- “Pull to Insert Months…” feature implemented to insert months at beginning of Calendar.
- Custom toolbar buttons Clear and Cancel added to single-select mode to support ability to clear a date or revert any updates to calendar with existing date selection.

Multi-select Example



September 2013

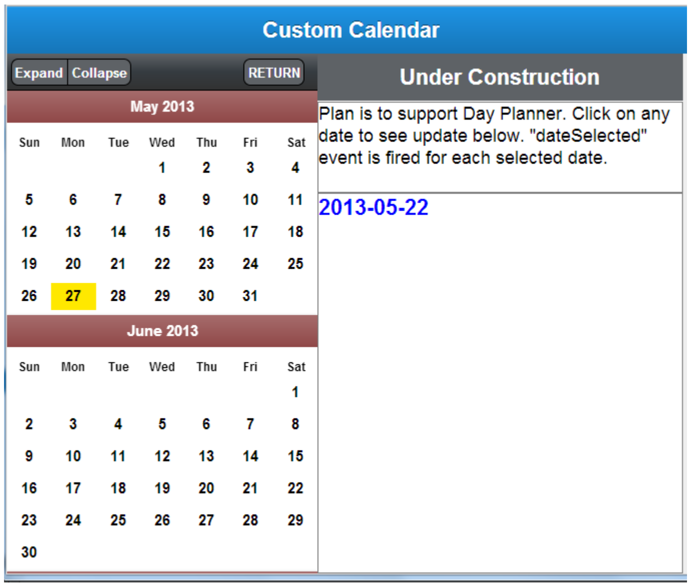
Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

November 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23

- Calendar supports feature to randomly select dates. Upon return only months with pre-selected dates are expanded.

Day Planner Example



Plan is to support Day Planner. Click on any date to see update below. "dateSelected" event is fired for each selected date.

2013-05-22

May 2013

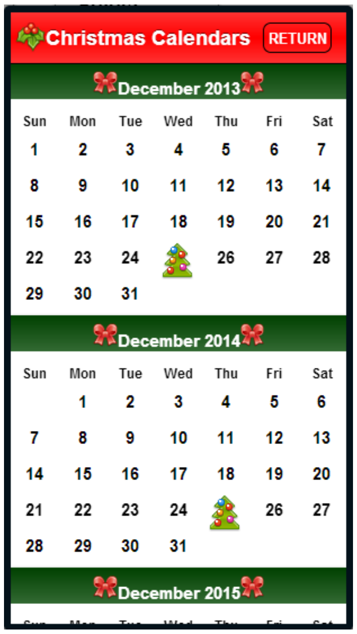
Sun	Mon	Tue	Wed	Thu	Fri	Sat
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

June 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

- Examples forthcoming on how descriptions behind custom dates can be displayed upon selecting particular day
- Full day-planner support with re-occurring meetings and events also planned.

Custom Filter Example with Embedded Icons



December 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	🎄	26	27	28
29	30	31				

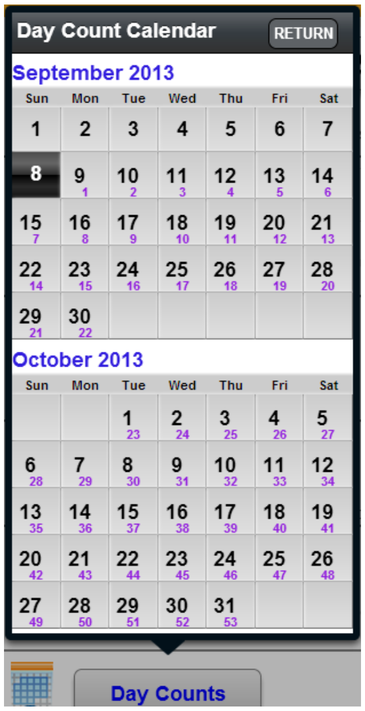
December 2014

Sun	Mon	Tue	Wed	Thu	Fri	Sat
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	🎄	26	27
28	29	30	31			

December 2015

- Easy to implement filter function supported to only include December months.
- Can add icons to Calendar Title, month headers and individual days.

Financial DayCounts Calendar Example



September 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9 1	10 2	11 3	12 4	13 5	14 6
15 7	16 8	17 9	18 10	19 11	20 12	21 13
22 14	23 15	24 16	25 17	26 18	27 19	28 20
29 21	30 22					

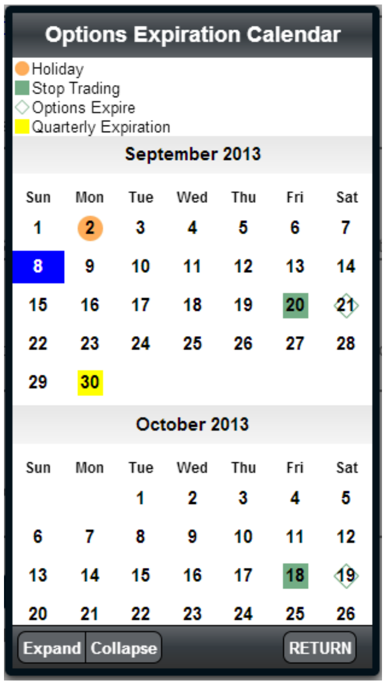
October 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
		1 23	2 24	3 25	4 26	5 27
6 28	7 29	8 30	9 31	10 32	11 33	12 34
13 35	14 36	15 37	16 38	17 39	18 40	19 41
20 42	21 43	22 44	23 45	24 46	25 47	26 48
27 49	28 50	29 51	30 52	31 53		

- IPhone-style read-only Financial calendar that includes DayCounts after current day

- Uses custom rendering function

MarketWatch-style Options Expiration Calendar



● Holiday
🟩 Stop Trading
◇ Options Expire
🟨 Quarterly Expiration

September 2013

Sun	Mon	Tue	Wed	Thu	Fri	Sat
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

October 2013

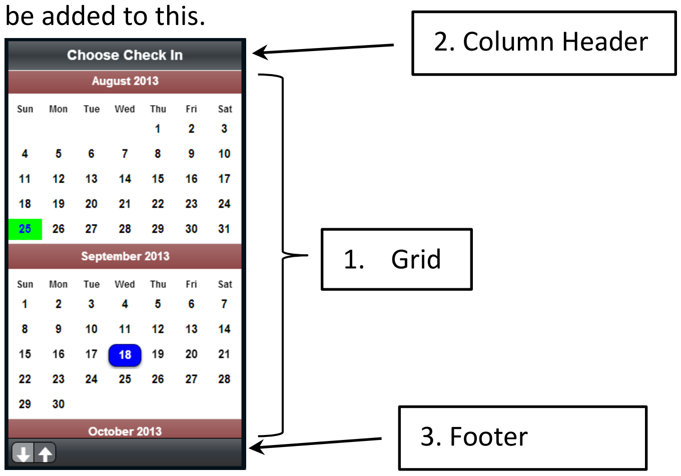
Sun	Mon	Tue	Wed	Thu	Fri	Sat
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26

- Read-only MarketWatch-style Options Expiration calendar
- Supports multiple custom date categories with auto-generated Legend

CalendarPicker

Features

- The CalendarPicker component automatically creates following components simply by linking it as a child to a parent container and specifying a few configurations:

  1. Grid displaying collapsible months
  2. Optional titlebar with customizable title
  3. Optional footer containing buttons to expand all, collapse all, and custom buttons can be added to this.
  4. 

