Форма № 17

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Кафедра програмних засобів
(найменування кафедри)

КУРСОВИЙ ПРОЕКТ (РОБОТА)

з дисципліни «Операційні системи»
(назва дисципліни)

на тему: «Розробка програми для захоплення екрану»

Студента (ки) 3 курсу групи КНТ-137
Спеціальності Інженерія програмного забезпечення
Козлова В. В.
(прізвище та ініціали)

Керівник доцент, доцент, к.т.н.,
Сердюк С. М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Національна шкала _____

Кількість балів: _____ Оцінка: ECTS _____

Члени комісії _____ Сердюк С. М.
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

_____ _____
(підпис) (прізвище та ініціали)

2020 рік

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Національний університет «Запорізька політехніка»
(повне найменування вищого навчального закладу)

Інститут, факультет інститут інформатики та радіоелектроніки, факультет комп'ютерних наук і технологій

Кафедра програмних засобів

Ступінь вищої освіти (освітній ступінь) бакалавр

Спеціальність 121 «Інженерія програмного забезпечення»

Курс 3 Група КНТ-137 Семестр V

ЗАВДАННЯ

на курсовий проект студентові

Козлову Валерію Валентиновичу
(прізвище, ім'я, по батькові)

1. Тема проекту: Розробка програми для захоплення екрану
2. Термін здачі студентами закінченого проекту: 10 червня 2020 року
3. Початкові дані до проекту: Всі дані, надані операційною системою
4. Зміст розрахунково пояснювальної записки:

Вступ

1 Аналіз предметної області

2 Аналіз програмних засобів

3 Розробка й реалізація компонентів системи

4 Керівництво програміста

5 Керівництво оператора

Висновки

Перелік посилань

Додаток А Тексти програми

5. Дата видачі завдання: 10 лютого 2020 року

КАЛЕНДАРНИЙ ПЛАН

№	Назви етапів курсового проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1.	Отримання та аналіз індивідуального завдання	1 - 2 тиждень	
2.	Аналіз програмних засобів, що будуть використовуватись в роботі	3 - 4 тиждень	
3.	Розробка логічної моделі системи. Заповнення даними	4 - 5 тиждень	
4.	Вивчення можливостей програмної реалізації інтерфейсу користувача	5 - 6 тиждень	
5.	Проектування додатків системи	6 - 8 тиждень	
6.	Оформлення відповідних пунктів пояснювальної записки	9 тиждень	Розділи 1 - 5 ПЗ
7.	Захист курсової роботи	10 тиждень	

Студент _____ Козлов В. В.

Керівник _____ Сердюк С. М.

«_____» _____ 2020 р.

РЕФЕРАТ

ПЗ: 140 с., 23 рис., 1 додаток, 10 посилань.

Об’єкт проектування – програма для захоплення зображення й відео з екрану комп’ютера та вбудований графічний редактор.

Область дослідження – функції операційної системи й можливості мови програмування C#.

Тема курсового проекту – розробка програми для захоплення екрану.

Мета курсового проекту – створення програми з мінімалістичним й інтуїтивно зрозумілим інтерфейсом, що дозволяє користувачу захоплювати зображення (довільна область або увесь екран, одноразово або з визначеним інтервалом) й відео з екрану комп’ютера, редагувати отримані зображення та зберігати їх.

Розроблена програма відповідає таким критеріям:

- мінімалістичний й інтуїтивно зрозумілий інтерфейс;
- можливість захоплення зображення з довільної області або всього екрану;
- можливість захоплення скріншотів з визначеним інтервалом часу;
- можливість графічної обробки скріншоту;
- можливість запису відео з екрану;
- можливість зміни налаштувань програми (інтервал, формати зображень, папки для збереження, сполучення клавіш).

Програму було розроблено за допомогою інтегрованого середовища розробки Microsoft Visual Studio 2019, з використанням таких технологій, як .NET Framework 4.7.2, C# 7.3 та патерн проектування Command.

Програма призначена для роботи на пристроях під керуванням операційної системи Windows 10.

SCREEN RECORDER, GRAPHICS EDITOR, SCREENSHOT, C#, .NET FRAMEWORK, VISUAL STUDIO, COMMAND, WINDOWS.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ І СКОРОЧЕНЬ

ПЗ – пояснювальна записка

ТЗ – технічне завдання

ОС – операційна система

GNU GPL – GNU General Public License

IDE – Integrated development environment

CLR – Common Language Runtime

CIL – Common Intermediate Language

MSIL – Microsoft Intermediate Language

ISO – International Organization for Standardization

ECMA – European Computer Manufacturers Association

FCL – Framework Class Library

API – Application Programming Interface

ЗМІСТ

ВСТУП

З кожним днем інформаційні технології, що з успіхом впроваджуються в повсякденне життя сучасної людини, стають її невід'ємною частиною. А комп'ютери, які раніше використовали тільки на промислових виробництах, тепер застосовують в різних сферах діяльності: від стартапів та бізнес проектів до перегляду фільмів та геймінгу.

При повсякденному використанні комп'ютера в якості основного пристрою для роботи чи розваг інколи виникає необхідність збереження отриманих результатів та представлення їх громадськості. Найпростіший спосіб комунікації інформації – це візуально показати те, що необхідно донести іншій людині. Саме тому серед користувачів персональних комп'ютерів набувають великої популярності програми для захоплення зображень й відео з екрану.

За допомогою таких програм можна робити скрішоти екрану та робочої області, редагувати їх та наносити різні графічні об'єкти, записувати відео з власними коментарями, створювати навчальні відеоуроки й захоплювати процес проходження комп'ютерних ігор.

Створення таких програм – це неймовірно цікавий та корисний досвід не тільки для новачка в програмуванні, а і для фахового спеціаліста. Адже при написанні програми для захоплення зображення й відео з екрану комп'ютера необхідно не тільки досконало знати використовувану мову програмування, а й устрій операційної системи та супутні технології для обробки зображень і відеозаписів.

Саме тому було прийнято рішення створити вільне програмне забезпечення, яке дозволяло би виконувати вищезазначені функції, та могло би використовуватися в якості незамінного інструменту для створення навчальних матеріалів та відеокурсів для учнів шкіл і студентів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд існуючих аналогів програмного засобу

Перед початком розробки необхідно дослідити доступні програми на ринку, що виконують аналогічні функції. Проаналізувати їх переваги та недоліки й виділити основні технічні та програмні властивості, якими має володіти створюване програмне забезпечення.

1.1.1 Ножиці

«Ножиці» – це стандартна програма для створення знімків екрану, яка вбудована в ОС Windows 10, Windows 8 та Windows 7 [1]. Інтерфейс програми наведено на рисунку 1.1.

«Ножиці» надають можливість робити знімок вікна або всього екрану, обирати область прямокутної або довільної форми, встановлювати затримку для створення скріншоту. На нього можна наносити графічні об’єкти за допомогою таких інструментів, як маркер, перо та гумка. Знімок можна зберегти у вигляді зображення (PNG, GIF або JPEG формату) або веб-сторінки, також є можливість надіслати знімок електронною поштою.

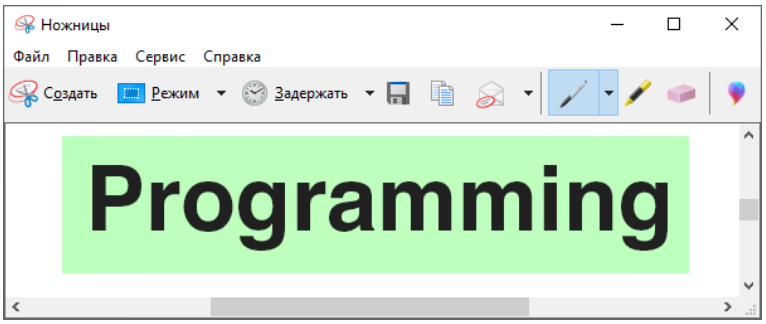

Рисунок 1.1 – Інтерфейс програми «Ножиці»

Умови розповсюдження програми:

- безкоштовне (freeware) програмне забезпечення;

- поставляється разом з операційною системою сімейства Windows;

Серед переваг програми можна виділити такі:

- стандартна програма, яка вбудована в операційну систему;
- простий та інтуїтивно зрозумілий інтерфейс;
- наявність декількох режимів створення знімку;
- можливість графічного редагування скріншоту;
- можливість збереження знімку в різних форматах.

Серед недоліків програми можна виділити такі:

- малий вибір інструментів для графічного редагування;
- відсутність можливості створення інтервальних знімків;
- відсутність можливості захвату відео з екрану;
- відсутність підтримки гарячих клавіш;
- відсутність курсора миші на знімку.

1.1.2 Screenshot Captor

«Screenshot Captor» – це вільно розповсюджувана програма для створення скріншотів в ОС Windows [2]. Інтерфейс програми наведено на рисунку 1.2.

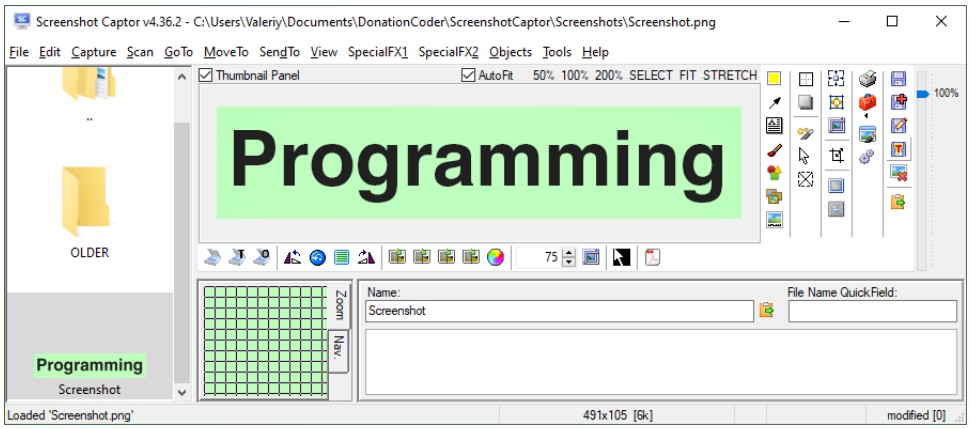

Рисунок 1.2 – Інтерфейс програми «Screenshot Captor»

«Screenshot Captor» надає можливість створення знімків екрану за допомогою різних режимів з використанням налаштовуваних користувачем

сполучень клавіш. Особливістю програми є вбудований графічний редактор, що дозволяє змінювати отриманий скріншот й наносити на нього деякі графічні об’єкти. Знімок можна зберегти у вигляді зображень різних форматів.

Умови розповсюдження програми:

- безкоштовне програмне забезпечення;
- тип ліцензії – Freeware;

Серед переваг програми можна виділити такі:

- вільно розповсюджувана програма;
- наявність декількох режимів створення знімку;
- можливість графічного редагування скріншоту;
- можливість збереження знімку в різних форматах;
- можливість детального налаштування програми;
- підтримка налаштовуваних гарячих клавіш.

Серед недоліків програми можна виділити такі:

- складний користувацький інтерфейс;
- застарілий дизайн;
- відсутність можливості створення інтервальних знімків;
- відсутність можливості захвату відео з екрану;
- відсутність курсора миші на знімку.

1.1.3 LightShot

«LightShot» – це вільно розповсюджувана програма для захоплення екрану в ОС Windows, що працює схожим на «Ножиці» чином, та надає додаткові можливості по редагуванню зображення і його публікації в мережі інтернет [3]. Інтерфейс програми наведено на рисунку 1.3.

«LightShot» дозволяє робити швидкий знімок виділеної області екрана й редагувати його за допомогою вбудованих інструментів для нанесення графічних об’єктів. Програма надає можливість збереження відредагованого

знімку у вигляді зображень різних форматів та легкої публікації в мережі інтернет на різних сервісах.

Рисунок 1.3 – Інтерфейс програми «LightShot»

Умови розповсюдження програми:

- безкоштовне програмне забезпечення;
- тип ліцензії – Freeware;

Серед переваг програми можна виділити такі:

- вільно розповсюджувана програма;
- простий та інтуїтивно зрозумілий інтерфейс;
- можливість графічного редагування скріншоту;
- можливість збереження знімку в різних форматах.

Серед недоліків програми можна виділити такі:

- наявність тільки одного режиму створення знімку;
- відсутність можливості створення інтервальних знімків;
- відсутність можливості захвату відео з екрану;
- відсутність курсора миші на знімку.

1.1.4 Bandicam

«Bandicam» – це компактне пропрієтарне програмне забезпечення для запису екрану в ОС Windows в форматі високоякісного відео [4]. Інтерфейс програми наведено на рисунку 1.4.

«Bandicam» дозволяє захоплювати зображення й відео з екрану комп’ютера за допомогою різних режимів (увесь екран, активне вікно або

виділена область). Разом з відео записується і звукова дорожка. Програма підтримує гарячі клавіши й може зберігати отримані зображення й відео в різних форматах.

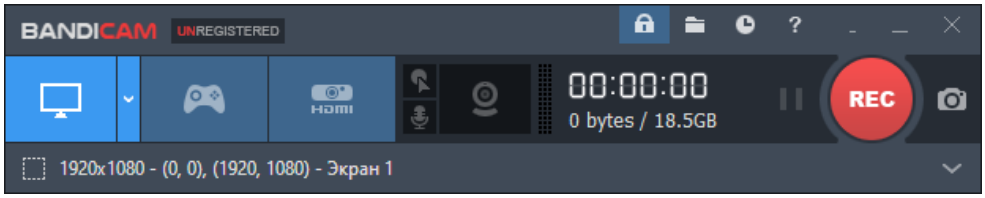

Рисунок 1.4 – Інтерфейс програми «Bandicam»

Умови розповсюдження програми:

- умовно безкоштовне (пропрієтарне) програмне забезпечення (trial версія);

- вартість ліцензії – 39 доларів США;

- тип ліцензії – Shareware;

Серед переваг програми можна виділити такі:

- простий та інтуїтивно зрозумілий інтерфейс;

- наявність декількох режимів захоплення екрану;

- можливість збереження знімків та відео в різних форматах

- підтримка налаштовуваних гарячих клавіш.

- можливості захвату відео з екрану;

Серед недоліків програми можна виділити такі:

- пропрієтарне програмне забезпечення;

- відсутність можливості графічного редагування зображень;

- відсутність можливості створення інтервальних знімків;

- відсутність курсора миші на знімках.

