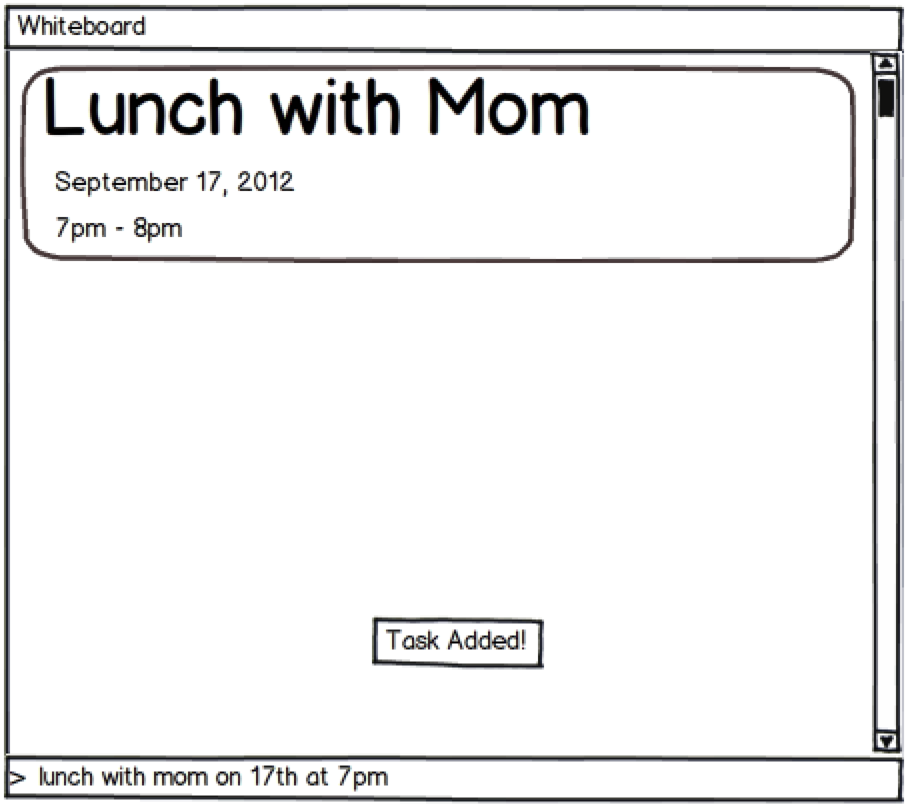
WHITE**BOARD**™



Whiteboard

Lunch with Mom

September 17, 2012

7pm - 8pm

Task Added!

> lunch with mom on 17th at 7pm

Salman Gadit Team lead, Developer, Bug fixer, External documentation	Varun Ganesh Developer, Tester, Product Documentation	Nizra Mohamed Nilufer Developer, Tester, Product Documentation	Saikrishnan Rangananthan Developer, Bug fixer, deadline watcher, External documentation

User Guide

Introducing Whiteboard

Of the many regular computer users out there, there is a subset that enjoys working only with their keyboard, but don't love to memorise too many technical-like commands. To users like this looking for a task management app or simply, a to-do list - WHITE**BOARD**™ is here. With capabilities of understanding near-human language (in a limited scope), a clean simple GUI and a SuperSleuth search, Whiteboard is geared for all the users who rely on to-do lists heavily in their daily tasks.

Functions

- Overview

WHITE**BOARD**™ comes with a host of simple functions with a powerful back-end. By taking in various known combinations of words used by people, the ease of adding, removing, editing and viewing tasks is simple and effective. In addition, using the power-search feature called SuperSleuth and the undo functionality, the use of the WHITE**BOARD**™ is simple, clean and easy.

- Adding tasks

With WHITE**BOARD**™, adding tasks is as simple as typing what you plan to do and when. Try entering "Lunch with mom at 7pm" or maybe "Hit the gym" and WHITE**BOARD**™ will add it to your list of things to do. It's that simple.