- **Can be used as an overlay panel for pop-up style selection or in any container for fixed display**

- **Following are minimum recommended configurations to create a Calendar:**
  1. **xtype : 'calendarpicker'**
  2. **selectMode: 'SINGLE', 'RANGE', 'MULTI', 'NONE'**
  3. **backMonths (defaults to 3 months in past)**
  4. **forwardMonths (defaults to 3 months in future .. result is calendar containing 6 months total)**
  5. **returnItem: Object reference to container/button that launched the calendar (read more below)**
  6. **itemId**

- Refer to Appendix C for documentation on selectMode config for other options.

- Option to disable selection for:
  1. Custom list of dates
  2. Holidays
  3. Weekends
  4. Days prior to today

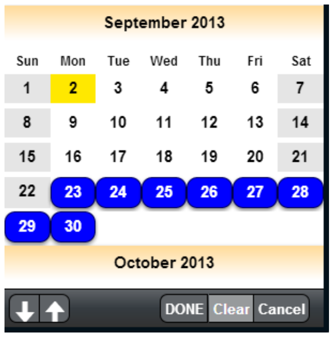
5. Days after today

- Pull-refresh plugin style option to allow user to insert more months into beginning of calendar.

- Paging-plugin style option to allow user to append more months at end of calendar.

- CSS selectors provided to customize styling of:
  1. Disabled days (defaults to light grey text)
  2. Selected days (defaults to blue background and white text)
  3. Holidays (defaults to yellow background)
  4. Current day (defaults to lime-green background and blue text)
  5. Custom Date Categories

- Supports customizable buttons on footer toolbar. Default for RANGE selection is DONE, Clear, Cancel:



- Most features supported by TouchTreeGrid supported for Calendar. Examples:
  1. Disallow expand/collapse
  2. Initially collapse or expand
  3. singleExpand accordion mode
  4. Supports gradient coloring for category rows.
  5. Renderers to customize display of each day (example display icons in individual cells, different text colors, secondary-text in each cell, etc..)

- Uses methods from Ext.Date for all date math

- Supports reverse month sorting.

- All styling defined within CSS file (refer to Appendix B for detailed documentation)

- Custom Rendering functions
  1. Following rendering functions are used by CalendarPicker:
     - renderer_month: function (values)
       `{return values.month + ' ' + values.year;}`
       - used to format calendar month titles

- Refer to Example 5, onExample5_December() controller method for how this was overridden to include Christmas icons in each category row.

- renderer_dates: function (fldName, values) {
```
var elem=values[fldName];
return elem;}
```
  - used to render actual calendar day values. For Header row ‘Mon’, ‘Tue’, etc.. is rendered.
  - Refer to Example 5, onExample5_December() controller method for how this was overridden to include Christmas Tree icons on Christmas day instead of ‘25’
  - Refer to Example 6, onExample6_DayCount() controller method for how this was overridden to include additional DayCount on calendar for all days after today.

- cls_renderer_dates: function (fldName, values)

```
{var cls="", dt, sel, hol, dis, par = this.scope.parent, cust;
 dt = values['dt_'+fldName];
 hol = values['isHoliday_'+fldName];
 dis = values['isDisabled_'+fldName];
 cust = values['customCls_'+fldName];

 if (values.rowType === 'H') {
    cls = cls + ' calendarpicker-header';}
 else if (Ext.isEmpty(dt)) {} // do nothing for empty dates
 else {
    sel = (par.getSelDtArr().indexOf(Ext.Date.format(dt, 'Y-m-d'))>-1);
    if (Ext.Date.format(dt, 'w')==='0' || Ext.Date.format(dt, 'w')==='6') {
      cls = cls+' calendarpicker-weekend';
    }
    if (hol){cls = cls+' calendarpicker-holiday';}
    if (!Ext.isEmpty(cust)) {cls = cls + ' ' + cust;}
    if (!Ext.isEmpty(this.scope.todayDt)){
      if (this.scope.todayDt === Ext.Date.format(dt, 'Y-m-d')) {
         cls = cls+' calendarpicker-today';
      }
    }
    if (dis){cls = cls+' calendarpicker-disabled';}
    if (sel){cls = cls+' calendarpicker-selected';}
 }

return ("calendarpicker-days" + cls);
}
```

  - Used to to define CSS Selector to be applied for each day in calendar.
  - **calendarpicker-days** selector is initially applied to each cell
  - Header rows (Mon, Tues, Wed, etc..) apply **calendarpicker-header** selector to format
  - If no date is provided then no additional selector is applied
  - Then If a weekend then calendarpicker-weekend is applied after those above
  - Then if a holiday (in holidayDtArr[]) then apply calendarpicker-holiday after those above

- Then if a custom date then apply css per customType as defined in customDateTypes config after those above
- Then if current day then apply **calendarpicker-today** after those above
- Then disabled dates styled via **calendarpicker-disabled** selector applied after those above. User not allowed to select disabled days.
- Selected Days then styled via **calendarpicker-selected** selector after those above.

- Refer to Example 5, onExample5_December() controller method for how this was overridden to include Christmas Tree icons on Christmas day instead of '25'
- Refer to Example 6, onExample6_DayCount() controller method for how this was overridden to style special DayCount calendar example
-

2.