1.1.5 OBS Studio

«OBS Studio» – вільне програмне забезпечення з відкритим вихідним кодом для запису відео й потокового мовлення, що розробляється проектом

OBS і співтовариством незалежних розробників [5]. Інтерфейс програми наведено на рисунку 1.5.

«OBS Studio» надає можливість перехоплення зображень з пристроїв і джерел в реальному часі, композиції сцен, декодування, запису й мовлення. Програма містить детальні налаштування й підтримку сполучень клавіш, що допомагають спростити її використання.

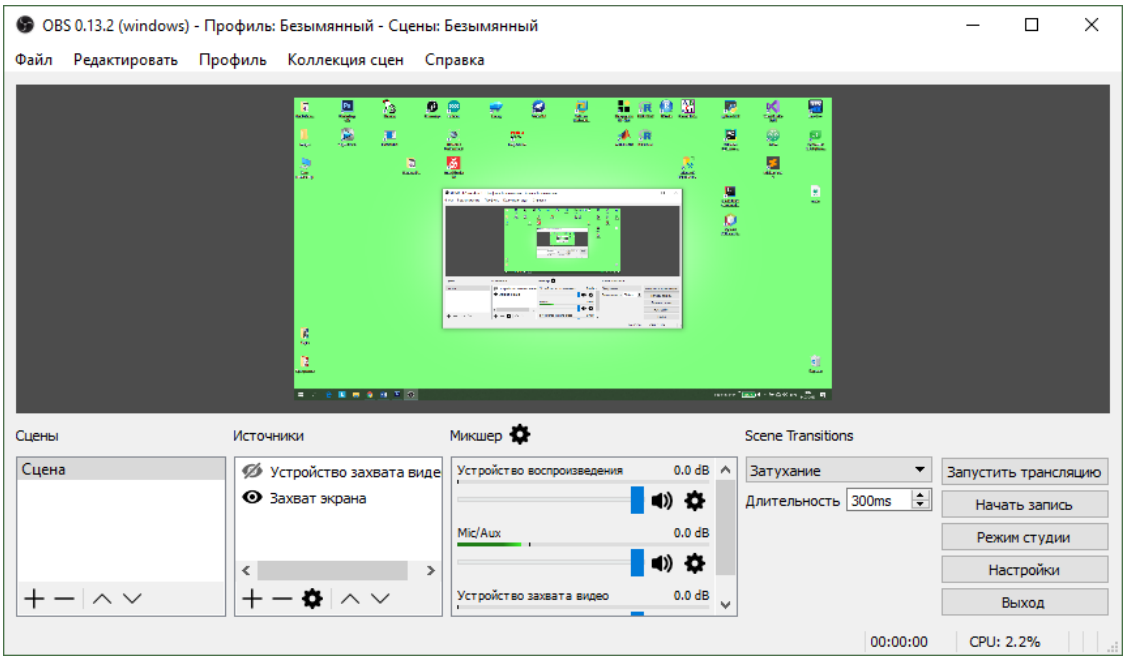

Рисунок 1.5 – Інтерфейс програми «OBS Studio»

Умови розповсюдження програми:

- вільне програмне забезпечення з відкритим вихідним кодом;
- тип ліцензії – GNU GPL;

Серед переваг програми можна виділити такі:

- вільно розповсюджувана програма;
- простий та інтуїтивно зрозумілий інтерфейс;
- наявність декількох режимів захоплення екрану;
- можливість детального налаштування програми;
- підтримка налаштовуваних гарячих клавіш.
- можливості захвату відео з екрану;

Серед недоліків програми можна виділити такі:

- відсутність можливості захоплення зображення з екрану;
- відсутність можливості графічного редагування зображень;
- відсутність можливості створення інтервальних знімків;

1.1.6 Підсумки огляду аналогів

В результаті проведення аналізу існуючих аналогів розроблюваного програмного забезпечення було визначено, що жодна з існуючих програм не реалізує в повній мірі усіх функцій, що вимагаються. Існуючі аналоги пропонують можливості захоплення або зображень, або відео з екрану при цьому обмежують функціонал або умови розповсюдження програми.

Програма розроблюється в якості вільно розповсюджуваного програмного забезпечення, об’єднує всі переваги існуючих аналогів, а отже її розробка є доцільною.

1.2 Аналіз цільової аудиторії

Велика кількість компаній, використовує інформаційні технології не тільки для організації й упорядкування даних, а й для контролю та регулювання виконання робочих задач співробітниками. За допомогою можливості захоплення зображень й відео з екранів комп’ютерів цей процес стає менш складним й трудомістким.

Такі моливості можуть бути необхідними не тільки для демонстраціїї ходу роботи та звітування про досягнуті результати, а й для швидкої комунікаціїї та докладних пояснень і демонстрації особливостей використання функцій комп’ютера та програмного забезпечення.

Функції захоплення екрану необхідні як для офісного співробітника, який займається повсякденною роботою й демонструє результати її виконання своєму керівництву, так і для звичайного користувача персонального

комп’ютера, який хоче поділитися інформацією, що відображено на його екрані. Також використання таких можливостей відкриває нові простори для викладачів, які можуть готувати докладні методичні матеріали з ілюстраціями, що демонструють хід виконання робіт,й детальні онлайн відеокурси з різних наукових напрямків та дисциплін.

Згідно з проведеним аналізом, стає зрозумілим те, що цільовою аудиторією такого програмного забезпечення є всі користувачі персональних комп’ютерів, які мають на меті зберегти й продемонструвати результати своєї роботи за комп’ютером іншим людям.

1.3 Технічне завдання

Найважливішим документом, який встановлює основне призначення, показники якості, техніко-економічні та спеціальні вимоги до розроблюваного програмного забезпечення, його обсягу, стадій розробки та складу конструкторської документації є технічне завдання, яке представлено в цьому розділі.

1.3.1 Підстава для розробки

Підставою для розробки є завдання на курсовий проект з дисципліни «Операційні системи» на тему «Розробка програми для захоплення екрану», затверджене науковим керівником.

1.3.2 Мета та призначення розробки

Метою створення програмного забезпечення є розробка програми з інтуїтивно зрозумілим інтерфейсом, що дозволяє користувачу захоплювати зображення (довільна область або увесь екран, одноразово або з визначеним

інтервалом) й відео з екрану комп’ютера, редагувати отримані зображення та зберігати їх.

Призначення розробки – спростити процес створення зображень й відеозаписів екрану комп’ютера для звичайного користувача.

1.3.3 Вимоги до системи, що розробляється

Програма має працювати під керівництвом ОС Windows 10 та .NET Framework версії 4.7.2 чи новіше. В якості інструменту розробки має бути використано мову програмування C# версії 7.3 чи новіше та інтегроване середовище розробки Microsoft Visual Studio 2019 чи новіше.

Програмне забезпечення має складатися з таких частин:

- VarietyScreenRecorder – утиліта для захоплення зображення й відео з екрану комп’ютера;

- GraphicsEditor – графічний редактор для редагування зроблених скріншотів.

VarietyScreenRecorder має надавати такі можливості:

- захоплення зображення з усього екрану комп’ютера;

- захоплення зображення з виділеної прямокутної області на екрані;

- редагування отриманих зображень за допомогою графічного редактора;

- захоплення зображень з екрану з заданим інтервалом;

- захоплення відео з екрану комп’ютера;

- налаштування папок для збереження скріншотів та їх форматів (png, jpeg, bmp, gif), інтервалу збереження, папки для збереження відеозапису екрану;

- перевизначення використовуваних гарячих клавіш;

- автоматичне збереження скріншоту й відео в задану папку з визначеним форматом та копіювання скріншоту в буфер обміну;

- перегляд інформації про програму.

GraphicsEditor має надавати такі можливості:

- включення \ виключення відображення курсора миші на скріншоті;

- малювання пензлем (не прозорорий колір) та маркером (напів прозорий колір);

- нанесення тексту на зображення;

- додавання графічних об’єктів на зображення (лінії, стрілки, прямокутники, квадрати, овали та круги). Прямокутники, квадрати, овали та круги можуть бути як порожнисті так і заповнені;

- зміна кольору для елементів, які наносяться;

- налаштування діаметру пензля та маркера і зміна товщини межі фігур;

- зміна кольору для заповнення фігур;

- налаштування розміру шрифта тексту;

- історія подій малювання («відмінити» та «повторити»);

- згортання вікна;

- збереження відредагованого зображення в різних форматах (png, jpeg, bmp, gif).

Початковими даними для роботи програми є всі дані, які надані операційною системою, зокрема зображення на екрані та поточний системний час.

Вихідними даними програмного забезпечення є зображення, що відредаговані і \ або збережені в задану папку на диску та скопійовані до буферу обміну, та відеозаписи подій на екрані комп’ютера, що збережені в задану папку на диску.

1.3.4 Вимоги до складу та параметрів технічних засобів

Як апаратно-технічні засоби для експлуатації програмного засобу повинні використовуватися IBM-сумісний комп’ютер з характеристиками не нижче:

- процесор Intel Pentium Dual-Core E6800 2.5 ГГц;

- ОЗП 4 Гб;
- відеокарта з підтримкою DirectX 9 з WDDM драйвером;
- жорсткий диск об’ємом 32 Гб;
- маніпулятор типу «миша» і / або клавіатура.

На комп’ютері повинна бути встановлена ОС Windows 10 [6] та .NET Framework версії 4.7.2 та новіше.

1.3.5 Умови експлуатації

Програма може бути записана на пристрій зберігання інформації: магнітний чи твердотільний накопичувач.

Запуск програми має здійснюватися шляхом запуску файлу «VarietyScreenRecorder.exe».

Експлуатація програмного продукту здійснюється відповідно до експлуатаційної документації на розроблений програмний продукт, що відповідає стандартам і містить інформацію, необхідну для його освоєння та експлуатації.

1.3.6 Вимоги до програмної документації

Програмне забезпечення має складатися з керівництва програміста, керівництва оператора, пояснювальної записки та вихідного коду розробленої програми.

1.3.7 Вимоги до маркування й пакування

Програма може поставлятися на диску чи на флеш накопичувачі. На упакуванні має бути зазначена назва програмного продукту – «Variety Screen Recorder», та логотип (рис. 1.6).

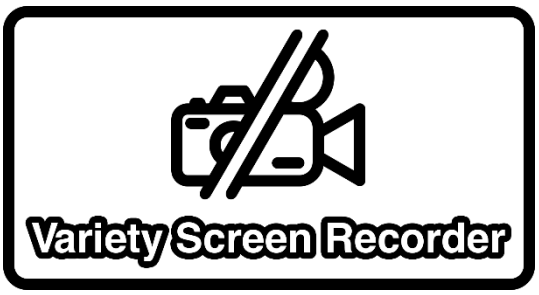

Рисунок 1.6 – Логотип програмного продукту «Variety Screen Recorder»

1.3.8 Вимоги до транспортування та збереження

Вимоги до транспортування та збереження аналогічні тим, що висуваються до накопичувачів на яких зберігається програмний продукт.

2 АНАЛІЗ ПРОГРАМНИХ ЗАСОБІВ

2.1 Операційна система

Вибір операційної системи найважливіший крок на початку розробки програмного забезпечення. Операційна система – це комплекс програм, що керує апаратною складовою комп’ютера, забезпечує керування обчислювальним процесом та організовує взаємодію з користувачем за допомогою текстового чи графічного інтерфейсу.

Однією з найбільших компнаній, що розробляють операційні системи є корпорація Microsoft. Microsoft – це одна з найбільших транснаціональних компаній з виробництва пропрієтарного програмного забезпечення для різного роду обчислювальної техніки – персональних комп’ютерів, ігрових приставок, мобільних телефонів та іншого, розробник найбільш широко поширеної на даний момент в світі програмної платформи – сімейства операційних систем Windows.

Windows – це узагальнююча назва операційних систем для комп’ютерів, розроблених корпорацією Microsoft, що бере свій початок в 1986 році. Згідно з дослідженнями, проведеними компаніює NetMarketShare, близько 90% персональних комп’ютерів працюють під керуванням ОС Windows різних версій. [7].

Найновішою версією операційної системи Microsot є Windows 10, яку було випущено 29 липня 2015 року. Система покликана стати єдиною для різних пристроїв, таких як персональні комп’ютери, планшети, смартфони, консолі Xbox One та інші.

В якості операційної системи, під керівництвом якої має працювати розроблюване програмне забезпечення було обрано одну з версій сімейства комерційних ОС корпорації Microsoft, які орієнтовані на управління за допомогою графічного інтерфейсу, а саме – Windows 10.

2.2 Мова програмування

Мова програмування – це формальна мова, призначена для запису комп'ютерних програм. Мова програмування визначає набір лексичних, синтаксичних та семантичних правил, що визначають зовнішній вигляд програми і дії, які виконує комп'ютер під її управлінням.

На даний момент з популярних мов програмування можна виділити такі: C++, C#, C, Java, Python, JavaScript та PHP.

C# – це універсальна об'єктно-орієнтована мова програмування. ЇЇ мета полягає в забезпеченні продуктивності роботи програмістів. Для цього в мові дотримується баланс між простотою, виразністю і продуктивністю.

C# є дуже близьким родичем мови програмування Java. Microsoft вирішили, користуючись своєю вагою на ринку, створити свій власний аналог Java – мову, в якій корпорація стане повновладним господарем. C# успадкувала від Java концепції віртуальної машини (середовище .NET), байт-коду і більшої безпеки вихідного коду програм.

Середовище .NET – це платформа для створення застосунків, запропонована компанією Microsoft.

Нововведенням C# стала можливість легшої взаємодії, порівняно з мовами-попередниками, з кодом програм, написаних іншими мовами, що є важливим при створенні великих проектів. А саму мову програмування було визначено флагманською мовою корпорації Microsoft, бо вона найповніше використовує нові можливості середовища .NET.

З самої першої версії головним архітектором мови був Андерс Хейлсберг (творець Turbo Pascal та Delphi). Мова C # нейтральна щодо платформ і працює з різними компіляторами, специфічними для платформ, найбільш значимою з яких вважається Microsoft .NET Framework для ОС Windows [8].

В якості мови програмування було обрано мову програмування C# версії 7.3.

2.3 Середовище розробки

Інтегроване середовище розробки (IDE) – це комплексне програмне рішення для розробки програмного забезпечення. Зазвичай воно складається з редактора вихідного коду та інструментів для автоматизації складання та відлагодження програми. Більшість сучасних середовищ розробки має можлівість автодоповнення коду.