[Verb] BY/AT/ON/BEFORE [Time/Date] - *{Deadline task}*

[Verb] - *{Floating task}*

[Verb] FROM [Time/Date] TO/- [Time/Date] - *{Timed tasks}*

*Verb : Any user task description

- Editing Tasks

Tasks can be edited anytime, either the task description, the task date and/or time or the starting or/and ending times. Due to the simple one-line command feature editing tasks just becomes a whole lot easier with WHITE**BOARD**™. Simple, easy-to-follow, basic commands are given below:

MODIFY:/CHANGE:/UPDATE: [TaskID] TO [Verb] -*default description*

MODIFY:/CHANGE:/UPDATE: [TaskID] START [Time/Date] AND/OR END [Time/Date]

MODIFY:/CHANGE:/UPDATE: [TaskID] TO [Verb] START [Time/Date] AND/OR END [Time/Date]

- Viewing Tasks

Tasks can be viewed for the days and times the user requires. This includes tasks for the day, the week, a particular time/date, a particular period; or before a certain time/date; or even the entire archive. The one-step view feature contains a few simple basic commands which are given below:

VIEW: [Time/Date]

VIEW: ALL

VIEW: ARCHIVE

VIEW: FROM [Time/Date] TO/- [Time/Date]

VIEW: BY/AT/ON/BEFORE [Time/Date]

VIEW: WEEK –*displays all the tasks from Monday to Sunday of that week*

- Removing Tasks

Tasks can also be deleted or archived. The user simply needs to specify the Task ID of the task he/she wishes to remove or mark as done (i.e. archive)

DELETE:/REMOVE: “TASK ID” –*deletes the task completely*

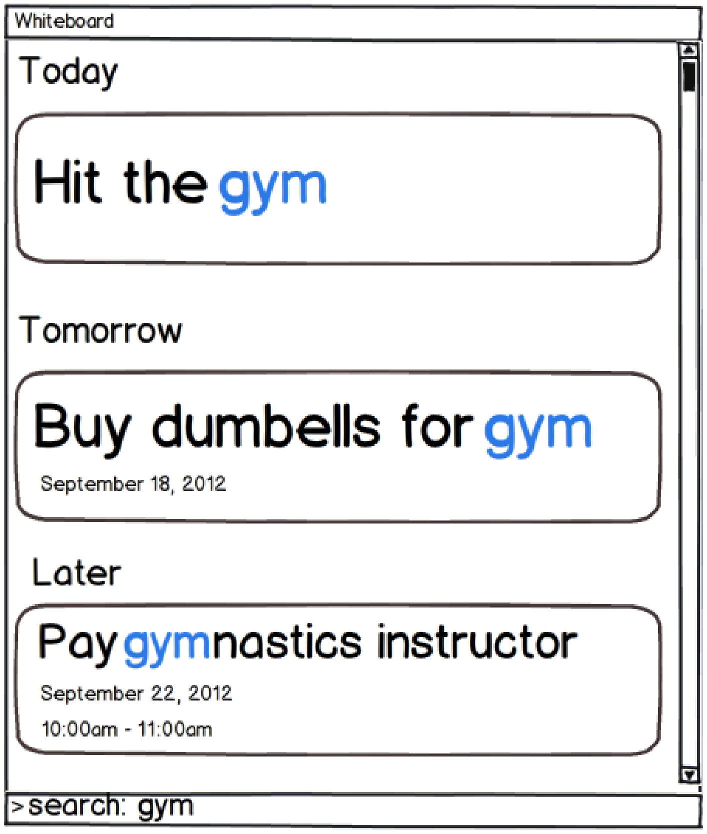
MARK: “TASK ID” as done –*removes task from list and archives it for future reference*

- SuperSleuth™ Search:

WHITE**BOARD**™ supports an intelligent search function. Using it is straightforward and intuitive. The user may search for any task just by typing into the search bar and the system will return all the tasks which have the corresponding letter, word or sentence contained in them.

The SuperSleuth search differs from normal command line search in its ability to automatically filter search results concurrently as the user types into the search bar. This allows the user to get to the particular task he/she wants quickly and efficiently.

Moreover SuperSleuth search has the ability to identify words within words. For example, please refer to the prototype diagram below.



