**Конвертор на мерни единици**

Курсова работа по Програмни Среди



**Разработил:**

Валентин Георгиев Александров

факултетен номер: 121216074

гр. 46.

**Задание:**

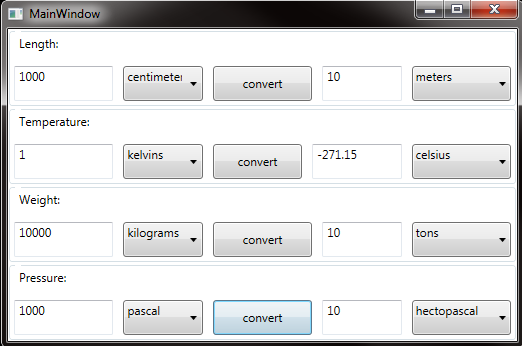
8. Направете програма която да превръща от едни мерни единици в други.

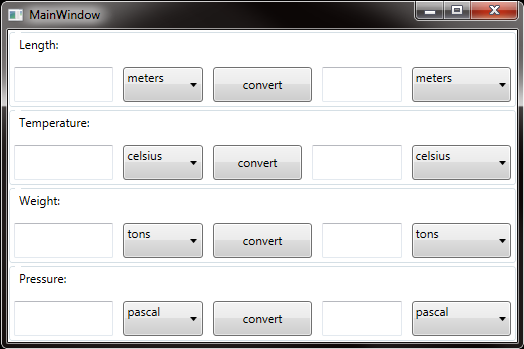
Реализирайте преобразуване на температурните мерки (Целзий, Фаренхайт,

Келвин), Дължина (метри, футове, мили,...), Маса (кг, грам, паунд, унция, ...) и

други.

**Дизайн и демонстрация на работа**





**Приложение на софтуера:**

Програмата получава като входни данни стойности, които са под формата на рационални числа. Интерфейсът е направен с основна цел да бъде ясен и лесен за ползване. За всяка мерна единица има специално поле, в което се въвежда информацията. Потребителя може да въвежда за:

* Дължина
* Температура
* Тегло
* Налягане

До полето, в което въвежда входната стойност има падащо меню, от което потребителя може да си избере размерността на мерната единица.

Примерно за температура може да избере едно то следните:

* Целзий
* Феранхайт
* Келвин

И веднега след това е бутона за конвертиране.

Ако потребителя в момента го цъкне, то просто ще изведе същата стойност. Продуктът е предвиден за хора, които се нуждаят от конвертор, а не просто да си цъкат върху него. Съвесем интуетивно от дясната страна имаме още едно падащо меню, от което потребителя, може да си избере в каква размерност на мерната единица би желал въведената от него стойност да бъде конвертирана.

Бизнес логиката е реализиране чрез използване на следната ООП архитектура:

Имаме абстрактен клас - Converter, който има един абстрактен метод,

* **double convert(String from, String to, double value)**

Основната идея е че какъвто и конвертор да пишем, то той ще има

* От каква размерност.
* В каква размерност.
* Входна стойност.

Този клас има 4 наследника(по един за всяка физична величина, за която ще пишем как се конвертира).

При наследяване всеки наследник е задължен да имплементира абстракция метод **convert**, като му зададе тяло, как точно ще работи. Използвайки този подход, постигам Single Responsibility(един от SOLID принципите) - на кратко всеки клас да прави само едно нещо.

Няма смисъл класа Converter, да може да конвертира всичките мерни единици. А и с този подход постигам още едно приимущество:

* **Надграждане.**

Ако искам да добавя още функционалности(като например да конвертирам повече физични величини), то просто пиша още един клас, който ще имплементира абстрактния клас.

**MainWindow.xaml.cs** използва композиция и държи като private член променливи всяка имплементация, но използва полиморфизъм и всяка ще бъде декларирана като **Converter** за тип на данната. Имаме четири action listeners и всеки при активирането си извиква метода convert, на съответния конвертор.

Ако искаме надграждане, просто ще имаме още бутони и всеки ще извиква съответния си клас с неговата версия на convert метода.

**Силни страни на проекта:**

* Добра ООП структура.
* Лесен за поддръжка.
* Удобен за надграждане.
* Ясен, точен и лесен за ползване графичен интерфейс.