Microsoft Visual Studio – це серія продуктів фірми Microsoft, які включають інтегроване середовище розробки програмного забезпечення та низку інших інструментальних засобів. Ці продукти дозволяють розробляти як консольні програми, так і програми з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-застосунки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, що підтримуються. IDE Visual Studio підтримує такі мови програмування: C#, C++, Basic, Python, JavaScript.

Версії Visual Studio є абсолютно безкоштовними для учнів, студентів та розробників програм з відкритим програмним кодом. Остання на даний момент версія IDE Microsoft Visual Studio була випущена в грудні 2018 року та мала назву Visual Studio 2019 [9].

В якості середовища для розробки програмного забезпечення було вибрано IDE Microsoft Visual Studio 2019.

2.4 Застосовувані технології

Використання вже створених технологій сприяє можливості повторного використання коду й спрощення процесу розробки програм. Це найпопулярніша практика створення програмних засобів, адже зникає необхідність реалізації великої кількості аналогічних рішень, що дозволяє поглибитися в процес реалізації індивідуальних особливостей програми.

2.4.1 .NET Framework

.NET Framework – це програмна платформа, випущена компанією Microsoft в 2002 році. Основою платформи є загальномовне середовище виконання Common Language Runtime, яке підходить для різних мов програмування. Функціональні можливості CLR доступні в будь-яких мовах програмування, що використовують цю середу.

Вважається, що платформа .NET Framework стала відповіддю компанії Microsoft на популярну платформу Java компанії Sun Microsystems.

Хоча .NET є патентованою технологією корпорації Microsoft і офіційно розрахована на роботу під операційними системами сімейства Microsoft Windows, існують незалежні проекти, що дозволяють запускати програми .NET на деяких інших операційних системах.

В даний час .NET Framework отримує розвиток у вигляді .NET Core, що передбачає кроссплатформенную розробку й експлуатацію.

Програма для .NET Framework, написана на будь-якій підтримуваній мові програмування, спочатку перекладається компілятором в єдиний для .NET проміжний байт-код Common Intermediate Language (раніше називався Microsoft Intermediate Language). Потім код або виконується віртуальною машиною CLR, або транслюється в виконуваний код для конкретного процесора.

Використання віртуальної машини більш популярне рішення, оскільки позбавляє розробників від необхідності піклуватися про особливості апаратної частини.

Архітектуру .NET Framework описано й опубліковано в специфікації CLI, розробленої Microsoft і затвердженої ISO і ECMA. У CLI описані типи даних .NET, формат метаданих про структуру програми, система виконання байт-коду та інше.

В якості програмної платформи було обрано .NET Framework версії 4.7.2.

2.4.2 Windows Forms

Об’єктні класи .NET, доступні для всіх підтримуваних мов програмування та містяться в бібліотеці Framework Class Library. У FCL входять такі класи: Windows Forms, ADO.NET, ASP.NET, Language Integrated Query, Windows Presentation Foundation, Windows Communication Foundation та інші.

Windows Forms – це інтерфейс програмування додатків, що відповідає за графічний інтерфейс користувача і є частиною Microsoft .NET Framework. Даний інтерфейс спрощує доступ до елементів інтерфейсу Microsoft Windows за рахунок створення обгортки для існуючого Win32 API в керованому коді. Причому керований код – це класи, що реалізують API для Windows Forms, що не залежать від мови розробки.

В якості інтерфейсу програмування додатків було обрано Windows Forms.

2.4.3 NuGet пакети

NuGet – система управління пакетами для платформ розробки Microsoft, в першу чергу бібліотек .NET Framework. Керується за допомогою .NET Foundation.

Використання сторонніх бібліотек допомагає розширювати функціонал програмного забезпечення при цьому зменшуючи час розробки програми та створюючи програмний код, що можна використовувати як вже налагоджений компонен. Такий підхід дозволяє розроблювати складне та надійне програмне забезпечення на основі вже готових та перевірених частин, об’єднуючи їх в одне ціле.

В якості NuGet пакета прийнято рішення використати сторонню бібліотеку ScreenRecorderLib [10] версії 1.9.0.

ScreenRecorderLib – це бібліотека .NET для запису екрана в Windows, використовуючи вбудовану програму Microsoft Media Foundation для кодування в режимі реального часу для зображень h264 відео. Для роботи цієї бібліотеки потрібна система Windows 8 або новішої версії, а також встановлений Visual C ++ Redistributable Db2015. Компонент розповсюджується як вільне програмне забезпечення.

Бібліотека ScreenRecorderLib надає зручний програмний інтерфейс, що дозволяє захоплювати зображення з екрану комп'ютера та кодувати їх в режимі реального часу за допомогою ліцензованого стандарту стиснення відео H.264. Використання цієї бібліотеки дозволяє захоплювати відео в високій якості з екрану, при достатньо низькому навантаженню на процесор та відеокарту комп'ютера, що є однією з вимог до такого типу програм. А отже використання цієї бібліотеки є доцільним при розробці такого програмного забезпечення.

2.4.4 Шаблон проектування Command

Шаблон проектування Command – це поведінковий патерн проектування, який перетворює запити на об'єкти, дозволяючи передавати їх як аргументи під час виклику методів, ставити запити в чергу, логувати їх, а також підтримувати скасування операцій.

Головна річ, яка потрібна для того, щоб мати можливість скасовувати операції – це збереження історії дій користувача. Серед багатьох способів реалізації цієї можливості патерн Command є найпопулярнішим.

Історія команд виглядає як стек, до якого потрапляють усі виконані об'єкти команд. Кожна команда перед виконанням операції зберігає поточний стан об'єкта, з яким вона працюватиме. Після виконання операції копія команди потрапляє до стеку історії, продовжуючи нести у собі збережений стан об'єкта. Якщо знадобиться скасування, програма візьме останню команду з історії та відновить збережений у ній стан.

Цей спосіб має дві особливості:

- точний стан об’єктів не дуже просто зберегти, адже його частина може бути приватною.

- копії стану можуть займати досить багато оперативної пам’яті. Тому іноді можна вдатися до альтернативної реалізації, тобто замість відновлення старого стану, команда виконає зворотню дію.

За допомогою використання поведінкового шаблону проектування Command має бути реалізовано можливість збереження історії дій користувача, а отже і можливість скасування операцій, а саме: «Undo» та «Redo», що буде здійснювати невелике навантаження на систему комп’ютера, на відміну від інших можливих реалізацій такого функціоналу.

3 РОЗРОБКА ТА РЕАЛІЗАЦІЯ КОМПОНЕНТІВ СИСТЕМИ

3.1 Use Case діаграма

Use Case діаграма – це діаграма варіантів використання програми та її реакції на дії користувача. Тобто діаграма, яка відображає відносини між акторами і прецедентами, що дозволяє описати систему на концептуальному рівні.

Прецедент – це частина функціональності модельованої системи, завдяки якій користувач може отримати конкретний та потрібний йому результат. Прецедент відповідає окремому сервісу системи, визначає один з варіантів її використання й описує типовий спосіб взаємодії користувача з системою. Такі діаграми зазвичай застосовуються для специфікації зовнішніх вимог до системи.

На рисунку 3.1 наведено зображення Use Case діаграми розроблюваного програмного забезпечення.

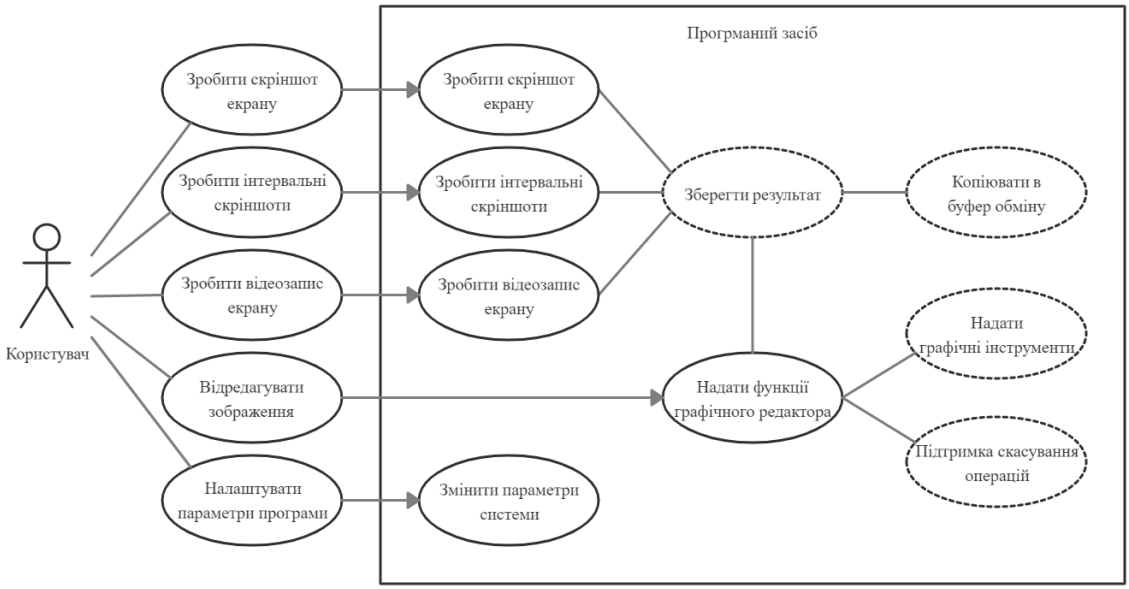

Рисунок 3.1 – Use Case діаграма програми «VarietyScreenRecorder»

В даному випадку користувач програмного засобу – це актор системи, а розроблюваний програмний засіб – прецедент.

Користувач при роботі з програмою ставить собі на меті виконати такі функції: зробити скріншот екрану, зробити інтервальні скріншоти, зробити відеозапис екрану, відредагувати зображення та налаштувати параметри програми. Взаємодіючи з графічним інтерфейсом програми (натискаючи на кнопки та інші елементи), саме він посилає сигнали програмному засобу.

Програмний засіб, отримуючи результати взаємодії користувача з графічним інтерфейсом, виконує необхідні дії над системою (виконує код, відкриває нові вікна програми, змінює елементи інтерфейсу) та викликає різні модулі: робить скріншот екрану, робить інтервальні скріншоти, робить відеозапис екрану, надає функції графічного редактора.

Функції збереження результату, копіювання в буфер обміну, надання графічних інструментів, підтримки скасування операцій є складовими частинами вищезазначених модулів.

3.2 Розробка логотипу та інших зображень

Програмне забезпечення для захоплення зображення й відео з екрану комп’ютера асоціюється з процесом зйомки за допомогою фото- та відеокамери. Також однією з невід’ємних частин написання коду програми є коментарі. Знак подвійного слеша – знак коментаря мови C#.

Для створення логотипу програми було прийнято рішення поєднати ці елементи та використати мінімалістичний проте інтуїтивно зрозумілий стиль оформлення (рис. 3.2).

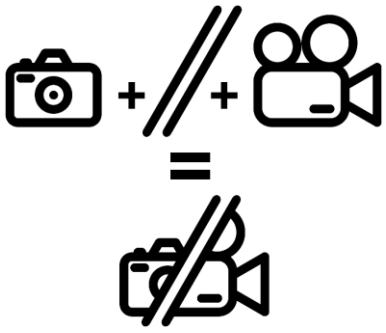

Рисунок 3.2 – Розробка логотипу програми

Головними критеріями для всіх зображень, що мають використовуватися в розроблюваній програмі є мінімалістичний стиль та інтуїтивна зрозумілість.

Для графічного інтерфейсу тієї частини додатку, яка відповідає за процес захоплення зображення та відео було розроблено спеціальні зображення для кнопок (рис. 3.3).

Рисунок 3.3 – Зображення для кнопок однієї частини інтерфейсу

Для графічного інтерфейсу тієї частини додатку, що реалізує основні функції графічного редактора у відкритому доступі було знайдено відповідні зображення популярних графічних інструментів у мінімалістичному й плоскому дизайні (рис. 3.4).

Рисунок 3.4 – Зображення для графічних інструментів

3.3 Проектування та реалізація класів

Протягом проектування класів системи було прийнято рішення розділити програму на два модулі:

- VarietyScreenRecorder – відповідає за процес захоплення зображення й відео з екрану комп’ютера, дозволяє налаштувати програму та переглянути основну інформацію;

- GraphicsEditor – відповідає за редагування отриманого зображення, надаючи інтерфейс графічного редактора з підтримкою можливості скасування операцій.

В результаті реалізації програмного забезпечення було створено такі класи:

- HotKeyManager;
- ScreenshotManager;
- VideoManager;
- WindowVSR_Main;
- WindowVSR_ScreenshotCrop;
- WindowVSR_Settings;
- WindowVSR_Information;
- WindowGE_Main;
- IShape;
- ShapeCursor;
- ShapeEllipse;
- ShapeLine;
- ShapeMarker;
- ShapePen;
- ShapeRectangle;
- ShapeText;
- ShapesHistory;
- WindowGE_GetSize;
- WindowGE_GetText;
- Program.

Клас HotKeyManager відповідає за функції керуванням та розпізнаванням різних сполучень клавіш, що можуть використовуватися в програмі. Він реалізує простий програмний інтерфейс (функції GetHotKey та isHotKey), що спрощує перетворення системної інформації про натиснуті клавіши до формату, який використовується в системі.

Клас ScreenshotManager відповідає за процес захоплення зображення з екрану комп’ютера та функції його збереження в різних форматах (функції MakeScreenshot та SaveScreenshot).

Клас VideoManager є оболонкою для NuGet пакета з бібліотекою ScreenRecorderLib, що використовується для захоплення відео з екрану комп’ютера. Цей клас реалізує простий програмний інтерфейс (функції StartRecording, StopRecording, GetRecordingTime), для спрощення роботи з використовуваним NuGet пакетом.

Клас WindowVSR_Main є основним класом програми, що реалізує функції графічного інтерфейсу користувача для програмного модуля VarietyScreenRecorder. Він надає основний інтерфейс, що дозволяє захоплювати зображення й відео з екрану та налаштовувати основні параметри програми.

Клас WindowVSR_ScreenshotCrop використовується для захоплення зображення з екрану з вибором довільної області. Цей клас зберігає отримане зображення й передає його до основного класу програми.