Whiteboard

Today

Hit the gym

Tomorrow

Buy dumbells for gym

September 18, 2012

Later

Pay gymnastics instructor

September 22, 2012

10:00am - 11:00am

>search: gym

Figure 1: SuperSleuth Search

The user enters the word "gym" in the search bar and the system has returned results sorted in the order of earliest to ones that are due later. It is to be noted that a task with "gymnastics" has also been identified and retrieved.

- Undo:

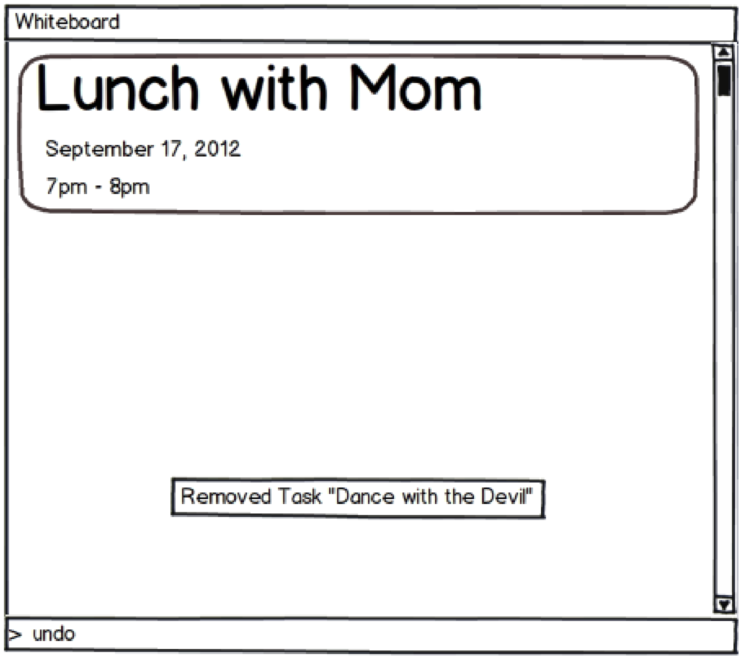
Command: undo:

WHITE**BOARD**™ supports a nifty 'undo' feature as well.

If you have made a mistake while entering a new task, simply pressing "CTRL+Z" on your keyboard will remove the newly added task

Similarly, the user need not panic upon unintentionally removing a task as pressing "CTRL+Z" will add the removed task back to its respective slot in the calendar.

Alternatively, the user can also type "Undo:" in the command line to perform an undo operation.



Removed Task "Dance with the Devil"

```
> undo
```

Figure 2: Undo operation

Developer Guide

Welcome to the team! As a new developer this guide will help you understand the architecture and design patterns that make WHITE**BOARD**™ happen.

Architecture

WHITE**BOARD**™ follows an MVC architecture where the view (our beautiful UI) has been designed ground up using WPF, a controller that mediates the input and a model in the form of an XML file that stores the task created by the user.