Notes on Implementation

1. Setup if not using Architect
   a. Copy 'CalendarPicker.js' and 'TouchTreeGrid.js' from ./CalendarPicker/app/view/ into similar view directory for your project and update Class Name as appropriate.
   b. Include 'TouchTreeGrid' and 'CalendarPicker' in list of 'views' in app.js or your controller
   c. Copy resource files as discussed below.
   d. I'm assuming you know what to do from here if not using Architect…

2. Setup if using Sencha Architect
   a. Import TouchTreeGrid.xdc and CalendarPicker.xdc into your toolbox (via right-click). Created using Architect Version: 2.2.2 Build: 991

   b. Drag these components on top of "Views" in your project inspector to create parent class TouchTreeGrid and CalendarPicker. This will add views reference in your "Application" (app.js)

   c. Suggest copying 'resources' subdirectory from download as is into the same directory level of your project. Key files to include in your project:
      1../resources/css/TouchTreeGrid.css
      2../resources/css/calendarpicker.css (this must be after TouchTreeGrid.css !!)

   d. Add CSS Resource to 'Resources" section of your project inspector:
   Update url = './resources/css/TouchTreeGrid.css'
   Update url = './resources/css/calendarpicker.css'

   e. Optionally add custom CSS Resource **after** these:

1. Update url ='./resources/css/custom.css
2. Definitions in subsequent stack will override prior definitions. When project is saved you will notice in APP.HTML that treegriddemo.css is loaded <u>after</u> TouchTreeGrid.css

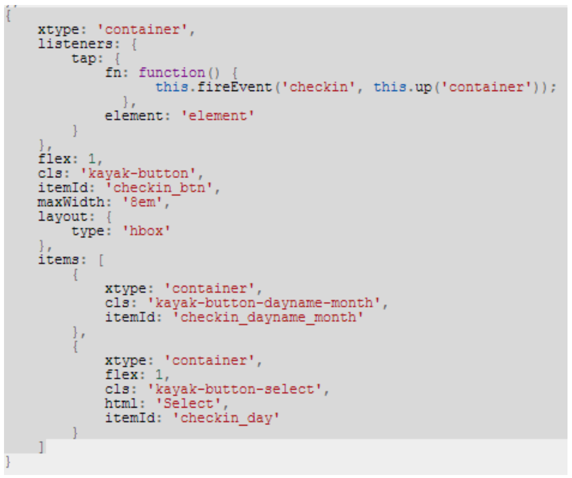
```
<!DOCTYPE html>

<!-- Auto Generated with Sencha Architect -->
<!-- Modifications to this file will be overwritten. -->
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>CalendarPicker</title>
    <script>
        var Ext = Ext || {};
        Ext.theme = {
            name: "Default"
        };
    </script>
    <script src="http://cdn.sencha.com/touch/sencha-touch-2.2.1/sencha-
touch-all-debug.js"></script>
    <link rel="stylesheet" href="http://cdn.sencha.com/touch/sencha-touch-
2.2.1/resources/css/sencha-touch.css">
    <link rel="stylesheet" href="./resources/css/TouchTreeGrid.css">
    <link rel="stylesheet" href="./resources/css/calendarpicker.css">
    <link rel="stylesheet" href="./resources/css/custom.css">
    <script type="text/javascript" src="app.js"></script>
</head>
<body></body>
</html>
```

f. No need to create store or model for CalendarPicker as it creates this automatically (unless implementing custom dates)

g. **Convention for implementation**.

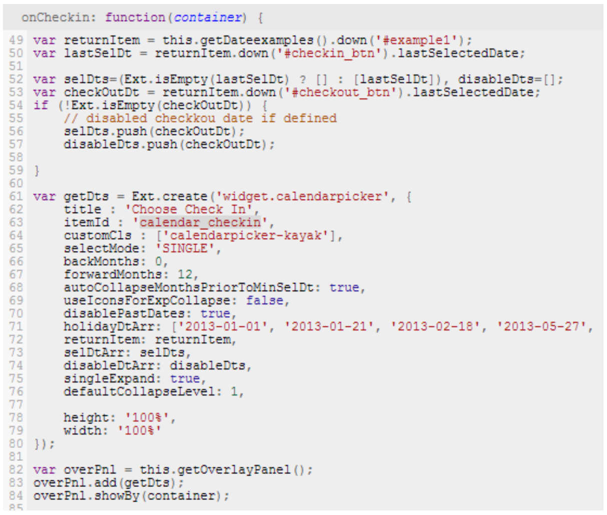
1. Refer to **selectMode** documentation in Appendix C for calendar types.

2. Custom button listens for tap event and fires custom event to Controller.



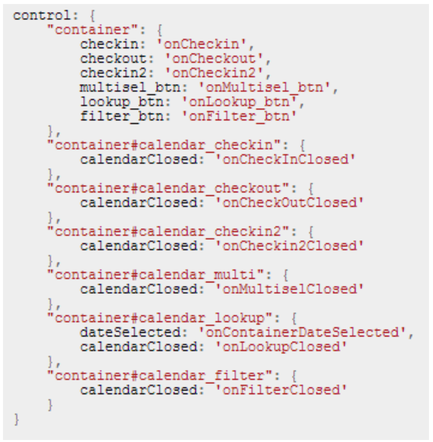
```
{
    xtype: 'container',
    listeners: {
        tap: {
            fn: function() {
                this.fireEvent('checkin', this.up('container'));
            },
            element: 'element'
        }
    },
    flex: 1,
    cls: 'kayak-button',
    itemId: 'checkin_btn',
    maxWidth: '8em',
    layout: {
        type: 'hbox'
    },
    items: [
        {
            xtype: 'container',
            cls: 'kayak-button-dayname-month',
            itemId: 'checkin_dayname_month'
        },
        {
            xtype: 'container',
            flex: 1,
            cls: 'kayak-button-select',
            html: 'Select',
            itemId: 'checkin_day'
        }
    ]
}
```

3. Custom Controller method listens for this custom event and launches Calendar. Each calendar is given a unique ItemId during creation. Note: All provided examples utilize common overlayPanel for displaying calendars with exception of onExample4_Lookup() method which uses datePlanner.js view for display for Day Planner purposes.



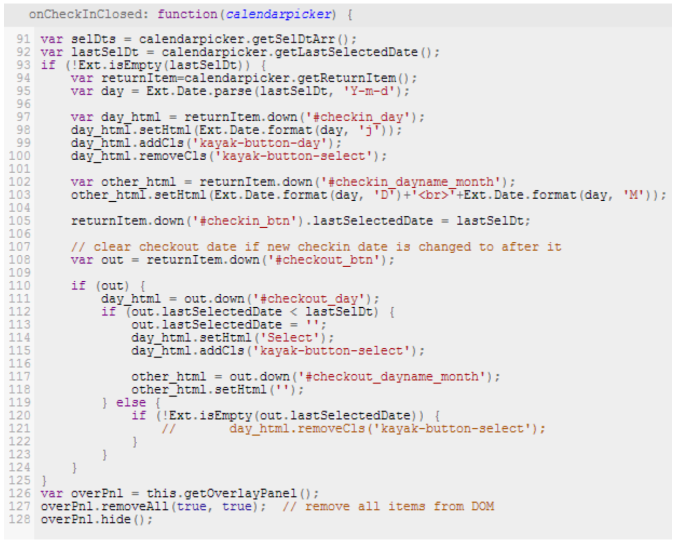
```
onCheckin: function(container) {
var returnItem = this.getDateexamples().down('#example1');
var lastSelDt = returnItem.down('#checkin_btn').lastSelectedDate;

var selDts=(Ext.isEmpty(lastSelDt) ? [] : [lastSelDt]), disableDts=[];
var checkOutDt = returnItem.down('#checkout_btn').lastSelectedDate;
if (!Ext.isEmpty(checkOutDt)) {
    // disabled checkkou date if defined
    selDts.push(checkOutDt);
    disableDts.push(checkOutDt);

}

var getDts = Ext.create('widget.calendarpicker', {
    title : 'Choose Check In',
    itemId : 'calendar_checkin',
    customCls : ['calendarpicker-kayak'],
    selectMode: 'SINGLE',
    backMonths: 0,
    forwardMonths: 12,
    autoCollapseMonthsPriorToMinSelDt: true,
    useIconsForExpCollapse: false,
    disablePastDates: true,
    holidayDtArr: ['2013-01-01', '2013-01-21', '2013-02-18', '2013-05-27',
    returnItem: returnItem,
    selDtArr: selDts,
    disableDtArr: disableDts,
    singleExpand: true,
    defaultCollapseLevel: 1,

    height: '100%',
    width: '100%'
});

var overPnl = this.getOverlayPanel();
overPnl.add(getDts);
overPnl.showBy(container);
```