Клас WindowVSR_Settings реалізує користувацький інтерфейс для роботи з основними налаштуваннями програми. Надає можливість як змінити так і відновити налаштування за замовчанням.

Клас WindowVSR_Information відповідає за відображення інформаційного повідомлення про програмне забезпечення. Відображає на екрані логотип програми та основну інформацію.

Клас WindowGE_Main – це другий основний клас програми, що реалізує функції графічного інтерфейсу користувача для програмного модуля GraphicEditor. Він надає основний користувацький інтерфейс, що дозволяє редагувати отримане зображення за допомогою використання різних графічних інструментів (курсор, лінії, текст, прямокутники, овали та інші).

IShape – інтерфейс що визначає функції, які мають містити графічні інструменти, що використовуються в програмі. Тобто всі графічні інструменти, що використовуються, мають імплементувати функції інтерфейсу IShape.

Клас ShapeCursor є реалізацією інтерфейсу IShape, що додає до графічного редактора можливість відображення курсора на знімку.

Клас ShapeEllipse є реалізацією інтерфейсу IShape, що додає до графічного редактора можливість нанесення графічних об’єктів – овалів та кругів, як порожнистих так і заповнених.

Клас ShapeLine є реалізацією інтерфейсу IShape, що додає до графічного редактора можливість нанесення графічних об’єктів – ліній та стрілок.

Клас ShapeMarker є реалізацією інтерфейсу IShape, що додає до графічного редактора можливість малювання на зображенні напів прозорим кольором (наприклад, для виділення тексту).

Клас ShapePen є реалізацією інтерфейсу IShape, що додає до графічного редактора можливість малювання на зображенні визначеним не прозорим кольором.

Клас ShapeRectangle є реалізацією інтерфейсу IShape, що додає до графічного редактора можливість нанесення графічних об’єктів – прямокутників та квадратів, як порожнистих так і заповнених.

Клас ShapeText є реалізацією інтерфейсу IShape, що додає до графічного редактора можливість нанесення тексту різного розміру на зображення.

Клас ShapesHistory реалізує поведінковий шаблон проектування Command та використовується для збереження історії редагування отриманого зображення. Цей клас надає простий програмний інтерфейс (функції Add, Undo, CanUndo, Redo, CanRedo, Draw), що надає можливість підтримувати скасування операцій малювання та відтворення зображення.

Клас WindowGE_GetSize відповідає за відображення діалогового вікна для отримання числового значення від користувача. Використовується для отримання інформації про діаметр кисті та розмір тексту.

Клас WindowGE_GetText відповідає за відображення діалогового вікна для отримання текстового значення від користувача. Використовується для отримання інформації про текст, який необхідно нанести на зображення.

Статичний клас Program – це головний програмний клас, що відповідає за запуск системи. Це так звана точка входу до програмного засобу та перший клас, що реалізується при створенні програми мовою програмування C# автоматично. Він містить один статичний метод Main, що реалізує функції вибору основного вікна та запуску програми в потоці.

Особливістю інтерфейсу IShape та класів, що його реалізують є те, що вони є вкладеними типами в клас WindowGE_Main.

Всі класи, імена яких починаються зі слова Window, це класами, що відповідають за графічний інтерфейс користувача та є вікнами програми.

Діаграму класів наведено на рисунку 3.5.

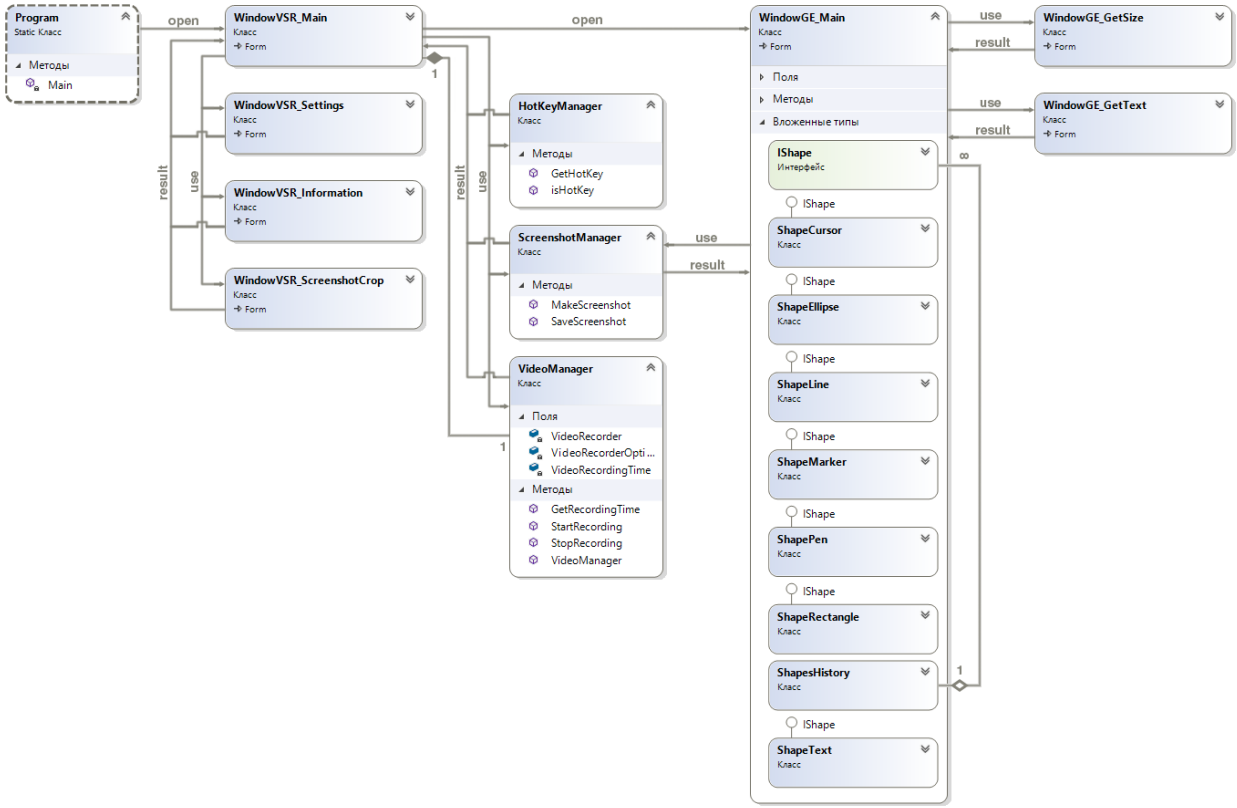

Рисунок 3.5 –Діаграма класів програми

Розроблена діаграма класів відображає всі наявні класи програми та зв’язки між ними:

- Клас Program – це головна точка входу в програму. Саме він має відкривати головну форму користувацького інтерфейсу;

- Клас WindowVSR_Main взаємодіє з класами WindowVSR_Settings, WindowVSR_Information, WindowVSR_ScreenshotCrop, HotKeyManager, ScreenshotManager, VideoManager при цьому отримуючи результат їх роботи. Саме він відповідає за відкриття головного вікна графічного редактора;

- Клас WindowVSR_Main містить один екземпляр класу VideoManager, що відображено на діаграмі як тип зв’язку композиція;

- Класи WindowVSR_Information, WindowVSR_ScreenshotCrop та WindowVSR_Settings є допоміжними класами програми, що відповідають за відображення відповідних вікон графічного інтерфейсу;

- Клас WindowGE_Main взаємодіє з класами WindowGE_GetSize, WindowGE_GetText та ScreenshotManager при цьому отримуючи результат їх роботи;

- За відкриття вікон, що реалізовані класами WindowGE_GetSize та WindowGE_GetText відповідає клас WindowGE_Main;

- Класи ShapeCursor, ShapeEllipse, ShapeLine, ShapeMarker, ShapePen, ShapeRectangle, ShapeText є вбудованими підкласами класу WindowGE_Main;

- Класи ShapeCursor, ShapeEllipse, ShapeLine, ShapeMarker, ShapePen, ShapeRectangle та ShapeText імпелементують інтерфейс IShape;

- Клас ShapesHistory є контейнером для реалізацій інтерфейсу IShape, що відображено на діаграмі як тип зв’язку агрегація.

3.4 Розробка інтерфейсу користувача

Візуально зрозумілий та привабливий інтерфейс сприяє більш приємній та ефективній роботі з програмою. Він має бути реалізований інтуїтивно зрозумілим для користувача – непрофесіонала в комп’ютерній галузі.

При розробці графічного інтерфейсу за мету було визначено створення функціонального, проте простого, неперенавантаженого зайвою інформацією,

інтерфейсу. Інтерфейс, створений за такими принципами сприятиме зручному використанню програми, та підвищить конкурентоздатність системи.

В процесі розробки графічного користувацького інтерфейсу перевага була віддана спокійним, однотонним відтінкам, які привертають увагу користувача. Для полегшення процесу візуального сприйняття, основні елементи управління мають закруглені кути, так як вони не просто приємні для погляду, а й полегшують сприйняття графіки та обробку інформації людиною.

За допомогою інтерфейсу програмування додатків Windows Forms, що надає доступ до стандартних елементів інтерфейсу Microsoft Windows, було розроблено всі необхідні вікна (в термінах розроблюваної програми – форми) для графічного інтерфейсу програми.

В якості структурних елементів інтерфейсів було використано такі стандартні елементи керування:

- Form – вікно програми, що містить всі елементи керування;
- Button – кнопка, що при натисканні може викликати певні обробники;
- CheckBox – поле з прапорцем, що визначає активність деякої умови;
- Label – текст, що може слугувати підказкою;
- MenuStrip – елемент для додавання меню програми;
- NumericUpDown – поле для вводу числових значень;
- Panel – контейнер для розташування елементів інтерфейсу;
- PictureBox – контейнер для зображення;
- TableLayoutPanel – таблиця для розташування елементів;
- TextBox – поле для відображення даних або їх вводу з клавіатури;
- Timer – програмний таймер;
- ToolStrip – меню для вибору інструментів;
- TrackBar – повзунок для швидкого редагування числового значення.

Для головного вікна модуля VarietyScreenRecorder було використано розроблені зображення кнопок, а його інтерфейс побудовано за принципом інтуїтивно зрозумілого та мінімалістичного дизайну (рис. 3.6).

Рисунок 3.6 – Інтерфейс головного вікна програми

Зображення інтерфейсу вікна налаштувань із активною вкладкою основних налаштуваннь програми наведено на рисунку 3.7.

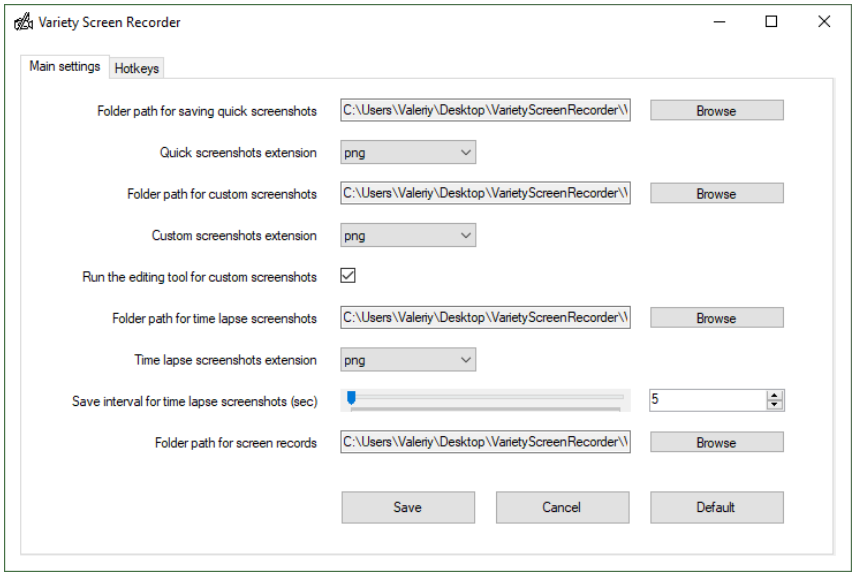

Рисунок 3.7 – Інтерфейс вікна основних налаштувань програми

Зображення інтерфейсу вікна налаштувань з активною вкладкою налаштування сполучень клавіш наведено на рисунку 3.8.

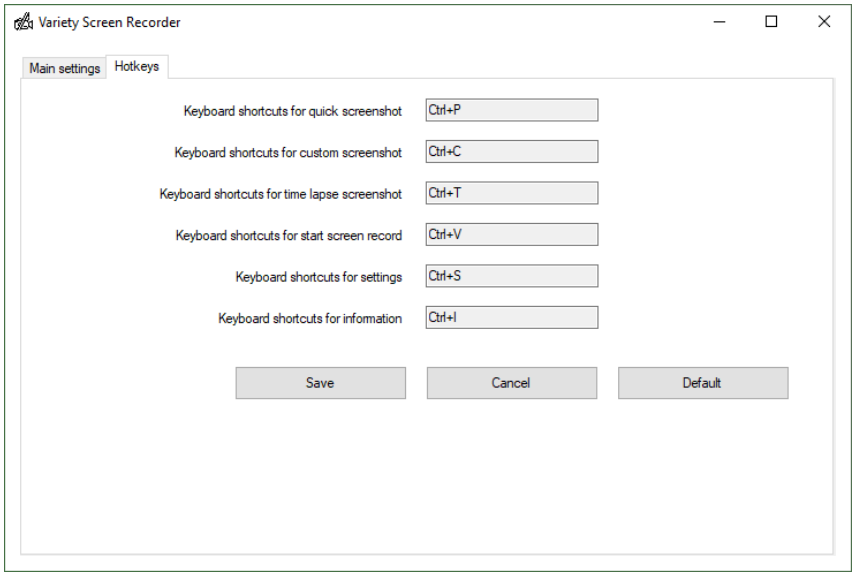

Рисунок 3.8 – Інтерфейс вікна налаштувань сполучень клавіш

На рисунку 3.9 наведено зображення інформаційного вікна програми.

Рисунок 3.9 – Інтерфейс вікна з інформацією про програму

Для головного вікна модуля GraphicsEditor було використано знайдені у вільному доступі зображення графічних інструментів. Його інтерфейс побудовано відповідно до основних вимог в технічному завданні та аналогічно сучасним графічним редакторам (рис. 3.10).

Рисунок 3.10 – Інтерфейс розробленого графічного редактора

Для того, щоб користувач зміг змінити розмір тексту чи товщину лінії було створено вікно для отримання числових значення (рис. 3.11).