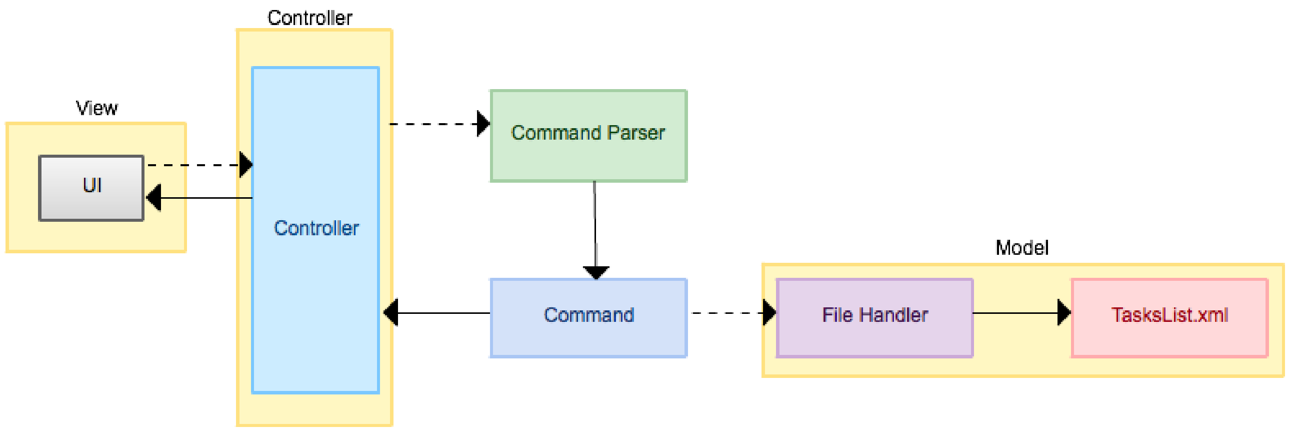


Figure 3: Whiteboard Architecture

Program Flow

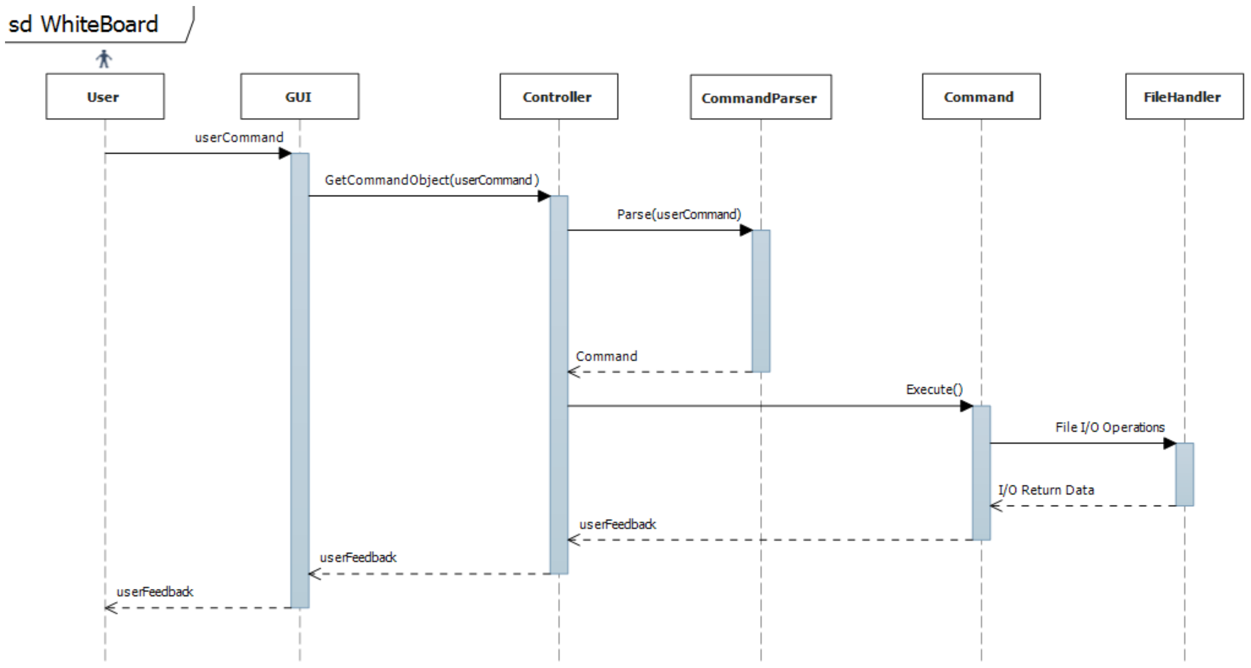


Figure 4: Sequence Diagram

WHITE**BOARD**™ takes in command from the user through its elegant UI and sends it over to the Controller which then calls Command Parser to process the natural language like command. The Command Parser decides which Command object to instantiate and returns it back to the Controller, the controller then called the abstract method Execute() on the Command which returns the feedback to be given back to the user. Thus the controller effectively mediates the input and creates a single point of entry and exit for the GUI.

Command and Task Objects

All user initiated actions in WHITE**BOARD**™ create Command objects that carry out tasks like adding a task, editing them, etc. All command objects inherit from an abstract Command class that provides an abstract method Execute() which contains the task specific implementation.