4. Calendar fire's 'calendarClosed' event when user is done.   Here is partial list of Control Action definitions for the provided examples:



```
control: {
    "container": {
        checkin: 'onCheckin',
        checkout: 'onCheckout',
        checkin2: 'onCheckin2',
        multisel_btn: 'onMultisel_btn',
        lookup_btn: 'onLookup_btn',
        filter_btn: 'onFilter_btn'
    },
    "container#calendar_checkin": {
        calendarClosed: 'onCheckInClosed'
    },
    "container#calendar_checkout": {
        calendarClosed: 'onCheckOutClosed'
    },
    "container#calendar_checkin2": {
        calendarClosed: 'onCheckin2Closed'
    },
    "container#calendar_multi": {
        calendarClosed: 'onMultiselClosed'
    },
    "container#calendar_lookup": {
        dateSelected: 'onContainerDateSelected',
        calendarClosed: 'onLookupClosed'
    },
    "container#calendar_filter": {
        calendarClosed: 'onFilterClosed'
    }
}
```

5. Custom controller method listens for 'calendarClosed' event on custom calendar ItemId and processes calendar selection. Developer responsibility to destroy the calendar from DOM (refer to "overPnl.removeAll(true, true); " in the examples).



```
onCheckInClosed: function(calendarpicker) {
var selDts = calendarpicker.getSelDtArr();
var lastSelDt = calendarpicker.getLastSelectedDate();
if (!Ext.isEmpty(lastSelDt)) {
    var returnItem=calendarpicker.getReturnItem();
    var day = Ext.Date.parse(lastSelDt, 'Y-m-d');

    var day_html = returnItem.down('#checkin_day');
    day_html.setHtml(Ext.Date.format(day, 'j'));
    day_html.addCls('kayak-button-day');
    day_html.removeCls('kayak-button-select');

    var other_html = returnItem.down('#checkin_dayname_month');
    other_html.setHtml(Ext.Date.format(day, 'D')+'<br>'+Ext.Date.format(day, 'M'));

    returnItem.down('#checkin_btn').lastSelectedDate = lastSelDt;

    // clear checkout date if new checkin date is changed to after it
    var out = returnItem.down('#checkout_btn');

    if (out) {
        day_html = out.down('#checkout_day');
        if (out.lastSelectedDate < lastSelDt) {
            out.lastSelectedDate = '';
            day_html.setHtml('Select');
            day_html.addCls('kayak-button-select');

            other_html = out.down('#checkout_dayname_month');
            other_html.setHtml('');
        } else {
            if (!Ext.isEmpty(out.lastSelectedDate)) {
                //       day_html.removeCls('kayak-button-select');
            }
        }
    }
}
var overPnl = this.getOverlayPanel();
overPnl.removeAll(true, true);  // remove all items from DOM
overPnl.hide();
```

Summary of Provided Examples

Refer to provided example source code as summarized here and outlined above under "Conventions for Implementation":

Example	Description	Button/TextField ItemId(s)	Event Fired from Button	Controller Method Processing Btn Events	Calendar ItemId(s)	Controller Method Processing calendarClosed Event

1	Kayak-style Check-in/Check-out date	#checkin_btn
		#checkout_btn
			checkin
			checkout
				onExample1CheckIn()
				onExample1Checkout()
					#calendar_checkin
					#calendar_checkout
						onExample1CheckInClosed()
						onExample1CheckOutClosed()

2	Range Selection in Single Popup	#checkout_btn2	checkin2	onExample2Checkin()	#calendar_checkin2	onExample2CheckinClosed()
2B	IOS-Style Calendar Example	#checkin_btn2B #checkout_btn2B	checkin2B	onExample2BCheckin()	#calendar_checkin2B	onExample2BCheckinClosed()
2C	Single Popup selection Example from TextField	#sel_date	sel_date	onExample2C_Textfield()	#calendar_checkin2C	onExample2C_TextfieldClosed()
3	Multi-select Example	#multisel_btn	multisel_btn	onExample3_Multisel()	#calendar_multi	onExample3_MultiselClosed()
4	Day Planner Example	#lookup_btn	lookup_btn	onExample4_Lookup()	#calendar_lookup	onExample4_LookupSelected() onExample4_LookupClosed
5	Custom Filter Example with Embedded Icons	#filter_btn	filter_btn	onExample5_December()	#calendar_filter	onExample5_DecemberClosed()
6	Financial DayCounts Calendar Example	#day_count_btn	day_count_btn	onExample6_DayCount()	#calendar_daycount	onExample6_DaycountClosed()
7	MarketWatch-style Options Expiration Calendar	#optionDates_btn	optionDates_btn	onExample7_OptionDates()	#calendar_optionDates	onExample7_OptionDatesClosed()

Notable Configurations and Styling by Example:

Example #1: Kayak-style Check-in/Check-out date selection

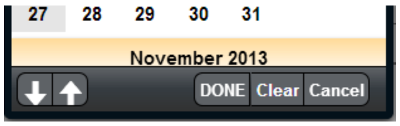
- Check-In
  - selectMode: 'SINGLE'
  - backMonths: 0
  - forwardMonths: 12

  - disablePastDates: true – days prior to today are disabled from selection (CSS selector **calendarpicker-disabled** applied for gray text color)
  - defaultCollapseLevel: 1 – initially collapse all months except current
  - singleExpand: true (only one month expanded at a time for Check-In calendar, unless expand all pressed)
  - customCls : ['calendarpicker-kayak'] – refer to custom styling overrides in custom.css for following selectors:
    - .calendarpicker-kayak .touchtreegrid-list-categ {…
    - .calendarpicker-kayak .calendarpicker-weekend {…
    - .calendarpicker-kayak .calendarpicker-holiday {…
    - .calendarpicker-kayak .calendarpicker-today {…
    - .calendarpicker-kayak .calendarpicker-selected {…
    -