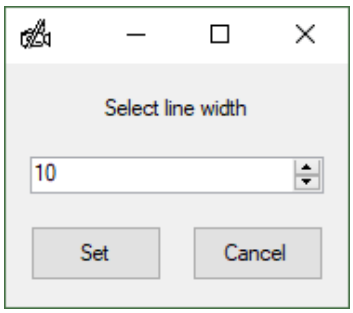

Рисунок 3.11 – Інтерфейс діалогового вікна, що отримує числові значення

Для того, щоб користувач зміг змінити використовуваний текст було створено вікно для отримання текстових значень (рис. 3.12).

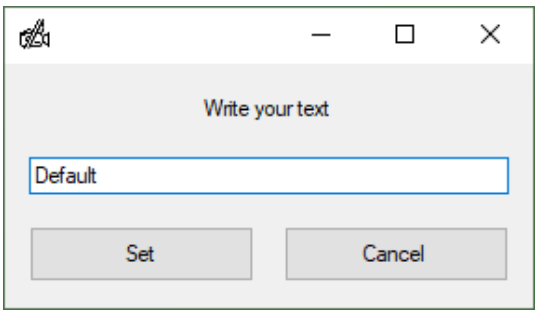

Рисунок 3.12 – Інтерфейс діалогового вікна, що отримує текстові значення

Кожному розробленому вікну графічного інтерфейсу відповідає певна графічна форма, вона ж і є класом обгорткою. Для спрощення розуміння структури програмного забезпечення назву форми було зіставлено з назвою класу, в якому її описано.

Відповідність назви форми до інтерфейсу вікна, який вона утворює:

- WindowVSR_Main – інтерфейс головного вікна програми;
- WindowVSR_Settings – інтерфейс вікна налаштувань програми;
- WindowVSR_Information – інтерфейс вікна з інформацією про програму;
- WindowGE_Main – інтерфейс розробленого графічного редактора;
- WindowGE_GetSize – інтерфейс діалогового вікна, що отримує числові значення;
- WindowGE_GetText – інтерфейс діалогового вікна, що отримує текстові значення.

Повний лістинг дизайну інтерфейсів кожної з форм наведено в додатку А з текстами програми, в пунктах, що мають назву «FormName.Designer.cs», де FormName – це назва форми.

3.5 Зберігання налаштувань програми

Для того, щоб надати користувачу можливість налаштовувати роботу програми «під себе» було прийнято рішення увести систему налаштувань і використовувати її за замовачнням.

В тому випадку, якщо користувач хоче змінити сполучення клавіш, формати зображень, інтервал та шляхи до папок в які зберігаються вихідні дані, йому необхідно просто зайти в налаштування й змінити інформацію у відповідних пунктах. Після збереження, налаштування будуть активними навіть після перезапуску програми.

Реалізовано це за допомогою вбудованої системи налаштувань в Visual Studio (Properties.Settings) та xml файлу що створюється при інсталяції програмного забезпечення.

В якості вбудованої системи налаштувань мається на увазі простір імен Properties, а саме об’єкт Properties.Settings, який автоматично стає доступним відразу при створенні розроблюваного проекту.

Налаштування мають чотири властивості:

- Name – властивість, що вказує на ім’я налаштування, за яким компоненти програми можуть отримати значення під час виконання;

- Type – вказує тип даних, до якого належить параметр. Це може бути числове значення, рядок або один з вбудованих системних типів даних;

- Scope – вказує на спосіб доступу. Може приймати два значення Application або User. User – призначені для користувача налаштування, що доступні для читання або запису під час виконання програми. Application – програмні налаштування, що під час виконання доступні тільки для читання.

- Value – зберігає значення, які повертаються при зверненні до назви параметра. Значення має той тип даних, що визначений у властивості Type.

Організація об’єкту Properties.Settings в файловій системі операційної системи являє собою звичайний xml файл, який можна знайти в папці користувача за шляхом: «C: \ Users \ [user name] \ AppData \ Local \ [(Project Name) or (AssemblyCompany)] \ [name project_cashBuild] \ [AssemblyVersion] \ user.config». Саме за таким шляхом створюється файл налаштувань при інсталяції або першому запуску програми.

Редагування цього файлу дозволяє користувачеві змінювати налаштування програми не запускаючи її.

Структура xml файлу налаштувань відповідає всім принципам побудови файлів на основі розширюваної мови розмітки XML. Структуру та зміст xml файлу налаштувань наведено в додатку А в пункті «user.config».

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови застосування програми

Найменування програмного продукту – «VarietyScreenRecorder». «VarietyScreenRecorder» – це програма призначена для захоплення зображення й відео з екрану комп’ютера та вбудований графічний редактор.

Область застосування – збереження зображеннь або відео з екрану комп’ютера в файлову систему ОС Windows.

Розроблене програмне забезпечення надає можливість користувачу зберігати та редагувати захоплені зображення й відео з екрану комп’ютера.

Як апаратно-технічні засоби для експлуатації програмного засобу повинні використовуватися IBM-сумісний комп’ютер з характеристиками не нижче:

- процесор Intel Pentium Dual-Core E6800 2.5 ГГц;
- ОЗП 4 Гб;
- відеокарта з підтримкою DirectX 9 з WDDM драйвером;
- жорсткий диск об’ємом 32 Гб;
- маніпулятор типу «миша» і / або клавіатура.

Як програмні засоби для належного функціонування програмного забезпечення повинні використовуватися:

- ОС Windows 10 та новіше;
- .NET Framework версії 4.7.2 та новіше;

4.2 Вимоги до програмного забезпечення

Вимоги до функціональних характеристик програми :

- програма повинна мати інтуїтивно зрозумілий інтерфейс;
- програма повинна мати гнучке налаштування параметрів;

- програма має реагувати на будь-які дії користувача з затримкою не більше ніж 0,25 с;

- програма має надавати можливість декількох режимів захоплення екрану: повнорозмірний знімок, знімок з довільним розміром, інтервальний знімок, відеозапис;

- програма має надавати можливість збереження зображень в різних форматах: png, jpeg, bmp, gif;

- програма має надавати можливість редагування знімка з довільним розміром за допомогою вбудованого графічного редактора;

- графічний редактор має надавати можливість редагування зображення за допомогою різних графічних інструментів (курсор, пензель, маркер, лінія, стрілка, прямокутник, овал та інші);

- графічний редактор має підтримувати скасування операцій (історія дій користувача).

4.3 Характеристика програми

Назва програми – «VarietyScreenRecorder».

Обсяг пам’яті, який займає програма – 7 MB.

Програма має міститися в пам’яті комп’ютера та запускатися у разі необхідності захоплення зображень й відео з екрану. Може працювати в фоновому режимі.

Програмне забезпечення складається з таких файлів та папок:

- AssemblyInfo.cs – файл з основною інформацією про налаштування маніфесту збірки;

- Resources.resx – файл, що описує всі ресурси програми;

- Resources.Designer.cs – автоматчно генерований код, що описує ресурси та відповідає за їх ініціалізацію та програмний інтерфейс доступу;

- Settings.Designer.cs – автоматчно генерований код, що описує налаштування програми та програмний інтерфейс доступу до них;

- Program.cs – файл з головним програмним класом, що відповідає за запуск системи;

- App.config – файл, що зберігає користувацькі налаштування програми;

- packages.config – файл з переліком встановлених пакетів;

- Resources – папка з усіма ресурсами програми;

- HotKeyManager.cs – файл, з додатковим класом HotKeyManager;

- ScreenshotManager.cs – файл, з додатковим класом ScreenshotManager;

- VideoManager.cs – файл, з додатковим класом VideoManager;

- WindowVSR_Main.cs – код-логіка головного вікна модуля «VarietyScreenRecorder»;

- WindowVSR_Main.Designer.cs – опис інтерфейсу головного вікна модуля «VarietyScreenRecorder»;

- WindowVSR_Main.resx – перелік ресурсів (параметрів) форми;

- WindowVSR_ScreenshotCrop.cs – код-логіка вікна вибору довільної області зображення;

- WindowVSR_ScreenshotCrop.Designer.cs – опис інтерфейсу вікна вибору довільної області зображення;

- WindowVSR_ScreenshotCrop.resx – перелік ресурсів форми;

- WindowVSR_Settings.cs – код-логіка вікна налаштувань програми;

- WindowVSR_Settings.Designer.cs – файл з описом інтерфейсу вікна налаштувань програми;

- WindowVSR_Settings.resx – перелік ресурсів форми;

- WindowVSR_Information.cs – файл з кодом-логікою вікна з інформацією про програму;

- WindowVSR_Information.Designer.cs – файл з описом інтерфейсу інформаційного вікна;

- WindowVSR_Information.resx – перелік ресурсів форми;

- WindowGE_Main.cs – код-логіка головного вікна модуля «GraphicsEditor»;

- WindowGE_Main.Designer.cs – опис інтерфейсу головного вікна модуля «GraphicsEditor»;

- WindowGE_Main.resx – перелік ресурсів форми;

- WindowGE_GetSize.cs – код-логіка вікна вводу числового значення;

- WindowGE_GetSize.Designer.cs – опис інтерфейсу вікна вводу числового значення;

- WindowGE_GetSize.resx – перелік ресурсів форми;

- WindowGE_GetText.cs – код-логіка вікна вводу текстового значення;

- WindowGE_GetText.Designer.cs – опис інтерфейсу вікна вводу текстового значення;

- WindowGE_ GetText.resx – перелік ресурсів форми.

На рисунку 4.1 наведено зображення структури програми, яку було отримано за допомогою оглядача рішень середовища розробки Visual Studio.

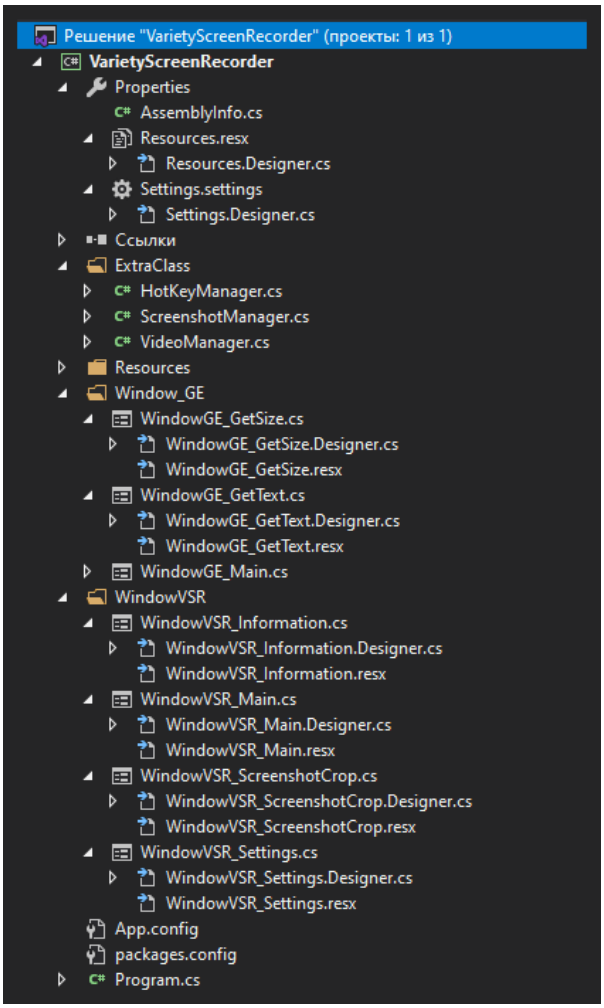

Рисунок 4.1 – Структури програми «VarietyScreenRecorder»

4.4 Звертання до програми

Звертання до програми виконується двома способами:

- за допомогою виконуваного файлу;
- за допомогою командного рядка (без аргументів командного рядка).

Після виклику програми, необхідно слідувати наданим програмою інструкціям.

4.5 Початкові та вихідні дані

Початковими даними для роботи програми є всі дані, які надані операційною системою, зокрема зображення на екрані та поточний системний час.

Вихідними даними програмного забезпечення є зображення, що відредаговані і \ або збережені в задану папку на диску та скопійовані до буферу обміну, та відеозаписи подій на екрані комп’ютера, що збережені в задану папку на диску.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення програми

Найменування програмного продукту – «VarietyScreenRecorder». «VarietyScreenRecorder» – це програма призначена для захоплення зображення й відео з екрану комп’ютера та вбудований графічний редактор.

Область застосування – збереження зображеннь або відео з екрану комп’ютера в файлову систему ОС Windows.

Розроблене програмне забезпечення надає можливість користувачу зберігати та редагувати захоплені зображення й відео з екрану комп’ютера.

5.2 Умови виконання програми

Як апаратно-технічні засоби для експлуатації програмного засобу повинен використовуватися комп’ютер з характеристиками не нижче:

- процесор Intel Pentium Dual-Core E6800 2.5 ГГц;
- ОЗП 4 Гб;
- відеокарта з підтримкою DirectX 9 з WDDM драйвером;
- жорсткий диск об’ємом 32 Гб;
- маніпулятор типу «миша» і / або клавіатура.

Як програмні засоби для програмного забезпечення повинні використовуватися: ОС Windows 10 та .NET Framework версії 4.7.2 чи новіше.

5.3 Виконання програми

Для початку роботи необхідно запустити саму програму. Зробити це можна за допомогою виконуваного файлу (файл в папці з програмою, що має

розширення «.exe») або за допомогою командного рядка (виклик системи без передачі жодних аргументів).

В результаті програму буде запущено, а користувач побаче на екрані головне вікно програмного засобу «VarietyScreenRecorder». На рисунку 5.1 наведено зображення інтерфейсу головного вікна.

