**Target audience & Purpose of developer guide**

This developer guide is meant for future developers or users, and it serves to familiarize readers such as yourself with the nuts and bolts of To-Do-This (TDT).

**Program description and Development requirements**

TDT is a simple and reusable desktop task manager which allows users to keep track of their daily tasks effectively. CRUD (Create, Read, Update, Delete) and functions such as search and undo are integrated with high flexibility, catering for a larger variety of user inputs. TDT is fully functional even without a mouse and no internet connection is required.

**Introduction to users**

Basic commands are available such as adding or deleting of tasks, display and even undo/redo operations. Colour coding is enabled such that you are able to differentiate between priorities, overdue and done. You can even create labels, and this will aid you in the grouping of tasks under the categories which you have specified. Graphical User Interface (GUI) is employed in TDT and this arms you with shortcuts that can ease the process of task management.

**Development environment**

Currently, the team is adopting Eclipse as the main platform for coding. Eclipse has the latest official support for Java 8 which includes Java development tools. This allows you as a developer to easily create or modify existing source code with the help of Quick Assist and Clean Up which are included in Eclipse.

**Challenges / Problematic areas**

Some challenges faced during the development phase include code committing on GitHub and the level of flexibility of user inputs. As a developer, you have the responsibility of maintaining and updating the latest code such that other developers are able to access them. However, GitHub has been sensitive in this area and many merge errors have emerged. Flexibility is also another issue which has not been tied down. It is impossible to cater to all forms of user input as TDT is unable to process natural language. Thus, there is a need to decide what the common forms are and how to decode them.

**Preface**

**Team vision / Team expectations**

*Productivity through basics. Organise and achieve. Great day, any day!*

Team Newbee strives to provide an alternative solution for users who focus on keyboard usage.

**Initial plan / Modifications**

We wanted to focus on the flexibility of TDT as it is our special feature. Previously commands had to be typed in full so that TDT would be able to understand. Fortunately, we decided to implement auto complete to aid users in typing commands. Currently, users are only required to key in the first two keywords of the command and TDT will complete the rest of it. Auto complete is also implemented for the editing of tasks and previously added task details will be displayed. For added flexibility, shortcuts have been created for commands such as undo, redo and several others. The initial idea to have a sort command was also removed, and the task manager is sorted simultaneously as tasks are added.

**How to launch**

Simply double click the To-Do-This java application located on your desktop.

**Product Design**

**Overall architecture**

This is the high-level architecture design of the components involved in To-Do-This. Under this section, you will understand how each component interacts with one another.

ToDoThis Commons

GUI

* GUI represents the graphical user interface seen by the user as he keys in his input.

Storage

Logic

* Logic consists of a Parser component.
* Parser decodes the user’s input and generates a Command object.
* Storage stores all tasks permanently in a text file called todothis.txt.
* Commons contain classes of static methods which are used by the other components.

GUI

Storage

TDTDateAndTime

Command

TDTParser

TDTLogic

Logic