Calendar Event and Task List Manager:

Final Report

Prepared by Team Thundercats

Shubhangi Rakhonde
David Schechter
Zayd Hammoudeh

Date of Submission: April 3, 2015

Table of Contents

1. Summary 2
2. Application Overview 2
3. Application Files and Source Code 2
4. Prototype Presentation 2
5. Design Patterns 3
   5.1. Inlay List Design Pattern 3
   5.2. Alternate Views Design Pattern 4
   5.3. Prompting Text Field Design Pattern 4
   5.4. Modal Dialog Design Pattern 6
   5.5. Drop Down Chooser Design Pattern 8

1. Summary

On March 12, 2015, Team Thundercats presented the prototype of our calendar appointment and to-do list manager application. We received positive feedback from both the class and Professor Mak. The application and presentation met all of our key objectives, and we felt they were a success based off both our personal standards as well as in comparison to our peers.

This document provides a brief description of our application including its key features, a source code overview, our presentation methodology, and examples of five design patterns we used.

2. Application Overview

It is common for a person to have multiple daily calendars stored on different, disconnected platforms. Managing and visualizing these disparate calendars can be cumbersome and difficult. Our application simplifies this otherwise burdensome task by integrating all of a user's different calendars into a unified calendar where a user can visualize and modify all of his/her calendars through a single, cohesive interface.

In addition to scheduled meetings and appointments, an individual usually must also complete a set of tasks, chores, errands, etc. Our application also integrates the ability to create and manage a user's tasks in the form of an advanced "to-do list".

By juxtaposing in a single interface an individual's calendar with the tasks s/he must perform, a user is able to easily visualize and prioritize all of his/her daily activities. Therefore, this application's integrated approach helps prevent the inefficiencies and issues (e.g. belated completion of tasks) associated with what for most is an unstructured system to daily activity management.

3. Application Files and Source Code

Our tool was written using a combination of jQuery and JavaScript and incorporated multiple open source utilities (e.g. jQuery FullCalendar[1]). As with many jQuery-based websites, the application will not render correctly unless it is run inside a web server. To solve this, we used the built in web server capabilities provided by GitHub. Below is a link to our two main web pages served by GitHub.

- **Login Page: http://rawgit.com/ZaydH/CS235/master/Assignment_2_ToDo_Web_Application/login.html**
- **Main Page: http://rawgit.com/ZaydH/CS235/master/Assignment_2_ToDo_Web_Application/mainpage.html**

In our submission, we also included our source code and libraries in a zip file named: "Assignment_2_ToDo_Web_Application.zip". During debug, some team members used Aptana Studio[2] to locally serve the pages. However, it is our view that the GitHub server coupled with the source code should be sufficient for evaluation and grading purposes.

4. Prototype Presentation

In contrast to the other teams, our prototype presentation utilized only a very small number of slides. A slide-focused presentation can quickly cause the audience to lose interest and/or become distracted. It was our position that a very short presentation followed by a longer demonstration would be more engaging and informative to the audience. Given the feedback we received from both Professor Mak and our peers in the

[1] JQuery Full Calendar is available at: http://fullcalendar.io/.
[2] Aptana Studio 3 is available at: http://www.aptana.com/. This program is based off of Eclipse, and its debug mode includes a built in web server.

class, we felt this strategy was very successful. Our PowerPoint presentation, named "CS235 - Assignment #3 - Application Prototype.pptx", is included with this submission.

5. Design Patterns

Design patterns provide solutions to often encountered software challenges; they serve as best practices that have been refined through proven design experience. In this section, we describe five of the design patterns that were incorporated into our application. Note that this list is not intended to be exhaustive; rather, when selecting which design patterns to describe in the subsequent sections, we strove to balance the criticality of the pattern to our overall design with how well we thought the implementation could be described in this format.

5.1. Inlay List Design Pattern

An inlay list displays a list of items (usually text based) as a single column; when a user selects/clicks on an item in the list, that item's details are displayed below it. This pattern allows for a significant amount of content to be displayed in a relatively compact space.

In our application, we used the Inlay List Pattern to display the user's to do list. We decided on this methodology because a user could have dozens of items or more in their to-do list. What is more, each to-do list item will have several pieces of information associated with it including: a title, description, priority, completion due date, etc. To display all of this information at once for each to-do list item would be a large visual cognitive load and would greatly reduce the scanability of the list. In contrast, the inlay list allowed us to display the minimum amount of information possible while at the same time giving the user the flexibility to display additional details as required by that user's specific goals.

Figure 1 shows our application's to-do inlay list in its unexpended form. Note that for the four items in the list, only the task title is display. When the user clicks on an item in the list, the item expands to display all of its associated information. Figure 2 shows the expanded information associated with the "Mow the Lawn" to-do list item. Note that the description, due date, priority, and completion check box are now displayed below the item's title.