- Check-Out, similar to Check-In except:
  - singleExpand: false (default for TouchTreeGrid)
  - defaultCollapseLevel: 99 – fully expand categories (default for TouchTreeGrid)
  - custom logic implemented to pass list of disabled dates prior to Check-In date

Example #2: Range Selection in Single Popup

- selectMode: 'RANGE'
- includeCustomFooterItems: true – adds default DONE, Clear, Cancel buttons



- useIconsForExpCollapse: true – applies up/down arrows instead of expand/collapse buttons to use less space on toolbar since custom footer items are also included
- backMonths: 0
- forwardMonths: 12
- defaultCollapseLevel: 99 - fully expand calendar (default for TouchTreeGrid)
- customCls : ['calendarpicker-kayak'] – refer to Example 1

Example #2B: IOS-Style Calendar Example

- Configs
  - selectMode: 'RANGE'
  - includeCustomFooterItems: true – DONE, Clear, Cancel buttons
  - backMonths: backMonths
  - forwardMonths: 3 (logic updates this for subsequent navigations if user appends months at end)

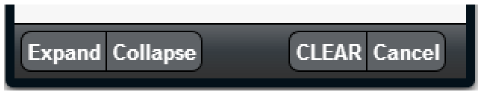
  - itemHeight: 16 - set minimum value for weekday names row to match IOS calendar appearance ... use css for other heights
  - **variableHeights**: **true** – required such that non-weekday rows can be styled > 16 pixels
    NOTE: because of this setting IOS-style calendars may not have same performance as other calendars on slower devices.
  - disablePastDates: true
  - defaultCollapseLevel: 0
  - allowMonthAdditions: true
  - monthsToAppend: 3 – with allowMonthAdditions=true allows user to append 3 months to end of calendar each time last list item tapped
  - monthsToInsert: 0 – user not allowed to insert months at beginning
  - customCls : ['calendarpicker-ios']] – refer to custom styling overrides in custom.css for following selectors:
    - .calendarpicker-ios .calendarpicker-month {…}
    - .calendarpicker-ios .touchtreegrid-list-categ {…}
    - .calendarpicker-ios .touchtreegrid-list-content{…}
    - .calendarpicker-ios .calendarpicker-days {…}
    - .calendarpicker-ios .calendarpicker-selected {…}
    - .calendarpicker-ios .calendarpicker-disabled {…}
    - .calendarpicker-ios .calendarpicker-today {…}
    - .calendarpicker-ios .calendarpicker-header {…}
    - .calendarpicker-ios .calendarpicker-header-rowtype{…}

- Other
  - overPnl.setMaxWidth('17em') - prevent resizing for Phone landscape since we want square boxes
  - overPnl.setModal({ transparent: true }) - keep background transparent instead of gray since IOS calendar is already gray

Example #2C: Single Popup selection Example from TextField

- selectMode: 'SINGLE'
- disableFutureDates: true – would be used when prompting for as-of date for reporting purposes for example
- allowMonthAdditions: true,
- monthsToAppend: 0 – user not allowed to append months in future
- monthsToInsert: 3 – user allowed to insert 3 months at a time at beginning
- customCls : ['calendarpicker-ios'] – refer above

- includeCustomFooterItems: true,
- customFooterItems: myCustomBtns – default buttons overridden as:



```
myCustomBtns = {
    xtype: 'segmentedbutton',
    itemId : 'pickerfooterbtns',
    docked : 'right',
    items: [
    {
        xtype: 'button',
        text: 'CLEAR',
        iconCls: '',
        cls: 'pickerfooterbtns',
        listeners : {
            tap: function (button, e, options) {
                this.up('calendarpicker').customBtns('CLEAR');     // Clear data
                this.up('calendarpicker').customBtns('DONE');
                 // Then fire event to close calendar
            }
        }
    },
    {
        xtype: 'button',
        text: 'Cancel',
        iconCls: '',
        cls: 'pickerfooterbtns',
        listeners : {
            tap: function (button, e, options) {
                this.up('calendarpicker').customBtns('CANCEL');     // Clear data
            }
        }
    }

]};
```

Example #3: Multi-select Example - Later

Example #4: Day Planner Example - Later

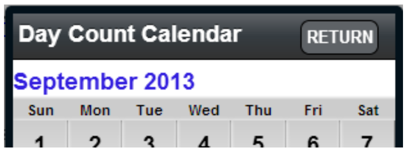
Example #5: Custom Filter Example with Embedded Icons - Later

Example #6: Financial DayCounts Calendar Example

- Configs
  - selectMode: 'NONE'
  - backMonths: 0
  - forwardMonths: 6
  - disableExpandCollapse: true – read-only calendar where expand/collapse disabled

  - hideTitleBar: true – replaced with footer toolbar docked to top
  - hideExpandCollapseBtns: true – refer to customFooterItems

- footerDock: 'top'



- renderers: myRend – refer below

- includeCustomFooterItems: true
- customFooterItems: myCustomBtns. Default config overridden with label and RETURN button via:

```
var myCustomBtns = [     // NOTE:  Currently can't define as array if including
Expand/Collapse All features  (in this case use Title instead of custom Label if
needed)
{
    xtype: 'label',
    html: 'Day Count Calendar',
    cls: 'daycount-label'
},
{
    xtype: 'segmentedbutton',
    itemId : 'pickerfooterbtns',
    docked : 'right',
    items: [
    {
        xtype: 'button',
        text: 'RETURN',
        iconCls: '',
        cls: 'pickerfooterbtns',
        listeners : {
            tap: function (button, e, options) {
                this.up('calendarpicker').customBtns('CANCEL');
            }
        }
    }
]}
];
```

- Renderers
  - Default renderers overridden to display additional day count for future days
  - renderer_dates:

```
renderer_dates: function (fldName, values) {
    var elem=values[fldName];   <= normal day value

    // compute day count value (diff)
    var dt = values['dt_'+fldName];
    var today= Ext.Date.clearTime(new Date(Date(Ext.Date.now())), true);
    var diff = Ext.Date.diff(today, dt, Ext.Date.DAY);

    if (diff>0) {
       // add 'daycount-inner' class to day count value on next line
        elem = elem+'<span class="daycount-inner"><br>'+diff.toString()+'</span>';
    }
    return elem;
},
```