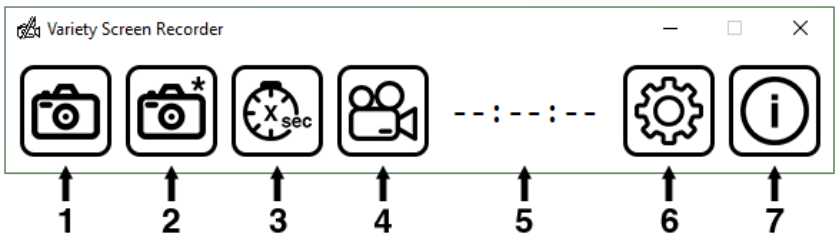

Рисунок 5.1 – Головне вікно програмного засобу «VarietyScreenRecorder»

Інтерфейс головного вікна програми складається з таких елементів:

1 – Кнопка для захоплення зображення з всього екрану комп’ютера;

2 – Кнопка для захоплення зображення з довільної прямокутної області екрану, яку визначає користувач, та подальшого редагування скріншота за допомогою вбудованого графічного редактора;

3 – Кнопка для початку захоплення зображень з всього екрану комп'ютера за заданим в налаштуваннях інтервалом;

4 – Кнопка для початку захоплення відеозапису (відео та звук) з екрану комп’ютера;

5 – Секундомер, що показує тривалість поточного відеозапису з екрану комп’ютера;

6 – Кнопка налаштувань, що викликає вкно налаштувань програми;

7 – Кнопка виклику інформаційного вікна.

Також слід зазначити, що всі зроблені скрішоти, окрім інтервальних, одразу копіюються в буфер обміну ОС. А активні елементи керування підсвічуються світло-зеленим кольором (наприклад, кнопка відеозапису, коли відбувається захоплення відео з екрану).

Викликати певні функції програми можна не тільки клацнув на певну кнопку мишею, а й натиснув визначені сполучення клавіш на клавіатурі.

Всі зроблені скріншоти та відеозаписи автоматично зберігаються в пам’яті комп’ютера з тим форматом та в ті папки, що зазначено в параметрах програми.

Для зміни основних параметрів та гарячих клавіш програми необхідно відкрити вікно налаштувань.

На рисунку 5.2 наведено зображення інтерфейсу вікна налаштувань.

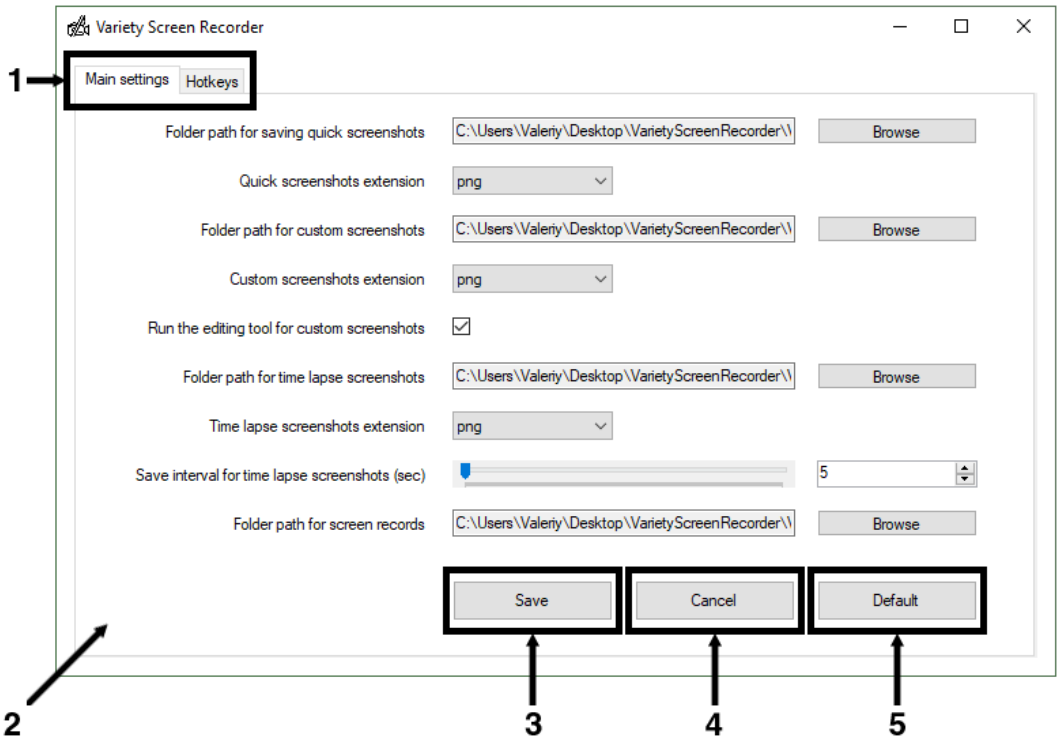

Рисунок 5.2 – Вікно налаштувань програми

Інтерфейс вікна налаштувань складається з таких елементів:

1 – вкладки Main setting (основні налаштування програми) та Hotkeys (налаштування сполучень клавіш). Переключаючи різні вкладки можна переходити до різних типів налаштувань;

2 – всі параметри для налаштування, що доступні в активній вкладці;

3 – кнопка зберегти – для збереження налаштувань в активній вкладці;

4 – кнопка відмінити, використовується, якщо необхідно відмінити останні зміни після вдалого збереження;

5 – кнопка за замовчуванням, призначена для встановлення заводських налаштувань програми.

Також слід зазначити, що при натисканні кнопки зберегти, відмінити чи за замовчуванням, зміни фіксуються тільки в пунктах активної вкладки. Для збереження значень в інших вкладках, необхідно натиснути аналогічні кнопки.

Після збереження, налаштування будуть активними навіть після перезапуску програми.

При захопленні зображення з довільної прямокутної області екрану, яку визначає користувач, в тому випадку, якщо в налаштуваннях встановлена галочка в пункті «Run the editing tool for custom screenshots» буде відкрито вбудований графічний редактор (модуль «GraphicsEditor»).

На рисунку 5.3 наведено зображення інтерфейсу графічного редактора.

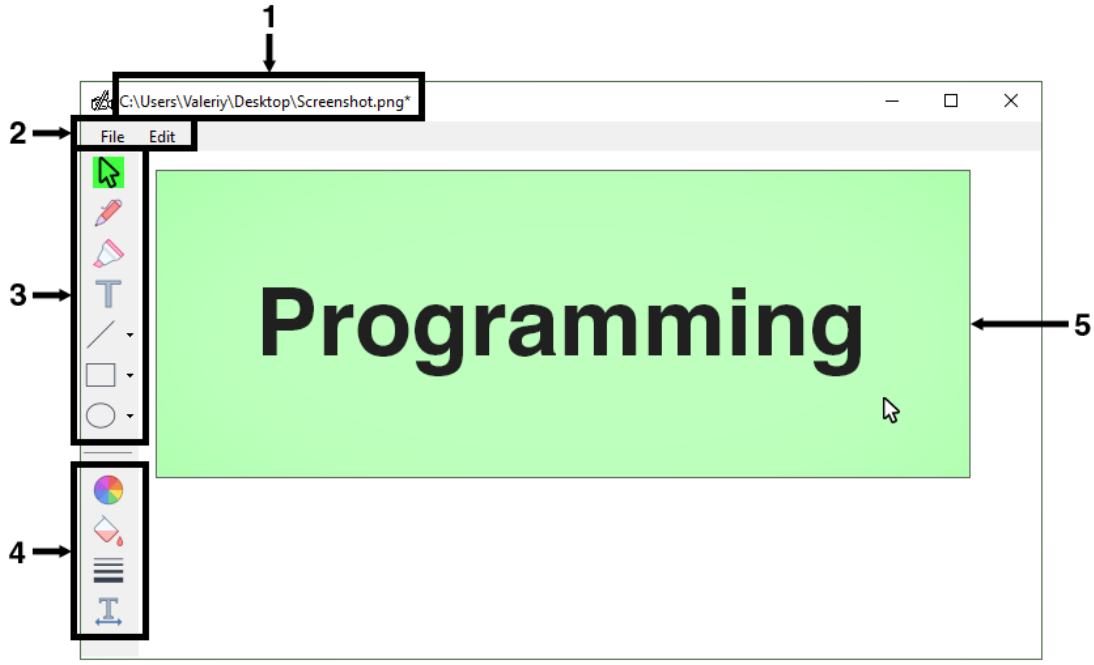

Рисунок 5.3 – Інтерфейс графічного редактора

Інтерфейс графічного редактора складається з таких елементів:

1 – ім’я зображення та шлях до папки, де його збережено;

2 – користувацьке меню програми. За допомогою вкладки «File» можна зберегти скріншот (або сполучення клавіш «Ctrl+S» та «Ctrl+Shift+S»). За допомгою вкладки «Edit» можна керувати історією операцій (або сполучення клавіш «Ctrl+Z» – відмінити та «Ctrl+Y» – повторити);

3 – панель графічних інструментів. Складається з усіх доступних елементів для редагування зображення. Вибраний інтсрумент підсвічується світло-зеленим кольором;

4 – панель налаштувань графічних інструментів. Використовується для редагування розмірів та кольорів фігур, що малюються;

5 – Область малювання. Фактично зроблений скріншот на який наносяться графічні об’єкти.

В тому випадку, коли відредагований скріншот або останні зміни не збережено, то в кінці імені зображення буде світитися зірочка.

Також слід зазначити, що на зображення можна нанести правильні геометричні фігури: вертикальні та горизонтальні лінії, стрілки, а також квадрати та круги. Для цього під час малювання такими інструментами, як: лінія, стрілка, прямокутник та овал, – необхідно затинути клавішу «Shift». При відпусканні клавіши, геометрична фігура прийме вид, який визначений положеннями курсора миші.

5.4 Повідомлення оператору

Для допомоги оператору та виключення ймовірності втрати не збережнного скріншоту при закритті вікон програми, передбачено спеціальне діалогове вікно. Це вікно пропонує зберегти зображення перед закриттям програми (рис. 5.4) і з’являється тільки в тому випадку, коли в історії операцій наявні не зафіксовані зміни скріншоту.

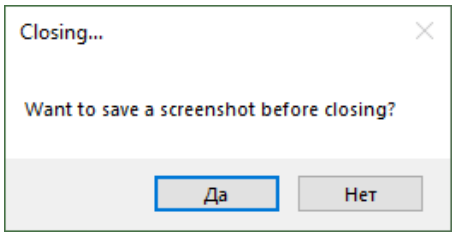

Рисунок 5.4 – Вікно попередження при закритті графічного редактора

ВИСНОВКИ

Під час розробки програми для курсового проекту було проведено аналіз технологій створення програмного забезпечення для операційних систем сімейства Windows.

В якості мови програмування програмного забезпечення було обрано популярну сучасну прогресивну мову програмування C#, адже саме вона відповідала основним вимогам, що висувалися.

В якості середовища розробки було обрано останню версію IDE Microsoft Visual Studio – найпопулярнішу IDE для розробки програм мовою програмування C#.

Протягом розробки програми було досліджено основні особливості використовуваної мови програмування, а саме: роботу з класами та інтерфейсами, обробку подій миші та клавіатури, роботу з елементами графічного інтерфейсу, процесу відображення зображень та графічних елементів системи, роботу з файловою системою ОС та засобами захоплення відео та зображень з екрану комп’ютера.

Створене програмне забезпечення відповідає всім вимогам, що були висунуті, та дозволяє виконувати всі завдання, що описано в технічному завданні та завданні на курсовий проект.

Прогамне забезпечення відповідає принципу вільного програмного забезпечення та надає інтуїтивно зрозумілий користувацький інтерфейс.

Для вдосконалення та розвитку програми, а також підтримки її конкурентоспроможності можна додати такі нові функції:

- додаткові режими захоплення відео з екрану;
- підтримка більшої кількості відео та аудіо девайсів (веб-камери, мікрофони та інше);
- збільшення кількості графічних інструментів;
- можливість автоматичного запуску при завантаженні операційної системи.

ПЕРЕЛІК ПОСИЛАНЬ

1. Використання інструменту «Ножиці» [Електронний ресурс]. – Режим доступу: https://support.microsoft.com/uk-ua/help/13776/windows-10-use-snipping-tool-to-capture-screenshots

2. Screenshot Captor [Electronic resource]. – Access mode: https://www.donationcoder.com/software/mouser/popular-apps/screenshot-captor

3. LightShot – програма для скріншотів [Електронний ресурс]. – Режим доступу: https://app.prntscr.com/ru

4. Bandicam [Електронний ресурс]. – Режим доступу: https://www.bandicam.com/ua

5. Open Broadcaster Software | OBS [Електронний ресурс]. – Режим доступу: https://obsproject.com/ru

6. Вимоги до системи для Windows 10 [Електронний ресурс]. – Режим доступу: https://support.microsoft.com/uk-ua/help/4028142/windows-10-system-requirements

7. Operating system market share [Electronic resource]. – Access mode: https://netmarketshare.com/operating-system-market-share

8. Албахари Дж. C# 7.0. Справочник. Полное описание языка / Дж. Албахари, Б. Албахари. – СпБ.: ООО «Альфа-книга», 2018. – 1024 с.