We included two additional features in our to-do inlay list to improve the overall implementation of this pattern. First, when a to-do list item is unexpanded, it has a "+" next to its title as indication to the user that this item can be expanded. When a to-do list item is expanded, the "+" changes to a "-" to indicate to the user that the item can be collapsed. While this may appear subtle to some users, for others, it is a clear affordance to indicate that the structure is an inlay list.

The second additional feature we added was to use color to distinguish between a to-do list item's title (which always has a white background) and its description field (which always has a light blue background). This will allow a user to quickly visually recognize the nature of a particular section of displayed text without relying on recall.

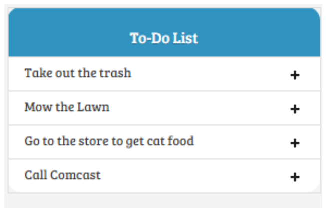
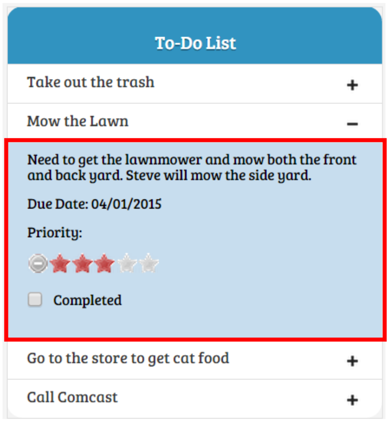


Figure 1 – Unexpanded Inlay To-Do List



Need to get the lawnmower and mow both the front and back yard. Steve will mow the side yard.

Due Date: 04/01/2015

Priority:

Completed

Figure 2 – Expanded Inlay To-Do List

5.2. Alternate Views Design Pattern

When using a software program or tool, often a single, "one-size-fits-all" view is insufficient to allow a user to extract all of the requisite information from a set of data. For example in any calendar application including ours, a user may want to know the meetings s/he has on a specific day; if his/her calendar is particularly full on that day, s/he may want the application to exclusively display that day's meetings. In contrast, if the same user wants to know what days in the next month s/he can schedule an all day trip, s/he may want to view the whole month's calendar at once. We used the Alternate Views Pattern in our application to addresses these types of varying user needs by allowing a user to select the calendar view that best suits his/her current goals. In our application, a user can select between three primary calendar views:

- Day View – View an hour by hour breakdown of a user's appointments for a specific day.
- Week View – View an hour by hour breakdown of a user's appointments for a specific week.
- Month View – A higher level view of a user's appointments for an entire month.

These three alternate views are shown in figures 3, 4, and 5 respectively. Note that the view is selectable by clicking on the name of the view (e.g. "day", "week", and "month") in the upper right corner of the calendar as shown in each of these figures.

5.3. Prompting Text Field Design Pattern

A web page is more likely to make traction with users if it is intuitive and easy to use. One of the ways to increase the intuitiveness of a page is to reduce the amount users must think when using it. The Prompting Text Field Pattern includes prefilled information inside a text field to inform the user regarding the nature of the input field.

To reduce the amount a user must think when using our application and to reduce the likelihood of user error, our application used the Prompting Text Field design pattern in two different places. First, on the login page (shown in figure 6), the username field is prepopulated with the text "Username", and the password field has pre-masked characters to indicate to the user that they should enter their password in the lower box. This type of formatting will allow a user to quickly recognize the type of input taken by each field with little to no thought. While the password prompting is somewhat more subtle than the model used for the "Username" field, we felt this was sufficient for the average user.