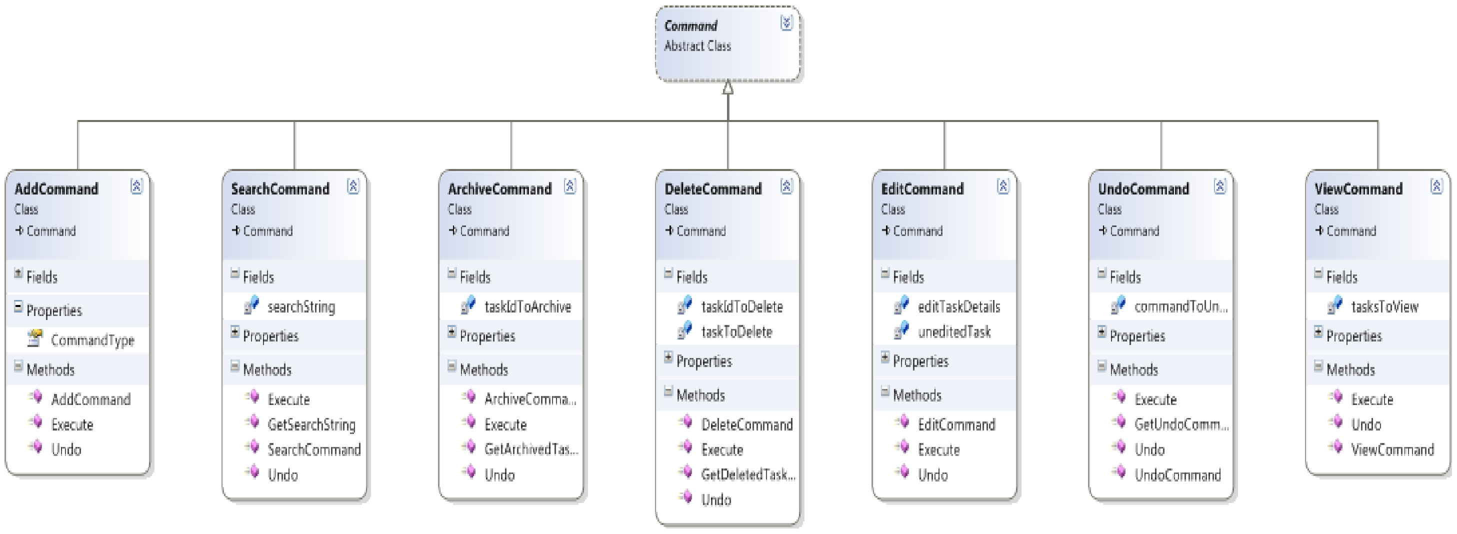


Figure 5: Command object

The Execute command returns the feedback in the form of a list of Task objects. The Task object contains all the details about the Task as seen in the class diagram representation below.

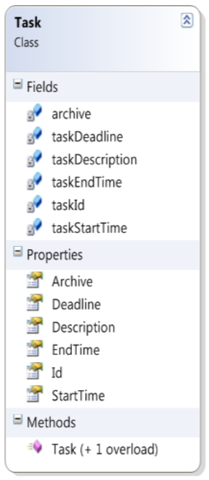


Figure 6: Task Object

UI

WHITE**BOARD**™ comes with a beautiful simple User Interface created in Windows Presentation Foundation (WPF 4.0), with no fancy interactions. It revolves around list of tasks and a text box to enter commands. Each listbox item is data-templated with labels for the different fields that a are contained in the Task object. This is illustrated in the code below.

```
<ListBox.ItemTemplate>
    <DataTemplate>
        <StackPanel Orientation="Vertical" Margin="5">
            <TextBlock FontSize="20" Text="{Binding Id}" />
            <TextBlock FontSize="20" Text="{Binding Description}" />
            <TextBlock Text="{Binding Deadline}"/>
            <TextBlock Text="{Binding StartTime}"/>
        </StackPanel>
    </DataTemplate>
 </ListBox.ItemTemplate>
```

We use data-binding to populate the list box with the required list of tasks. Data-binding makes it easy to bind a container on WPF with a List or Collection that implements the IEnumerable interface. Very simply, once the data context has been set to the list box, all it requires is a simple refresh to update the view.

Every action or command entered in the text box is responded with a small notification - called a *Toast* on the Android platform. Hence, we have also called it a toast notification.