9. Visual Studio IDE [Електронний ресурс]. – Режим доступу: https://visualstudio.microsoft.com/ru

10. GitHub – ScreenRecorderLib [Electronic resource]. – Access mode: https://github.com/sskodje/ScreenRecorderLib

ДОДАТОК А
ТЕКСТИ ПРОГРАМИ

AssemblyInfo.cs

```
using System.Reflection;
using System.Runtime.CompilerServices;
using System.Runtime.InteropServices;

// Общие сведения об этой сборке предоставляются следующим набором
// набора атрибутов. Измените значения этих атрибутов для изменения сведений,
// связанных со сборкой.
[assembly: AssemblyTitle("VarietyScreenRecorder")]
[assembly: AssemblyDescription("")]
[assembly: AssemblyConfiguration("")]
[assembly: AssemblyCompany("")]
[assembly: AssemblyProduct("VarietyScreenRecorder")]
[assembly: AssemblyCopyright("Copyright ©  2020")]
[assembly: AssemblyTrademark("")]
[assembly: AssemblyCulture("")]

// Установка значения False для параметра ComVisible делает типы в этой сборке
невидимыми
// для компонентов COM. Если необходимо обратиться к типу в этой сборке через
// COM, следует установить атрибут ComVisible в TRUE для этого типа.
[assembly: ComVisible(false)]

// Следующий GUID служит для идентификации библиотеки типов, если этот проект будет
видимым для COM
[assembly: Guid("01356a0a-05fd-478b-ba29-45dd5d2de07d")]

// Сведения о версии сборки состоят из указанных ниже четырех значений:
//
//      Основной номер версии
//      Дополнительный номер версии
//      Номер сборки
//      Редакция
//
// Можно задать все значения или принять номера сборки и редакции по умолчанию
// используя "*", как показано ниже:
// [assembly: AssemblyVersion("1.0.*")]
[assembly: AssemblyVersion("1.0.0.0")]
[assembly: AssemblyFileVersion("1.0.0.0")]
```

Resources.Designer.cs

```
//------------------------------------------------------------------------------
// <auto-generated>
//     Этот код создан программой.
//     Исполняемая версия:4.0.30319.42000
//
//     Изменения в этом файле могут привести к неправильной работе и будут потеряны в
случае
//     повторной генерации кода.
// </auto-generated>
//------------------------------------------------------------------------------

namespace VarietyScreenRecorder.Properties {
    using System;
```

```
    /// <summary>
    ///   Класс ресурса со строгой типизацией для поиска локализованных строк и т.д.
    /// </summary>
    // Этот класс создан автоматически классом StronglyTypedResourceBuilder
    // с помощью такого средства, как ResGen или Visual Studio.
    // Чтобы добавить или удалить член, измените файл .ResX и снова запустите ResGen
    // с параметром /str или перестройте свой проект VS.

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("System.Resources.Tools.Stron
glyTypedResourceBuilder", "16.0.0.0")]
    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]
    internal class Resources {

        private static global::System.Resources.ResourceManager resourceMan;

        private static global::System.Globalization.CultureInfo resourceCulture;


[global::System.Diagnostics.CodeAnalysis.SuppressMessageAttribute("Microsoft.Performa
nce", "CA1811:AvoidUncalledPrivateCode")]
        internal Resources() {
        }

        /// <summary>
        ///   Возвращает кэшированный экземпляр ResourceManager, использованный этим
классом.
        /// </summary>

[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel
.EditorBrowsableState.Advanced)]
        internal static global::System.Resources.ResourceManager ResourceManager {
            get {
                if (object.ReferenceEquals(resourceMan, null)) {
                    global::System.Resources.ResourceManager temp = new
global::System.Resources.ResourceManager("VarietyScreenRecorder.Properties.Resources"
, typeof(Resources).Assembly);
                    resourceMan = temp;
                }
                return resourceMan;
            }
        }

        /// <summary>
        ///   Перезаписывает свойство CurrentUICulture текущего потока для всех
        ///   обращений к ресурсу с помощью этого класса ресурса со строгой
типизацией.
        /// </summary>

[global::System.ComponentModel.EditorBrowsableAttribute(global::System.ComponentModel
.EditorBrowsableState.Advanced)]
        internal static global::System.Globalization.CultureInfo Culture {
            get {
                return resourceCulture;
            }
            set {
                resourceCulture = value;
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
```

```
        /// </summary>
        internal static System.Drawing.Bitmap ActiveButton_Info {
            get {
                object obj = ResourceManager.GetObject("ActiveButton_Info",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap ActiveButton_PhotoCam {
            get {
                object obj = ResourceManager.GetObject("ActiveButton_PhotoCam",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap ActiveButton_PhotoCamSized {
            get {
                object obj = ResourceManager.GetObject("ActiveButton_PhotoCamSized",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap ActiveButton_Settings {
            get {
                object obj = ResourceManager.GetObject("ActiveButton_Settings",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap ActiveButton_TimeLapse {
            get {
                object obj = ResourceManager.GetObject("ActiveButton_TimeLapse",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap ActiveButton_VideoCam {
            get {
                object obj = ResourceManager.GetObject("ActiveButton_VideoCam",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
```

```
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap EmptyImage {
            get {
                object obj = ResourceManager.GetObject("EmptyImage",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Arrow {
            get {
                object obj = ResourceManager.GetObject("GE_Arrow", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Color {
            get {
                object obj = ResourceManager.GetObject("GE_Color", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Cursor {
            get {
                object obj = ResourceManager.GetObject("GE_Cursor", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_CursorImage {
            get {
                object obj = ResourceManager.GetObject("GE_CursorImage",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Ellipse {
            get {
                object obj = ResourceManager.GetObject("GE_Ellipse",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
```

```
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_FillColor {
            get {
                object obj = ResourceManager.GetObject("GE_FillColor",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_FillEllipse {
            get {
                object obj = ResourceManager.GetObject("GE_FillEllipse",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_FillRectangle {
            get {
                object obj = ResourceManager.GetObject("GE_FillRectangle",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Line {
            get {
                object obj = ResourceManager.GetObject("GE_Line", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_LineWidth {
            get {
                object obj = ResourceManager.GetObject("GE_LineWidth",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Marker {
            get {
```

```
                object obj = ResourceManager.GetObject("GE_Marker", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Pen {
            get {
                object obj = ResourceManager.GetObject("GE_Pen", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Rectangle {
            get {
                object obj = ResourceManager.GetObject("GE_Rectangle",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_Text {
            get {
                object obj = ResourceManager.GetObject("GE_Text", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap GE_TextSize {
            get {
                object obj = ResourceManager.GetObject("GE_TextSize",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Icon, аналогичного
(Значок).
        /// </summary>
        internal static System.Drawing.Icon Logo {
            get {
                object obj = ResourceManager.GetObject("Logo", resourceCulture);
                return ((System.Drawing.Icon)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap LogoPNG {
```

```
            get {
                object obj = ResourceManager.GetObject("LogoPNG", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap NotActiveButton_Info {
            get {
                object obj = ResourceManager.GetObject("NotActiveButton_Info",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap NotActiveButton_PhotoCam {
            get {
                object obj = ResourceManager.GetObject("NotActiveButton_PhotoCam",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap NotActiveButton_PhotoCamSized {
            get {
                object obj =
ResourceManager.GetObject("NotActiveButton_PhotoCamSized", resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap NotActiveButton_Settings {
            get {
                object obj = ResourceManager.GetObject("NotActiveButton_Settings",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap NotActiveButton_TimeLapse {
            get {
                object obj = ResourceManager.GetObject("NotActiveButton_TimeLapse",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }

        /// <summary>
```

```
        ///   Поиск локализованного ресурса типа System.Drawing.Bitmap.
        /// </summary>
        internal static System.Drawing.Bitmap NotActiveButton_VideoCam {
            get {
                object obj = ResourceManager.GetObject("NotActiveButton_VideoCam",
resourceCulture);
                return ((System.Drawing.Bitmap)(obj));
            }
        }
    }
}
```

Settings.Designer.cs

```
//------------------------------------------------------------------------------
// <auto-generated>
//     Этот код создан программой.
//     Исполняемая версия:4.0.30319.42000
//
//     Изменения в этом файле могут привести к неправильной работе и будут потеряны в
случае
//     повторной генерации кода.
// </auto-generated>
//------------------------------------------------------------------------------

namespace VarietyScreenRecorder.Properties {


    [global::System.Runtime.CompilerServices.CompilerGeneratedAttribute()]

[global::System.CodeDom.Compiler.GeneratedCodeAttribute("Microsoft.VisualStudio.Edito
rs.SettingsDesigner.SettingsSingleFileGenerator", "16.4.0.0")]
    internal sealed partial class Settings :
global::System.Configuration.ApplicationSettingsBase {

        private static Settings defaultInstance =
((Settings)(global::System.Configuration.ApplicationSettingsBase.Synchronized(new
Settings())));

        public static Settings Default {
            get {
                return defaultInstance;
            }
        }

        [global::System.Configuration.ApplicationScopedSettingAttribute()]
        [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
        [global::System.Configuration.DefaultSettingValueAttribute("1")]
        public int Version {
            get {
                return ((int)(this["Version"]));
            }
        }

        [global::System.Configuration.UserScopedSettingAttribute()]
        [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
        [global::System.Configuration.DefaultSettingValueAttribute("")]
        public string FullSizeScreenshot_Path {
            get {
```

```
        return ((string)(this["FullSizeScreenshot_Path"]));
    }
    set {
        this["FullSizeScreenshot_Path"] = value;
    }
}

[global::System.Configuration.UserScopedSettingAttribute()]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Configuration.DefaultSettingValueAttribute("png")]
public string FullSizeScreenshot_Extension {
    get {
        return ((string)(this["FullSizeScreenshot_Extension"]));
    }
    set {
        this["FullSizeScreenshot_Extension"] = value;
    }
}

[global::System.Configuration.UserScopedSettingAttribute()]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Configuration.DefaultSettingValueAttribute("")]
public string SizedScreenshot_Path {
    get {
        return ((string)(this["SizedScreenshot_Path"]));
    }
    set {
        this["SizedScreenshot_Path"] = value;
    }
}

[global::System.Configuration.UserScopedSettingAttribute()]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Configuration.DefaultSettingValueAttribute("png")]
public string SizedScreenshot_Extension {
    get {
        return ((string)(this["SizedScreenshot_Extension"]));
    }
    set {
        this["SizedScreenshot_Extension"] = value;
    }
}

[global::System.Configuration.UserScopedSettingAttribute()]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Configuration.DefaultSettingValueAttribute("True")]
public bool SizedScreenshot_IsEdit {
    get {
        return ((bool)(this["SizedScreenshot_IsEdit"]));
    }
    set {
        this["SizedScreenshot_IsEdit"] = value;
    }
}

[global::System.Configuration.UserScopedSettingAttribute()]
[global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
[global::System.Configuration.DefaultSettingValueAttribute("")]
public string TimeLapseScreenshot_Path {
    get {
        return ((string)(this["TimeLapseScreenshot_Path"]));
    }
```

```
        set {
            this["TimeLapseScreenshot_Path"] = value;
        }
    }

    [global::System.Configuration.UserScopedSettingAttribute()]
    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.Configuration.DefaultSettingValueAttribute("png")]
    public string TimeLapseScreenshot_Extension {
        get {
            return ((string)(this["TimeLapseScreenshot_Extension"]));
        }
        set {
            this["TimeLapseScreenshot_Extension"] = value;
        }
    }

    [global::System.Configuration.UserScopedSettingAttribute()]
    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.Configuration.DefaultSettingValueAttribute("5")]
    public int TimeLapseScreenshot_Interval {
        get {
            return ((int)(this["TimeLapseScreenshot_Interval"]));
        }
        set {
            this["TimeLapseScreenshot_Interval"] = value;
        }
    }

    [global::System.Configuration.UserScopedSettingAttribute()]
    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.Configuration.DefaultSettingValueAttribute("")]
    public string VideoRecord_Path {
        get {
            return ((string)(this["VideoRecord_Path"]));
        }
        set {
            this["VideoRecord_Path"] = value;
        }
    }

    [global::System.Configuration.UserScopedSettingAttribute()]
    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.Configuration.DefaultSettingValueAttribute("Ctrl+P")]
    public string VSRHotKey_PhotoCam {
        get {
            return ((string)(this["VSRHotKey_PhotoCam"]));
        }
        set {
            this["VSRHotKey_PhotoCam"] = value;
        }
    }

    [global::System.Configuration.UserScopedSettingAttribute()]
    [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
    [global::System.Configuration.DefaultSettingValueAttribute("Ctrl+C")]
    public string VSRHotKey_PhotoCamSized {
        get {
            return ((string)(this["VSRHotKey_PhotoCamSized"]));
        }
        set {
            this["VSRHotKey_PhotoCamSized"] = value;
```

```
            }
        }

        [global::System.Configuration.UserScopedSettingAttribute()]
        [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
        [global::System.Configuration.DefaultSettingValueAttribute("Ctrl+T")]
        public string VSRHotKey_TimeLapse {
            get {
                return ((string)(this["VSRHotKey_TimeLapse"]));
            }
            set {
                this["VSRHotKey_TimeLapse"] = value;
            }
        }

        [global::System.Configuration.UserScopedSettingAttribute()]
        [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
        [global::System.Configuration.DefaultSettingValueAttribute("Ctrl+V")]
        public string VSRHotKey_VideoCam {
            get {
                return ((string)(this["VSRHotKey_VideoCam"]));
            }
            set {
                this["VSRHotKey_VideoCam"] = value;
            }
        }

        [global::System.Configuration.UserScopedSettingAttribute()]
        [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
        [global::System.Configuration.DefaultSettingValueAttribute("Ctrl+S")]
        public string VSRHotKey_Setting {
            get {
                return ((string)(this["VSRHotKey_Setting"]));
            }
            set {
                this["VSRHotKey_Setting"] = value;
            }
        }

        [global::System.Configuration.UserScopedSettingAttribute()]
        [global::System.Diagnostics.DebuggerNonUserCodeAttribute()]
        [global::System.Configuration.DefaultSettingValueAttribute("Ctrl+I")]
        public string VSRHotKey_Info {
            get {
                return ((string)(this["VSRHotKey_Info"]));
            }
            set {
                this["VSRHotKey_Info"] = value;
            }
        }
    }
}
```

Program.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
```

```
using System.Windows.Forms;

namespace VarietyScreenRecorder
{
    static class Program
    {
        /// <summary>
        /// Главная точка входа для приложения.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new WindowVSR_Main());
        }
    }
}
```

App.config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <configSections>
        <sectionGroup name="userSettings"
type="System.Configuration.UserSettingsGroup, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" >
            <section name="VarietyScreenRecorder.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089"
allowExeDefinition="MachineToLocalUser" requirePermission="false" />
        </sectionGroup>
        <sectionGroup name="applicationSettings"
type="System.Configuration.ApplicationSettingsGroup, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" >
            <section name="VarietyScreenRecorder.Properties.Settings"
type="System.Configuration.ClientSettingsSection, System, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
        </sectionGroup>
    </configSections>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.7.2" />
    </startup>
    <userSettings>
        <VarietyScreenRecorder.Properties.Settings>
            <setting name="FullSizeScreenshot_Path" serializeAs="String">
                <value />
            </setting>
            <setting name="FullSizeScreenshot_Extension" serializeAs="String">
                <value>png</value>
            </setting>
            <setting name="SizedScreenshot_Path" serializeAs="String">
                <value />
            </setting>
            <setting name="SizedScreenshot_Extension" serializeAs="String">
                <value>png</value>
            </setting>
            <setting name="SizedScreenshot_IsEdit" serializeAs="String">
                <value>True</value>
```

```
            </setting>
            <setting name="TimeLapseScreenshot_Path" serializeAs="String">
                <value />
            </setting>
            <setting name="TimeLapseScreenshot_Extension" serializeAs="String">
                <value>png</value>
            </setting>
            <setting name="TimeLapseScreenshot_Interval" serializeAs="String">
                <value>5</value>
            </setting>
            <setting name="VideoRecord_Path" serializeAs="String">
                <value />
            </setting>
            <setting name="VSRHotKey_PhotoCam" serializeAs="String">
                <value>Ctrl+P</value>
            </setting>
            <setting name="VSRHotKey_PhotoCamSized" serializeAs="String">
                <value>Ctrl+C</value>
            </setting>
            <setting name="VSRHotKey_TimeLapse" serializeAs="String">
                <value>Ctrl+T</value>
            </setting>
            <setting name="VSRHotKey_VideoCam" serializeAs="String">
                <value>Ctrl+V</value>
            </setting>
            <setting name="VSRHotKey_Setting" serializeAs="String">
                <value>Ctrl+S</value>
            </setting>
            <setting name="VSRHotKey_Info" serializeAs="String">
                <value>Ctrl+I</value>
            </setting>
        </VarietyScreenRecorder.Properties.Settings>
    </userSettings>
    <applicationSettings>
        <VarietyScreenRecorder.Properties.Settings>
            <setting name="Version" serializeAs="String">
                <value>1</value>
            </setting>
        </VarietyScreenRecorder.Properties.Settings>
    </applicationSettings>
</configuration>
```

user.config

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <userSettings>
        <VarietyScreenRecorder.Properties.Settings>
            <setting name="FullSizeScreenshot_Path" serializeAs="String">
                <value />
            </setting>
            <setting name="FullSizeScreenshot_Extension" serializeAs="String">
                <value>png</value>
            </setting>
            <setting name="SizedScreenshot_Path" serializeAs="String">
                <value />
            </setting>
            <setting name="SizedScreenshot_Extension" serializeAs="String">
                <value>png</value>
```