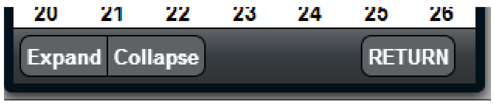
- cls_renderer_dates:

```
cls_renderer_dates: function (fldName, values){
    var cls="", dt, sel, hol, dis, par = this.scope.parent;
    dt = values['dt_'+fldName];

   // Apply 'daycount-outer' class to future days where additional day count
   // will be displayed
    var today= Ext.Date.clearTime(new Date(Date(Ext.Date.now())), true);
    var diff = Ext.Date.diff(today, dt, Ext.Date.DAY);
    if (diff>0) {
        cls = cls + ' daycount-outer';
    }

    hol = values['isHoliday_'+fldName];
    dis = values['isDisabled_'+fldName];

    if (values.rowType === 'H') {
    cls = cls + ' calendarpicker-header';}
    else if (Ext.isEmpty(dt)) {} // do nothing for empty dates
    else {
        sel = (par.getSelDtArr().indexOf(Ext.Date.format(dt, 'Y-m-d'))>-1);
        if (Ext.Date.format(dt, 'w')==='0' || Ext.Date.format(dt, 'w')==='6') {
            cls = cls+' calendarpicker-weekend';
        }
        if (hol){cls = cls+' calendarpicker-holiday';}
        if (!Ext.isEmpty(this.scope.todayDt)){
            if (this.scope.todayDt === Ext.Date.format(dt, 'Y-m-d')) {
                cls = cls+' calendarpicker-today';
            }
        }
        if (dis){cls = cls+' calendarpicker-disabled';}
        if (sel){cls = cls+' calendarpicker-selected';}
    }

    return ("calendarpicker-days" + cls);
}};
```

Example #7: MarketWatch-style Options Expiration Calendar

- Configs
  - selectMode: 'NONE' – selections disabled for purposes of read-only calendar
  - backMonths: 0
  - forwardMonths: 15

  - includeCustomFooterItems: true
  - customFooterItems: myCustomBtns

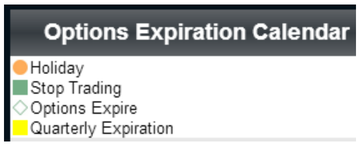


Default 'Cancel' button overridden with 'RETURN' text for read-only calendar:

```
var myCustomBtns = {
    xtype: 'segmentedbutton',
    itemId : 'pickerfooterbtns',
    docked : 'right',
    items: [{
```

```
                xtype: 'button',
                text: 'RETURN',
                iconCls: '',
                cls: 'pickerfooterbtns',
                listeners : {
                    tap: function (button, e, options) {
                        this.up('calendarpicker').customBtns('CANCEL');
                    }
                }
            }]
        };
```

- customDateTypes: customDateTypes – refer below
- includeCustomDatesLegend: true



Options Expiration Calendar

● Holiday
🟩 Stop Trading
◇ Options Expire
🟨 Quarterly Expiration

- customCls : ['calendarpicker-optionDates'] – refer to custom styling overrides in custom.css for following selectors:
  - .calendarpicker-optionDates .touchtreegrid-list-categ {…}
  - .calendarpicker-optionDates .calendarpicker-month {…}
  - .calendarpicker-optionDates .calendarpicker-today {…}

- Each category of custom dates is defined via array of objects. Example:

```
var customDateTypes = [
{
    customType: 'optionsExpire',
    customDescr: 'Options Expire',
    useCustomDtArr: false,
    storeId: 'optionDates',
    priority: 1,
    disabled: false,
    cls: 'optionDates-optionsExpire'
},
Etc..
];
```

  - customType – unique for each set of custom dates
  - customDescr – used for auto-generated Legend
  - storeId – defines store providing the dates for customType (see below)
  - cls – defines CSS Selector used to style this date unique from the other custom type. Refer to custom styling defined in custom.css for following selectors:
    - .optionDates-holiday {
       background: url(cal-holiday.png) no-repeat center;
      /* Images must be stored in ./resources/css/ directory as referenced */
      }
    - .optionDates-stopTrading {
      background: url(cal-stop-trading.png) no-repeat center;
      }

- .optionDates-optionsExpire {
  background: url(cal-expiration.png) no-repeat center;
  }
- .optionDates-quarterlyExpiration {
  background: url(cal-quarterly-expiration.png) no-repeat center;
  }
- .calendarpicker-optionDates .calendarpicker-legend-cls-override {
  background-size: 80% 80% !important;
  padding-right: 1em ;
  }

- Custom dates defined in Store and auto-applied to Calendar. Example:

```
{"customType":"optionsExpire","dateStr":"2013-09-21","descr":null},
{"customType":"optionsExpire","dateStr":"2013-10-19","descr":null},
Etc..
```

  - Fields "customType" and "dateStr" are the only required custom fields, otherwise store can be used for other custom purposes.

APPENDIX A – CSS Styling

Every project should include a copy of **TouchTreeGrid.css** and **calendarpicker.css** which define the default styling for CalendarPicker component which uses TouchGreeGrid component. Calendarpicker.css must be loaded after TouchTreeGrid.css. Any custom overrides should be included in a file loaded after TouchTreeGrid.css. custom**.css** contains examples of custom styling overrides.

Refer to Appendix B from "TouchTreeGrid – documentation.pdf" for explanation of TouchTreeGrid.css styling convetions.

Review of the CalendarPicker CSS classes and how they are used:

- .x-touchtreegrid-list-calendar .touchtreegrid-list-categ {…}
  - Styles month header row.
  - Default is maroon gradient color with white bold text

- .x-touchtreegrid-list-calendar .touchtreegrid-list-content {…}
  - Styles rows representing each week of the month.
  - Default is white background with black text

- .x-touchtreegrid-list-calendar .x-list-normal .x-list-item.x-list-item-tpl{ border: none; }
  - Used to hide horizontal lines

- .x-touchtreegrid-list-calendar .calendarpicker-month {…}
  - Used to style text on month headers.
  - Default is centered, white-bold text

- .x-touchtreegrid-list-calendar .calendarpicker-toolbar {…}
  - Used to style TitleBar above the calendars
  - Default is dark gray gradient

- .x-touchtreegrid-list-calendar .touchtreegrid-expand-collapse-buttons, .x-touchtreegrid-list-calendar .pickerfooterbtns{…}
  - Used to style buttons on footer toolbar

- .x-touchtreegrid-list-calendar .x-button.x-button-pressing, .x-touchtreegrid-list-calendar .x-button.x-button-pressing:after, .x-touchtreegrid-list-calendar .x-button.x-button-pressed, .x-touchtreegrid-list-calendar .x-button.x-button-pressed:after, .x-touchtreegrid-list-calendar .x-button.x-button-active, .x-touchtreegrid-list-calendar .x-button.x-button-active:after {…}
  - Used to style color of footer toolbar buttons when pressed

- .x-touchtreegrid-list-calendar .calendarpicker-header-rowtype{…}

- Used to style row containing day of week names

- .x-touchtreegrid-list-calendar .calendarpicker-days {…}
  - Default used to style all day cells
  - Default is center and bold

- .x-touchtreegrid-list-calendar .calendarpicker-today {…}
  - Used to style current day
  - Default is lime green with bold blue text color

- .x-touchtreegrid-list-calendar .calendarpicker-weekend {…}
  - Used to style weekend days
  - No defaults defined, but this can be used in custom css file for specific calendars.