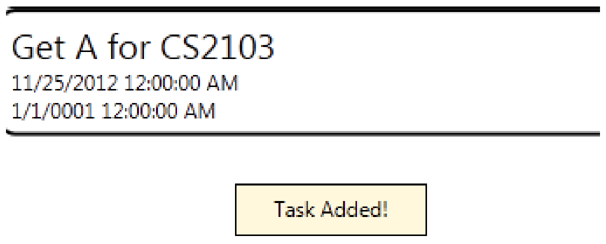

Get A for CS2103
11/25/2012 12:00:00 AM
1/1/0001 12:00:00 AM

Figure 7: Toast notification

The toast is implemented as a hidden label with dynamically adjustable width. When required, the text item for the toast is set after every command action and the toast label is made visible with an animation. The animation is timed control of it's Opacity, giving it the fade in and fade out effect like an actual toast notification.

Data Storage

Data Storage in WHITE**BOARD**™ is managed by the process of serialization. The list of tasks is stored into an XML file in the backend, by the process of serialization through C#. To retrieve the XML object and convert back into the list of Tasks, a process of deserialization is used. The file handler manages the actions of add, edit, remove, archive and delete. The general sequence of doing any of these tasks is:

1. De-serialize XML file into a List of Task objects
2. Perform required action on the list
3. Serialize and overwrite with the new XML file

Important Algorithms

Undo function

The undo functionality for WHITE**BOARD**™ is postulated under two expectations:

1. Every operation has a screen-state, i.e. whatever tasks that the user is seeing on screen. The operation can modify this screen state, so this screen state should be stored.
2. Every undo operation should, therefore, perform the *opposite* task that the previous command did, and at the same time time, restore the screen state to the previous version.

To make this work, we had to ensure that each command object stores the previous screen state by default. We also had to provide a unique behaviour for each command object's undo operation and therefore use the abstract method undo, which performs the undo task and returns the previous screen state. This result is executed via the UndoCommand object which then data binds the screen state to the main UI. All actions are stored on a global 'taskHistory' stack, and every undo operation retrieves the latest command object through that.

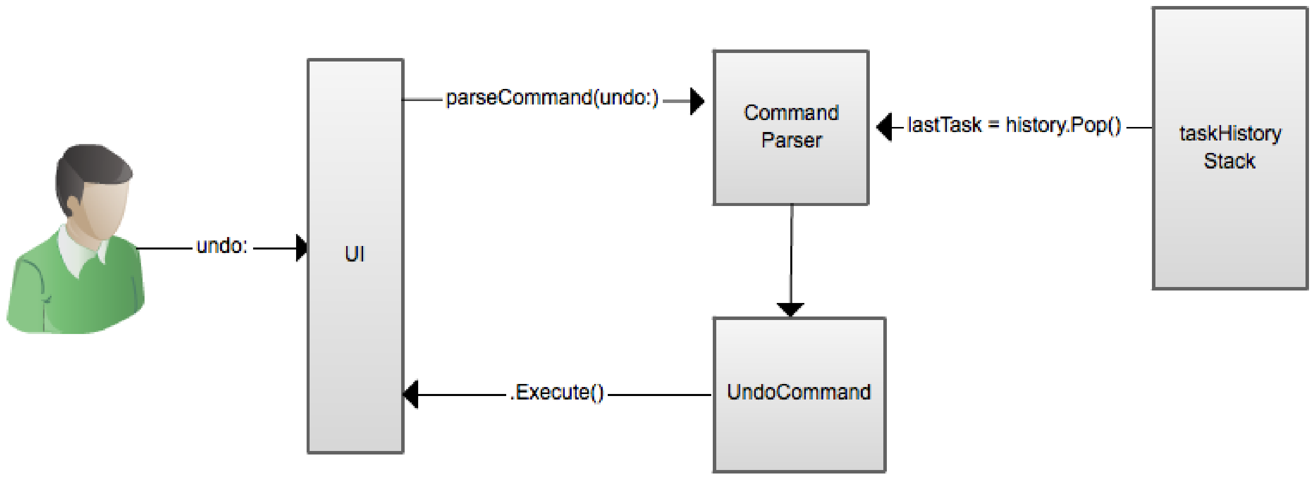


Figure 8: Undo operation