FullCalendar2 Wicket Component
Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Screenshots</td>
<td>2</td>
</tr>
<tr>
<td>Parented collection as calendar</td>
<td>2</td>
</tr>
<tr>
<td>Drill down</td>
<td>3</td>
</tr>
<tr>
<td>Standalone collection as calendar</td>
<td>4</td>
</tr>
<tr>
<td>Calendars</td>
<td>5</td>
</tr>
<tr>
<td>API &amp; Usage</td>
<td>7</td>
</tr>
<tr>
<td>CalendarEventable` interface</td>
<td>7</td>
</tr>
<tr>
<td>Calendarable interface</td>
<td>8</td>
</tr>
<tr>
<td>CalendarableDereferencingService</td>
<td>8</td>
</tr>
<tr>
<td>How to configure/use</td>
<td>9</td>
</tr>
<tr>
<td>Classpath</td>
<td>9</td>
</tr>
<tr>
<td>Known issues</td>
<td>10</td>
</tr>
<tr>
<td>Dependencies</td>
<td>11</td>
</tr>
</tbody>
</table>
This component (isis-wicket-fullcalendar2) renders events for a collection of entities within a fullpage calendar. Underneath the covers it uses this fullcalendar widget.
Screenshots

The module’s functionality can be explored by running the quickstart with example usage using the org.incode.domainapp.example.app.modules.ExampleDomWktFullCalendar2AppManifest.

A home page is displayed when the app is run:

![Home page](image)

Parented collection as calendar

The todo item’s collection contains a list of Calendarable entities (also todo items); this is indicated through a button to switch the view:
Clicking the button shows the same entities on a fullpage calendar:

Drill down

Clicking on the event in the calendar drills down to the corresponding entity:
Standalone collection as calendar

Invoking an action that returns a list of `Calendarable` entities:

... also results in the button to view in a fullpage calendar:
Each item is shown in the calendar view:

Calendars

Each entity can provides dates to either a single calendar or to multiple calendars. In the example app each todo item exposes its dueBy date to a single calendar, named after its category:
@Programmatic
@Override
public String getCalendarName() {
    return getCategory().name();
}

@Programmatic
@Override
public CalendarEvent toCalendarEvent() {
    if(getDueBy() == null) {
        return null;
    }
    return new CalendarEvent(getDueBy().toDateTimeAtStartOfDay(), getCalendarName(), container.titleOf(this));
}

The full page calendar uses colour coding to indicate the calendars, as well as checkboxes to show/hide each calendar. Unchecking the calendar toggle hides all events in that calendar:
**API & Usage**

Each entity must implement either the `CalendarEventable` interface or the `Calendarable` interface:

### `CalendarEventable` interface

Of the two interfaces, `CalendarEventable` interface is the simpler, allowing the object to return a single `CalendarEvent`:

```java
public interface CalendarEventable {
    String getCalendarName();  // ①
    CalendarEvent toCalendarEvent();  // ②
}
```

① groups similar events together; in the UI these correspond to checkboxes rendered near the top.
② returns a `CalendarEvent` value type representing the data to be rendered on the calendar.

`CalendarEvent` itself is:

```java
public class CalendarEvent implements Serializable {
    private final DateTime dateTime;
    private final String calendarName;
    private final String title;
    private final String notes;

    public CalendarEvent(
            final DateTime dateTime,
            final String calendarName,
            final String title)
    {
        this(dateTime, calendarName, title, null);
    }

    public CalendarEvent(
            final DateTime dateTime,
            final String calendarName,
            final String title,
            final String notes)
    {
        this.dateTime = dateTime;
        this.calendarName = calendarName;
        this.title = title;
        this.notes = notes;
    }
    ...
}
```

In the demo app, the `ToDoItem` implements `CalendarEventable`.
**Calendarable interface**

While the `CalendarEventable` interface will fit many requirements, sometimes an object will have several dates associated with it. For example, one could imagine an object with start/stop dates, or optionExercise/optionExpiry dates.

The `Calendarable` interface therefore allows the object to return a number of `CalendarEvents`; each is qualified (identified) by a `calendarName`:

```java
public interface Calendarable {
    Set<String> getCalendarNames();
    ImmutableMap<String, CalendarEventable> getCalendarEvents();
}
```

**CalendarableDereferencingService**

Sometimes the domain object that implements `Calendarable` or `CalendarEventable` will be a supporting object such as a `Note` attached to an `Order`, say. When the marker is clicked in the calendar, we would rather that the UI opens up the `Order` rather than the associated `Note` (in other words, saving a click).

This requirement is supported by providing an implementation of the `CalendarableDereferencingService`:

```java
public interface CalendarableDereferencingService {
    @Programmatic
    Object dereference(final Object calendarableOrCalendarEventable);
}
```

for example, one might have:

```java
public class LocationDereferencingServiceForNote implements CalendarableDereferencingService {
    @Programmatic
    public Object dereference(final Object calendarableOrCalendarEventable) {
        if (!((Locatable) instanceof Note)) {
            return null;
        }
        final Note note = (Note) calendarableOrCalendarEventable;
        return note.getOwner();
    }
}
```

Note that there can be multiple implementations of this service; the component will check all that are available. The order in which they are checked depends upon the `@DomainServiceLayout(menuOrder=...)` attribute.
How to configure/use

Classpath

Add this component to your project's dom module's pom.xml, eg:

```
<dependency>
    <groupId>org.isisaddons.wicket.fullcalendar2</groupId>
    <artifactId>isis-wicket-fullcalendar2-cpt</artifactId>
</dependency>
```

Check for later releases by searching Maven Central Repo.
Known issues

None known at this time.
Dependencies

Maven can report modules dependencies using:

```
mvn dependency:list -o -pl modules/wkt/fullcalendar2/impl -D excludeTransitive=true
```

which, excluding Apache Isis itself, returns these compile/runtime dependencies:

```
net.ftlines.wicket-fullcalendar:wicket-fullcalendar-core:jar:2.2.0
```

For further details on 3rd-party dependencies, see:

- [42Lines/wicket-fullcalendar](http://42Lines/wicket-fullcalendar)

  In turn, this uses Javascript components:

  - [http://jquery.com](http://jquery.com) (MIT License)