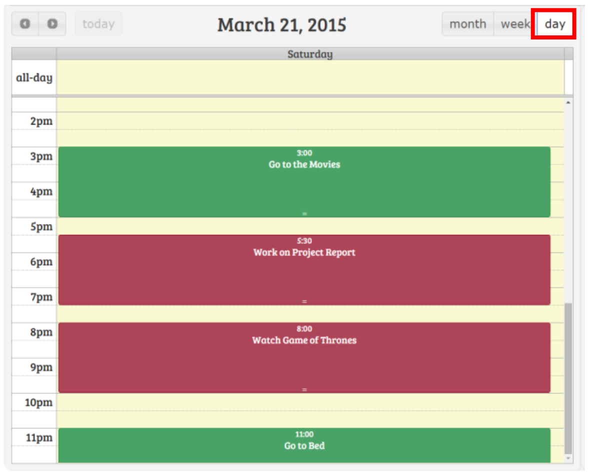


Figure 3 – Calendar Appointments Day View

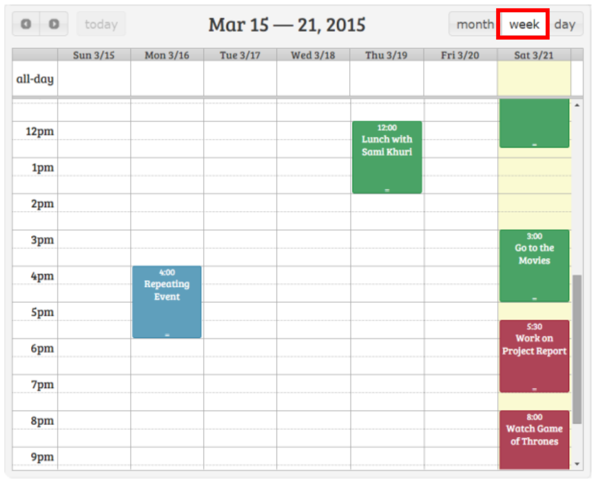


Figure 4 – Calendar Appointments Week View

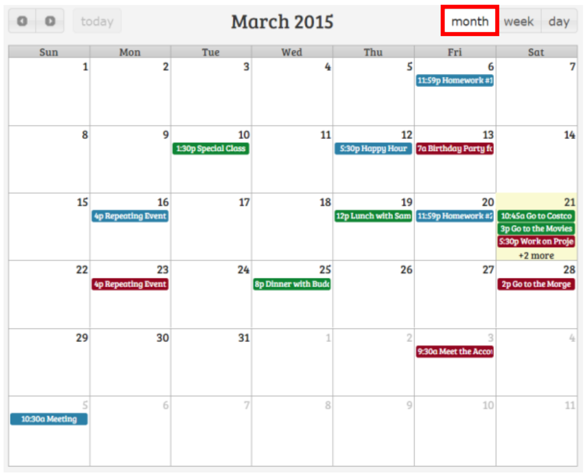


Figure 5 – Calendar Appointments Month View

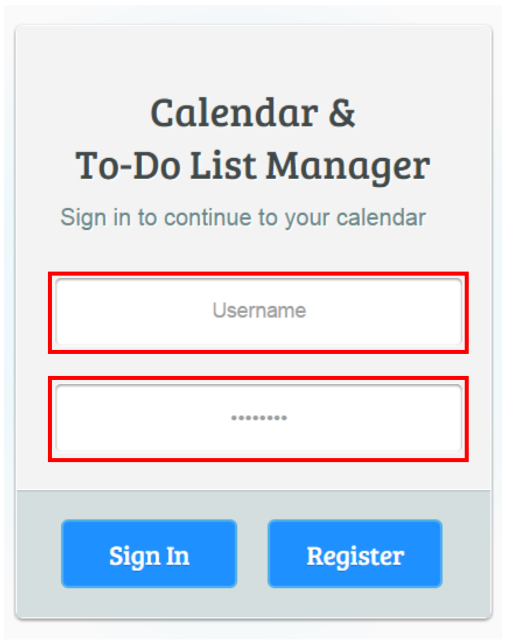


Figure 6 – Login Page with Prompting Text Fields for Username and Password

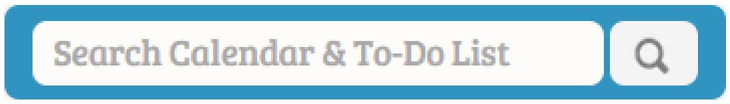


Figure 7 – Search Box as a Prompting Text Field

The second place we used the Prompting Text Field Pattern was in our page's search box as shown in figure 7. Since the search field was placed above the to-do list, there were concerns that some users may think that the search feature applied only to the to-do list and not the whole application. As such, we placed inside the search field the prompting text "Search Calendar & To-Do List" to inform the user of the search feature's full functionality. In addition to the intuitiveness improvements associated with the prompting text field for the search box, this approach also reduces the burden on a user's long term memory to remember the types of queries the search feature supports.

5.4. Modal Dialog Design Pattern

If a site requires a significant amount of navigation to access different features, then users may find the application difficult and/or confusing to use. Since not all users of our application will have very high frequencies of practice, we wanted to simplify the site's navigation model as much as possible.

One way to minimize the amount of page navigation is through modal dialogs, which open on top of an existing page. Through the use of modal dialogs, we were able to create a hub and spoke like navigation model where the user never actually leaves the main page (i.e. hub). Rather, the user accesses all other additional features through numerous modal dialogs (i.e. spokes). This approach will significantly reduce the risk when users explore different features because they know that by simply closing the modal dialog, they are able to return to where they just were. In addition, through the consistent use of modal dialogs, we expected that users would find the site's navigation model very predictable further improving the user experience.