- .x-touchtreegrid-list-calendar .calendarpicker-holiday {…}
  - Used to style holidays
  - Default is yellow background

- .x-touchtreegrid-list-calendar .calendarpicker-selected {…}
  - Used to style selected days
  - Default is blue with bold white text, border with radius and shadow

- .x-touchtreegrid-list-calendar .calendarpicker-disabled {…}
  - Used to style disabled days
  - Default is gray text, normal font

- .customDates-default {background-color: orange;}
  - Default color applied to custom dates supplied in customDtArr[]
  - Refer to config documentation on “customDateTypes” for customized styling for multiple date types

- .calendarpicker-legend {font-size: .7em;}
  - Default font size for auto-generated legend for custom dates

- .calendarpicker-legend-cls-override {}
  - Used by Example #7 to resize the legend icons in custom.css:

```
.calendarpicker-optionDates .calendarpicker-legend-cls-override {
    background-size: 80% 80% !important;
    padding-right: 1em ;
```

APPENDIX B – Upgrading CalendarPicker component

Follow steps outlined in Appendix C from “TouchTreeGrid – documentation.pdf”

APPENDIX C – CalendarPicker Config Definitions

Also refer to Appendix D from “TouchTreeGrid – documentation.pdf”

Note: minimum required configurations to define in linked instance of CalendarPicker:

- xtype : ‘calendarpicker’
- selectMode: 'SINGLE’, ‘RANGE’, ‘MULTI’, ‘NONE’ (refer below for details)
- returnItem: Object reference to container/button that launched the calendar (read more below)
- itemId

Configuration	Default	Purpose
allowMonthAdditions (added 9/8/13)	false	Used in conjunction with configs monthsToAppend and monthsToInsert to activate ability for user to auto-insert months into pre-defined calendar.
applyDefaultCollapseLevel	true	Set to false if collapse levels defined on server side.
autoCollapseMonthsPriorToMinSelDt	false	When true all months that do not have a prior selected date (defined in selDtArr) will be collapsed so that focus is on the earliest month with a prior selected date (refer to Example1)
autoExpandMonthsWithSelDates	True	If prior selected dates are passed via selDtArr[] then any such months will be auto-expanded regardless of any other settings.
backMonths	3	Used in conjunction with forwardMonths to build the calendar for display from current date. This is the number of months prior to today that will be included in the calendar display.
categColumns[]	[{ dataIndex: 'month', width: '100%', categCss: 'calendarpicker-month', renderer: 'this.renderer_month(values)' }]	Used to define month headers in calendar
categDepthColors	true	Set to true to apply default color schemes defined by categDepthColorsArr to category rows.
categDepthColorsArr[]	[' transparent, ' transparent,’ ‘transparent’]	Array defining colors to apply to category depths 1,2,3, etc… Use transparent to support gradients on month headers (i.e. category row).
columns	[]	Refer to Appendix A from TouchTreeGrid documentation. Also refer to CalendarPicker.js on how this config is used to render each day in calendar.

Configuration	Default	Purpose
customCls	[]	Specifies CSS selector to be added to TouchTreeGrid component for overriding default styling. Refer to provided examples.
customDateTypes (added 9/8/13)	[{ customType: 'default', customDescr: 'Custom', useCustomDtArr: true, storeId: null, priority: 1, disabled: false, cls: 'customDates-default' }]	Single list of custom dates can be passed via customDtArr[] in 'YYYY-MM-DD' format. If so, default configuration for customDateTypes as shown is required to process passed dates. Cls='customDates-default' is the default styling applied for this list but can be overridden. Refer to onExample7_OptionDates() Controller method for multi-custom date category example (Options Expiration). For this case all custom dates are provide in one or more stores as referenced in storeId attribute of cusotmDateTypes config. Other attributes: • customType = unique name for each type of custom dates • customDescr is displayed in auto-generated legend • useCustomDtArr = false for store-defined dates • priority = defines precedence when applying cls when different customTypes must be rendered for same calendar day. Priority 1 is applied last and takes precedence over Priority 2 for example. • Disabled = if true then .calendarpicker-disabled class is applied to this date (i.e. light gray) and the date is not selectable. • Cls – refer to custom.css for examples of custom styling: .optionDates-optionsExpire { background: url(cal-expiration.png) no-repeat center; } Notes regarding store defining custom dates: • Required fields: ○ customType – matches customType from customDateTypes config above ○ dateStr – 'YYYY-MM-DD' format • Additional fields can be included for custom purposes Notes regarding auto-generated Legend: • Must set includeCustomDatesLegend = true config • Physical order of customDateTypes[] array of objects as defined controls order of display in Legend. • .calendarpicker-legend-cls-override selector can be used to override size/styling of legend colors/icons (refer to custom.css)
customFooterItems	{}	Allows user-defined object of components to be added to footer toolbar (i.e. to same toolbar as expand/collapse buttons). Requires

Configuration	Default	Purpose
		includeCustomFooterItems=true config. Refer to CalendarPicker component for sample usage
customExpCollapseEvent	'monthExpCollapse'	Used internally, but can be overridden for custom purposes when expand/collapse all
defaultCollapseLevel	99	Tree level to expand to for initial display (99 is fully expanded).
disableDtArr	[]	Array of custom dates to be disabled from selection. CSS selector **calendarpicker- disabled** will be applied to these days. Date format is 'YYYY-MM-DD'
disableExpandCollapse	False	Allows expand/collapse feature for TreeGrids to be disabled and 'frozen' in initial rendered state.
disableFutureDates	False	If true, all dates after today will not be selectable and CSS selector **calendarpicker-disabled** will be applied to those days
disableHolidays	False	If true, all dates included in holidayDtArr[]will not be selectable and CSS selector **calendarpicker-disabled** will be applied to those days
disablePastDates	False	If true, all dates prior to today will not be selectable and CSS selector **calendarpicker-disabled** will be applied to those days
disableWeekends	false	If true, all Saturdays and Sunday's will not be selectable and CSS selector **calendarpicker-disabled** will be applied to those days
enableQuickDaySelection	false	Only applicable for selectMode = 'NONE' and controls whether simulated selection appears when day is pressed. Example4 uses this to quickly display selected day when updating day planner.
expandCurrentMonth (added 9/8/13)	True	If true causes current month to be expanded regardless of defaultCollapseLevel. Ignored if collapsing all via bottom toolbar button, or if auto-expanding months with prior-selected dates after current month.

