# Опис програмної частини

Програмний код можна поділити на 3 частини:

* Головний програмний код
* Класи генерування візерунку
* Допоміжні класи, які використовуються для парсингу файлів налаштувань

# Головний програмний код

При ініціалізації, програма завантажує з файлу letters.txt інформацію про всі літери українського алфавіту та інформацію про їх форматування у орнаментах. Також, з папки ornaments, зчитуються всі файли типу “json”, в яких міститься інформація про всі додані візерунки вишиванок. В папці shirts зберігається інформація про типи рубашок в такому самому форматі. Всі дані десеріалізуються у об’єкти відповідних класів. Після того, як користувач обрав необхідні параметри та натиснув на кнопку, викликається функція створення візерунку. Зчитавши введені користувачем дані про ім’я, програма перевіряє введені дані на валідність, створює відповідний клас та передає в нього введене ім’я. По цим даним, клас створює масив точок та їх кольорів для малювання візерунку. Далі, програма виконуючи бібліотеку GDI малює відповідні точки та створює файл, який містить згенерований візерунок разом з сіткою для майбутнього використання. Далі, програма користуючись даними від користувача зчитує відповідний малюнок базового орнаменту та інформацію про нього та малює на ньому згенерований візерунок. Відповідно до цього створюється і замальовка рубахи з потрібним орнаментом. Далі всі малюнки зберігаються до папки користувача та виводяться у програмі в зручному вигляді. Для роботи, програма потребує .NET Framework v. 4.0, бібліотеку Newtonsoft.JsonConverter для роботи з файлами налаштувань та стандартну бібліотеку ОС Windows GDI+ для створення малюнків.

# Класи генерування візерунку

Для генерування візерунків створено три класи: Point, Letter та Board. Клас Point(насправді це структура) зберігає інформацію про одну конкретну точку – її позиція та колір.

Клас Letter зберігає у собі інформацію про всі символи українського алфавіту, точки для відмальовки та методи для їх позиціонування у візерунку в потрібній формі.

Клас Board використовується для опрацювання ведених слів та для групування точок символів у єдине ціле. З точки зору програмування, клас використовує власноруч написаний алгоритм розміщення точок якнайближче до центру, щоб візерунок приймав більш кучніший вигляд.

Детальний опис структури класів представлено нижче:

* Point
  + int X – змінна, що приймає значення зміщення по осі Х;
  + int Y – змінна, що приймає значення зміщення по осі Y;
  + int Color – змінна, що приймає значення кольору точки;
* Letter
  + char Symbol – змінна, що приймає значення символу для цього об’єкту;
  + string[] VerticalPoints – масив, що зберігає інформацію про позиції точок для цього символа при вертикальному розміщенні;
  + string[] DiagonalPoints – масив, що зберігає інформацію про позиції точок для цього символа при діагональному розміщенні;
  + List<Point> getVPoints() – метод, що повертає список точок, які використовуються для малювання цього символу при вертикальному розміщенні, враховуючи зміщення на дошці;
  + List<Point> getDPoints() – метод, що повертає список точок, які використовуються для малювання цього символу при діагональному розміщенні, враховуючи зміщення на дошці;
* Board
  + List<Point> points – список, що містить точки для малювання очікуваного слова;
  + List<Letter> letters – список, що містить інформацію, про всі відомі програмі символи;
  + int lastColor – змінна, що зберігає стан останнього кольору;
  + int min – службова змінна, що використовується для оптимізації пошуку позиції;
  + int maxValue – службова змінна, що використовується для оптимізації пошуку позиції;
  + string word – змінна, що зберігає стан намальованого слова;
  + void Clear() – метод, що використовується для очищення поточного поля;
  + bool addLetter() – метод, що додає конкретну букву на поточне поле;
  + void setWord() – метод, що змінює слово поточного поля та перегенерує усі точки поля;

# Допоміжні класи, які використовуються для парсингу файлів налаштувань

Для роботи з бібліотекою Newtonsoft.JsonConverter потрібно було створити допоміжні класи, які б використовувались для парсингу файлі конфігурації.

Для файлів з орнаментами було створено два основних класів та один допоміжний. Клас OrnamentFounding що містить базову інформацію про орнаменти, клас OrnamentPlace що зберігає інформацію про місця розташування згенерованих візерунків на орнаменті, та допоміжний клас ColorO який зберігає інформацію про колір.

Для файлів з рубахами було створено два основних класи. Клас Shirt містить базову інформацію про рубаху. Клас Region зберігає інформацію про регіони на рубахах, на яких потрібно малювати створений візерунок.