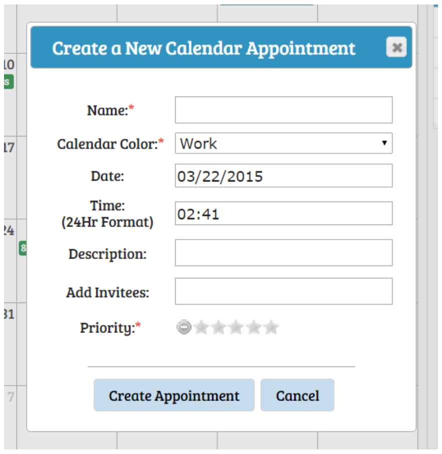


Figure 8 – Modal Dialog to Create a Calendar Appointment

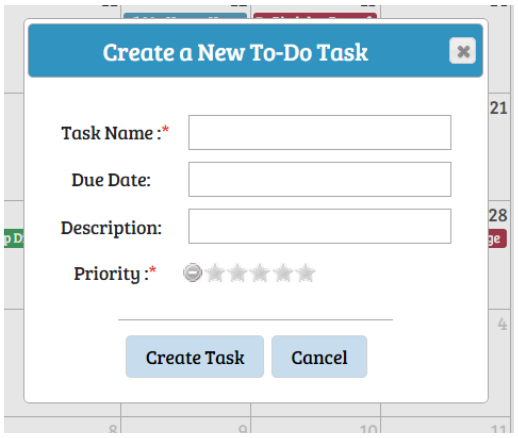


Figure 9 – Modal Dialog to Create a To-Do Task

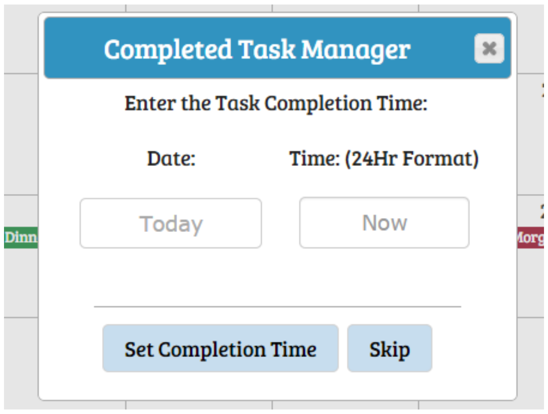


Figure 10 – Modal Dialog to Mark a To-Do Task Completed

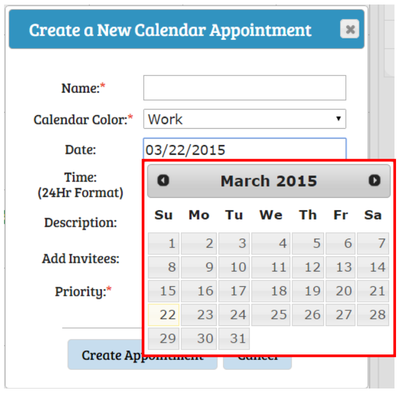
Specific examples where modal dialogs were used to provide users access to site features include: creating a calendar appointment, creating a to-do task, and marking a to-do task as completed; these modal dialog boxes are shown in figures 8, 9, and 10 respectively.

5.5. Drop Down Chooser Design Pattern

When a web page is not well-designed, users often become confused when trying to enter information into a form as they may be unsure of how to correctly format the input as well as the set of possible values the field accepts. In addition, when users are forced to manually type such information, they are more likely to make a mistake. To address these types of issues with user input, we used multiples drop down choosers in our application.

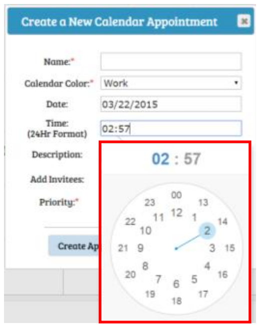
One type of drop down chooser used by our application is a date/calendar-style chooser where a user can select a specific day from a calendar month view. This type of drop down chooser is shown in figure 11 and was used in all three of our modal dialogs shown in figures 8, 9, and 10. One key added benefit of this chooser is that it eliminates the need for a user to recall the day of the week for a particular date. Instead, the calendar drop down chooser we used allows a user to quickly and visually recognize a particular date's day of the week reducing the likelihood of user error.

The second type of drop down chooser we utilized was a time chooser where the user could select a specific time that would be displayed in 24 hour formatting. The drop down chooser is shown in figure 12; it was used in the calendar appointment creation and to-do task completion modal dialogs shown figures 8 and 10.


Create a New Calendar Appointment

Name:*
Calendar Color:* Work
Date: 03/22/2015
Time: (24Hr Format)
Description:
Add Invitees:
Priority:*

Figure 11 – Calendar Date Selector Drop Down Chooser


Create a New Calendar Appointment

Name:*
Calendar Color:* Work
Date: 03/22/2015
Time: (24Hr Format) 02:57
Description:
Add Invitees:
Priority:*

Figure 12 – Time Selector Drop Down Chooser