filter	{}	Object defining filter function for custom filtering (refer to Christmas example which filters on December months only)
		`{`
		`enabled: true,`
		`displayNodesWithAllMembersFilteredAsLeafs: true,`
		`filterFn: function(rowObj) {return (rowObj.month === 'December');}`
		`}`

Configuration	Default	Purpose
footerDock	'bottom'	Valid values are 'top' or 'bottom'. Defines how auto-generated footer toolbar containing expand/collapse plus optional custom buttons will be docked. If docked top it will appear above the column header toolbar.
forwardMonths	3	Used in conjunction with backMonths to build the calendar for display from current date. This is the number of months from today that will be included in the calendar display.
hideExpandCollapseBtns	False	When true causes the auto-generated Expand/Collapse buttons to be hidden on the footer toolbar, where custom buttons defined by customFooterItems config would persist.
hideTitleBar	False	When true default Titlebar for calendar is not shown. This allows for custom title bars to be added to your calendar (external to this component)
holidayDtArr	[]	Array of holidays to be added to your calendar. CSS selector **calendarpicker-holiday** will be applied to these days. Date format is 'YYYY-MM-DD'. Example: holidayDtArr: ['2013-01-01', '2013-01-21', '2013-02-18', etc…]
includeCustomDatesLegend (added 9/8/13)	False	Refer to customDateTypes config.
includeCustomFooterItems	false	Refer to customFooterItems config.
includeFooter	true	Set false to prevent auto-creation of footer. Not applicable for simple grids (simpleList=true).
itemHeight	32	32 pixel height for each row of weeks
lastSelectedDate	[]	Used primarily for selectMode='SINGLE' to pass back the selected date in 'YYYY-MM-DD' format.
layout	{type: 'fit'}	For internal use only. Parent container to linked instance should have Layout = 'card' or 'fit' typically.

monthNodeHeightInPixels (added 9/8/13)	32	All of the examples specify .touchtreegrid-list-categ css selector which includes:
		min-height: 32px !important;
		height: 32px !important;
		max-height: 32px !important;
		
		monthNodeHeightInPixels should match whatever pixel height you specify for the category rows (i.e. month header rows). This is used to compute the ScrollTo value for first month to display based on minimum selected date. For now this has to be specified in pixels until Sencha allows support of list itemHeight in other terms.

monthsToAppend	3	Used in conjunction with allowMonthAdditions config

Configuration	Default	Purpose
(added 9/8/13)		to allow user to add # of months to end of current calendar using button at end of scrolled list that reads “Append more Months...”. See also config monthsToAppendTex. Refer to Example 2B, onExample2BCheckin() controller method for example.
monthsToAppendText (added 9/8/13)	'Append more Months...'	Refer to monthsToAppend config
monthsToInsert (added 9/8/13)	3	Used in conjunction with allowMonthAdditions config to allow user to add # of months to beginning of current calendar using push-refresh style interface that reads “Pull to Insert Months...” and “Release to Insert Months...”. See also configs monthsToInsertPullText and monthsToInsertRefreshText. Refer to Example 2C, onExample2C_Textfield() controller method for example.
monthsToInsertPullText (added 9/8/13)	'Pull to Insert Months...'	Refer to monthsToInsert config
monthsToInsertRefreshText (added 9/8/13)	'Release to Insert Months...'	Refer to monthsToInsert config
renderers	See code	Refer to Features discussion
returnItem	{}	Object reference to container/button that launched the calendar. CalendarPicker instances are typically initiated from Controller method after some button is pressed. The object reference to that button is stored to the calendar object via **returnItem** config. The different selectModes for CalendarPicker will eventually fire **‘calendarClosed’** event which would be picked up in a different Controller method. That method would use **returnItem** object stored with the calendar object to update the calendar selection on the original button. Refer to onCheckin() and onCheckInClosed() methods for typical example (Example1).
reverseSort	False	If true the months are sorted in reverse.
selDtArr	[]	Initial array of selected days to be added to your calendar. CSS selector **calendarpicker-selected** will be applied to these days. Date format is ‘YYYY-MM-DD’. Example: selDtArr: ['2013-01-01', '2013-01-21', '2013-02-18', etc…]
selectMode	‘SINGLE’	**SINGLE** - **‘calendarClosed’** event fired after first non-disabled day selection. Typically used with includeCustomFooterItems=false so that only expand/collapse buttons are displayed. User’s responsibility to close and destroy the calendar. Refer to onCheckOutClosed() method as an example of how to close and destroy for overlay panel. Refer to onLookupClosed() for how to close and destroy from non-modal panel.

Configuration	Default	Purpose
		RANGE – allows user to select min and max days (all non-disabled days between are auto-selected). Typically used with includeCustomFooterItems=true and useIconsForExpCollapse=true which auto-adds buttons to footer toolbar for user to press 'DONE', 'Clear' or 'Cancel'. 'DONE' and 'Cancel' cause '**calendarClosed'** event to be fired which would be processed in Controller by user. Refer to onCheckin2Closed() method for example. **MULTI** – allows user to select random days. Otherwise implemented same way as RANGE. Typically used with autoCollapseMonthsPriorToMinSelDt=true and defaultCollapseLevel=1 when prior selection is passed via selDtArr[] such that only months with selected days are expanded. Refer to onMultisel_btn() method for example. **NONE** – Prevents user selection. Exception is temporary display of selected day via enableQuickDaySelection=true. Typically used with custom footer buttons only including 'RETURN' button for example. Refer to onLookup_btn() method for example of how to define custom footer button and to onContainerDateSelected() method for how to process intermediate selections (like for a Day Planner) and to onLookupClosed() method for how to finally close the calendar when RETURN button pressed.
selectedCls	'touchtreegrid-item-selected'	For cases where row selection is desired (simpleList=true), TouchTreeGrid.css contains 'touchtreegrid-item-selected' selectors to specify color for selected items.
singleExpand	false	Typically used for Accordions where only one sibling branch expanded at a time.
title	'Select a Date'	Use to customize titlebar on calendar
useIconsForExpCollapse	true	By default replaces standard TouchTreeGrid 'expand' and 'collapse' buttons with up/down arrows to minimize space used on the toolbar. setIconCls('arrow_up') and setIconCls('arrow_down') on the buttons used for this. Override these classes in you CSS file for this particular calendar instance to use different icons.
variableHeights	false	Refer to Sencha documentation. For best scrolling performance variableHeights should be false when adjusting itemHeight (from default of 47 to 32 for the calendar). Would set this to true if applying custom heights to different rows.