Task Organizer

Projekt Programowanie II – Łukasz Tyszkiewicz, Justyna Kozik

# • 1.1 – Opis działania aplikacji

Task Organizer to prosta aplikacja typu to-do czyli lista zadań do zrobienia zbudowaną na bazie QWidgetów czyli przy pomocy narzędzia QtCreator.

Oferuje ona możliwość dodania własnego zadania, ustalenia terminu wykonania zadania na bazie kalendarza, dodania opisu, oraz ustalenie czy zadanie zostało wykonane. A wszystko w przejrzystym, minimalistycznym interfejsie graficznym.

**Program zapisuje dane tylko do momentu jego zamknięcia. Po ponownum uruchomieniu aplikacji, zapisane zadania zostają porzucone, jest to błąd spowodowany użyciem struktury QWidget zamiast QAbstractItem, co powoduje komplikacje związane z odczytywaniem struktury z bazy danych**

*Wymagania sprzętowe:*

* 50 MB wolnego miejsca na dysku
* 15 MB RAM
* System 64 bitowy

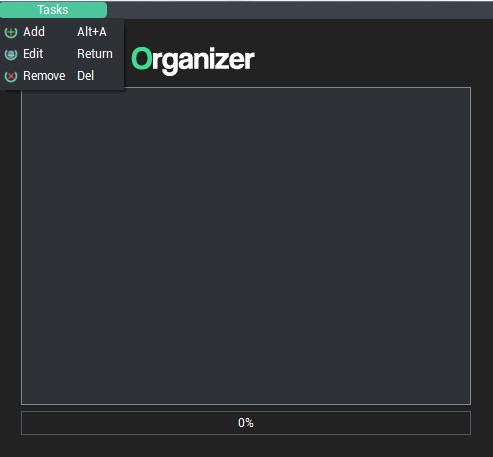
*Wymagania ogólne:*

* Folder **release-version** z plikami **.dll** oraz plikiem wykonywalnym **task organizer alpha release.exe**
* OPCJONALNE: Zainstalowany QtCreator w wersji 4.30

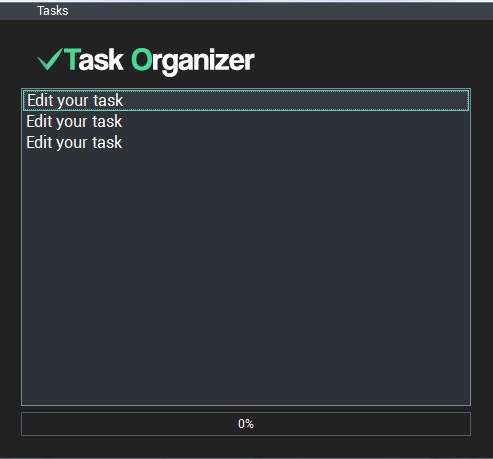
1.1 – Opis działania aplikacji  1

# • Część 1.2 – Instrukcja obsługi aplikacji

Aby dodać zadanie możemy skorzystać z opcji Add w zakładce Tasks, lub skrótu klawiszowego Alt + A.



Aby edytować zadanie należy dodane zadanie zaznaczyć przy pomocy Lewego Przycisku Myszki lub klawisza Tab, i wybrać z menu znajdującego się w lewym górnym rogu opcje Edit, kliknąć podwójnie w dane zadanie przy pomocy Lewego Przycisku Myszkilub wcisnąć klawisz Enter.

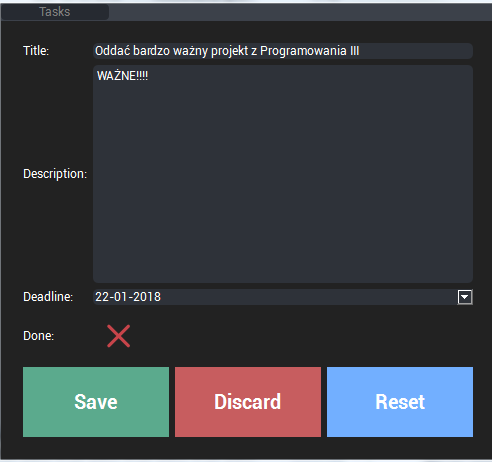


Część 1.2 – Instrukcja obsługi aplikacji  2

# • Część 1.2 – Instrukcja obsługi aplikacji

Aby usunąć zadanie należy wybrać zadanie i skorzystać z opcji Remove **l**ub klawisza Delete

Kolejną częścią jest edycja zadania, w kolejnym oknie edycja jest podzielona na 5 części



**Title** – Tytuł zadania

**Description** – Opis zadania

**Deadline** – Termin ostateczny zadania

**Done** – Checkbox do zaznaczania czy zadanie zostało spełnione

**Save, Discard, Reset Buttons** – Przyciski pełniące odpowiednie funkcje.

Część 1.2 – Instrukcja obsługi aplikacji  3

# • 2.1 – Podział projektu na pliki

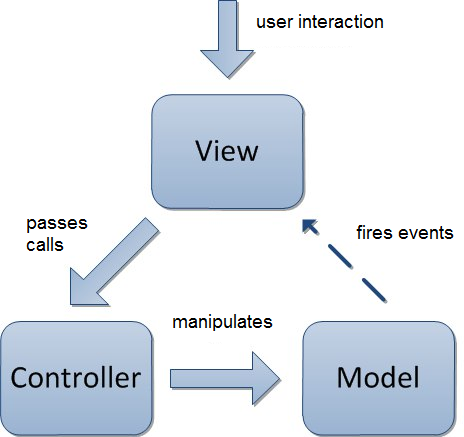
images.qrc – plik z zasobami mainwindow.ui – plik XML odpowiadający za UX i UI main.cpp – główne komendy odpowiadające za komunikacje między klasami

mainwindow.cpp (**View**) – metody odpowiadające za wyświetlanie zawartości logicznej w aplikacji oraz wprowadzanie do logiki programu argumentów

singletask.cpp (**Model**) – zawiera metody klasy dla pojedynczego zadania, zawiera metody setter i getter

tasklist.cpp (**Model**) – zawiera metody klasy dla kontenera typu QList składającego się z pojedynczych zadań, zawiera metody odpowiadające za tworzenie logiki projektu jak dodawanie zadania do listy, usuwanie go oraz zliczanie ilości wykonanych zadań

tasklistcontroller.cpp (**Controller)** – odpowiada za komunikacje pomiędzy tasklist (**Model**) a mainwindow (**View**), metody zawarte w tej klasie ułatwiają rozbudowę programu i zwiększają jego czytelność. \*.h – Definicje klas, zmienne oraz prototypy metod



2.1 – Podział projektu na pliki  4