```
            </setting>
            <setting name="SizedScreenshot_IsEdit" serializeAs="String">
                <value>True</value>
            </setting>
            <setting name="TimeLapseScreenshot_Path" serializeAs="String">
                <value />
            </setting>
            <setting name="TimeLapseScreenshot_Extension" serializeAs="String">
                <value>png</value>
            </setting>
            <setting name="TimeLapseScreenshot_Interval" serializeAs="String">
                <value>5</value>
            </setting>
            <setting name="VideoRecord_Path" serializeAs="String">
                <value />
            </setting>
        </VarietyScreenRecorder.Properties.Settings>
    </userSettings>
</configuration>
```

packages.config

```
<?xml version="1.0" encoding="utf-8"?>
<packages>
  <package id="ScreenRecorderLib" version="1.9.0" targetFramework="net472" />
</packages>
```

HotKeyManager.cs

```
using System.Windows.Forms;

namespace VarietyScreenRecorder.ExtraClass
{
    public class HotKeyManager
    {
        public static string GetHotKey(KeyEventArgs e)
        {
            string HotKey = "";

            if (e.Control)
                HotKey += "Ctrl+";

            if (e.Alt)
                HotKey += "Alt+";

            if (e.Shift)
                HotKey += "Shift+";

            HotKey += e.KeyCode;

            return HotKey;
        }

        public static bool isHotKey(KeyEventArgs e)
        {
            if (e.KeyCode >= Keys.A && e.KeyCode <= Keys.Z)
```

```
                return true;
            else
                return false;
        }
    }
}
```

ScreenshotManager.cs

```
using System.Drawing;
using System.IO;
using System.Windows.Forms;

namespace VarietyScreenRecorder.ExtraClass
{
    class ScreenshotManager
    {
        public static Image MakeScreenshot()
        {
            Image Screenshot = new Bitmap(Screen.PrimaryScreen.Bounds.Width,
Screen.PrimaryScreen.Bounds.Height);
            Graphics ScreenshotGraphics = Graphics.FromImage(Screenshot);

            ScreenshotGraphics.CopyFromScreen(0, 0, 0, 0, Screenshot.Size);

            return Screenshot;
        }

        public static void SaveScreenshot(Image Screenshot, string Path, string
Filename, string Extension = "png")
        {
            if (Extension == "")
                Extension = "png";

            if (!Directory.Exists(Path))
                Directory.CreateDirectory(Path);

            switch (Extension)
            {
                case "png":
                    {
                        Screenshot.Save(Path + "\\" + Filename + "." + Extension,
System.Drawing.Imaging.ImageFormat.Png);
                        break;
                    }
                case "jpeg":
                    {
                        Screenshot.Save(Path + "\\" + Filename + "." + Extension,
System.Drawing.Imaging.ImageFormat.Jpeg);
                        break;
                    }
                case "bmp":
                    {
                        Screenshot.Save(Path + "\\" + Filename + "." + Extension,
System.Drawing.Imaging.ImageFormat.Bmp);
                        break;
                    }
                case "gif":
                    {
```

```
                        Screenshot.Save(Path + "\\" + Filename + "." + Extension,
System.Drawing.Imaging.ImageFormat.Gif);
                        break;
                    }
            }
        }
    }
}
```

VideoManager.cs

```
using ScreenRecorderLib;
using System;
using System.Diagnostics;

namespace VarietyScreenRecorder.ExtraClass
{
    public class VideoManager
    {
        private Stopwatch VideoRecordingTime;

        private static RecorderOptions VideoRecorderOptions = new RecorderOptions
        {
            AudioOptions = new AudioOptions
            {
                Bitrate = AudioBitrate.bitrate_128kbps,
                Channels = AudioChannels.Stereo,
                IsAudioEnabled = true
            }
        };

        private Recorder VideoRecorder;

        public VideoManager()
        {
            VideoRecordingTime = new Stopwatch();
            VideoRecorder = Recorder.CreateRecorder(VideoRecorderOptions);
        }

        public void StartRecording(string Path, string FileName)
        {
            VideoRecordingTime.Reset();

            VideoRecorder.Record(Path + "\\" + FileName + ".mp4");

            VideoRecordingTime.Start();
        }

        public void StopRecording()
        {
            VideoRecorder.Stop();

            VideoRecordingTime.Stop();
        }

        public string GetRecordingTime()
        {
            TimeSpan RecordingTime = VideoRecordingTime.Elapsed;
```

```
            string Time = String.Format("{0:00}:{1:00}:{2:00}", RecordingTime.Hours,
RecordingTime.Minutes, RecordingTime.Seconds);

            return Time;
        }
    }
}
```

WindowVSR_Main.cs

```
using System;
using System.Drawing;
using System.Windows.Forms;
using VarietyScreenRecorder.WindowVSR;
using VarietyScreenRecorder.ExtraClass;
using VarietyScreenRecorder.Window_GE;

namespace VarietyScreenRecorder
{
    public partial class WindowVSR_Main : Form
    {
        private VideoManager VideoRecorder = new VideoManager();

        public WindowVSR_Main()
        {
            this.Icon = Properties.Resources.Logo;

            InitializeComponent();

            b_PhotoCam.FlatAppearance.BorderColor = Color.FromArgb(0, 255, 255, 255);
            b_PhotoCamSized.FlatAppearance.BorderColor = Color.FromArgb(0, 255, 255,
255);
            b_TimeLapse.FlatAppearance.BorderColor = Color.FromArgb(0, 255, 255,
255);
            b_VideoCam.FlatAppearance.BorderColor = Color.FromArgb(0, 255, 255, 255);
            b_Settings.FlatAppearance.BorderColor = Color.FromArgb(0, 255, 255, 255);
            b_Info.FlatAppearance.BorderColor = Color.FromArgb(0, 255, 255, 255);
        }
        private Image MakeScreenshot()
        {
            this.Opacity = 0;
            Image Screenshot = ScreenshotManager.MakeScreenshot();
            this.Opacity = 1;

            return Screenshot;
        }

        private void MakeFullSizeScreenshot()
        {
            Image Screenshot = MakeScreenshot();
            Clipboard.SetImage(Screenshot);

            string PathToFile = Properties.Settings.Default.FullSizeScreenshot_Path
== "" ? Environment.CurrentDirectory + "\\Screenshots" :
Properties.Settings.Default.FullSizeScreenshot_Path;
            string Filename = DateTime.Now.ToString("yyyyMMdd_HHmmssfff");

            ScreenshotManager.SaveScreenshot(Screenshot, PathToFile, Filename,
Properties.Settings.Default.FullSizeScreenshot_Extension);
```

```
            Screenshot.Dispose();
        }

        private void MakeSizedScreenshot()
        {
            Image Screenshot = MakeScreenshot();

            WindowVSR_ScreenshotCrop CropWindow = new
WindowVSR_ScreenshotCrop(Screenshot);
            CropWindow.ShowDialog();

            Screenshot.Dispose();
            Screenshot = CropWindow.ResultImage;

            if (Screenshot != null)
            {
                Clipboard.SetImage(Screenshot);

                string PathToFile = Properties.Settings.Default.SizedScreenshot_Path
== "" ? Environment.CurrentDirectory + "\\SizedScreenshots" :
Properties.Settings.Default.SizedScreenshot_Path;
                string Filename = DateTime.Now.ToString("yyyyMMdd_HHmmssfff");

                ScreenshotManager.SaveScreenshot(Screenshot, PathToFile, Filename,
Properties.Settings.Default.SizedScreenshot_Extension);

                if(Properties.Settings.Default.SizedScreenshot_IsEdit)
                {
                    WindowGE_Main GraphicsEditor = new WindowGE_Main(Screenshot);
                    GraphicsEditor.Show();
                }
                else
                {
                    Screenshot.Dispose();
                }
            }

            CropWindow.Dispose();
        }

        private void MakeTimeLapseScreenshot()
        {
            Image Screenshot = MakeScreenshot();

            string PathToFile = Properties.Settings.Default.TimeLapseScreenshot_Path
== "" ? Environment.CurrentDirectory + "\\TimeLapses" :
Properties.Settings.Default.TimeLapseScreenshot_Path;
            string Filename = DateTime.Now.ToString("yyyyMMdd_HHmmssfff");

            ScreenshotManager.SaveScreenshot(Screenshot, PathToFile, Filename,
Properties.Settings.Default.TimeLapseScreenshot_Extension);

            Screenshot.Dispose();
        }

        private void b_PhotoCam_Click(object sender, EventArgs e)
        {
            b_PhotoCam.BackgroundImage = Properties.Resources.ActiveButton_PhotoCam;
            b_PhotoCam.Refresh();

            MakeFullSizeScreenshot();
```

```
            b_PhotoCam.BackgroundImage =
Properties.Resources.NotActiveButton_PhotoCam;
            b_PhotoCam.Refresh();
        }

        private void b_PhotoCamSized_Click(object sender, EventArgs e)
        {
            b_PhotoCamSized.BackgroundImage =
Properties.Resources.ActiveButton_PhotoCamSized;
            b_PhotoCamSized.Refresh();

            MakeSizedScreenshot();

            b_PhotoCamSized.BackgroundImage =
Properties.Resources.NotActiveButton_PhotoCamSized;
            b_PhotoCamSized.Refresh();
        }

        private void b_TimeLapse_Click(object sender, EventArgs e)
        {
            if(t_TimeLapseTimer.Enabled)
            {
                b_TimeLapse.BackgroundImage =
Properties.Resources.NotActiveButton_TimeLapse;
                b_TimeLapse.Refresh();

                t_TimeLapseTimer.Stop();
            }
            else
            {
                b_TimeLapse.BackgroundImage =
Properties.Resources.ActiveButton_TimeLapse;
                b_TimeLapse.Refresh();

                t_TimeLapseTimer.Interval =
Properties.Settings.Default.TimeLapseScreenshot_Interval * 1000;
                t_TimeLapseTimer.Start();
            }
        }

        private void t_TimeLapseTimer_Tick(object sender, EventArgs e)
        {
            MakeTimeLapseScreenshot();
        }

        private void b_VideoCam_Click(object sender, EventArgs e)
        {
            if (t_VideoCamTimer.Enabled)
            {
                l_VideoCamTime.Text = "Saving..";
                l_VideoCamTime.Refresh();

                VideoRecorder.StopRecording();
                t_VideoCamTimer.Enabled = false;

                b_VideoCam.BackgroundImage =
Properties.Resources.NotActiveButton_VideoCam;
                b_VideoCam.Refresh();

                l_VideoCamTime.Text = "--:--:--";
                l_VideoCamTime.Refresh();
            }
```

```
            else
            {
                b_VideoCam.BackgroundImage =
Properties.Resources.ActiveButton_VideoCam;
                b_VideoCam.Refresh();

                l_VideoCamTime.Text = "00:00:00";
                l_VideoCamTime.Refresh();

                string PathToFile = Properties.Settings.Default.VideoRecord_Path ==
"" ? Environment.CurrentDirectory + "\\VideoRecords" :
Properties.Settings.Default.VideoRecord_Path;
                string Filename = DateTime.Now.ToString("yyyyMMdd_HHmmssfff");

                VideoRecorder.StartRecording(PathToFile, Filename);
                t_VideoCamTimer.Enabled = true;
            }
        }

        private void t_VideoCamTimer_Tick(object sender, EventArgs e)
        {
            l_VideoCamTime.Text = VideoRecorder.GetRecordingTime();
            l_VideoCamTime.Refresh();
        }

        private void b_Info_Click(object sender, EventArgs e)
        {
            b_Info.BackgroundImage = Properties.Resources.ActiveButton_Info;
            b_Info.Refresh();

            WindowVSR_Information InformationWindow = new WindowVSR_Information();
            InformationWindow.ShowDialog();

            InformationWindow.Dispose();

            b_Info.BackgroundImage = Properties.Resources.NotActiveButton_Info;
            b_Info.Refresh();
        }

        private void WindowVSR_Main_KeyDown(object sender, KeyEventArgs e)
        {
            if(HotKeyManager.isHotKey(e))
            {
                string HotKey = HotKeyManager.GetHotKey(e);

                if(HotKey == Properties.Settings.Default.VSRHotKey_PhotoCam)
                {
                    b_PhotoCam.PerformClick();
                }
                else if(HotKey ==
Properties.Settings.Default.VSRHotKey_PhotoCamSized)
                {
                    b_PhotoCamSized.PerformClick();
                }
                else if (HotKey == Properties.Settings.Default.VSRHotKey_TimeLapse)
                {
                    b_TimeLapse.PerformClick();
                }
                else if (HotKey == Properties.Settings.Default.VSRHotKey_VideoCam)
                {
                    b_VideoCam.PerformClick();
                }
```

```
                else if (HotKey == Properties.Settings.Default.VSRHotKey_Setting)
                {
                    b_Settings.PerformClick();
                }
                else if (HotKey == Properties.Settings.Default.VSRHotKey_Info)
                {
                    b_Info.PerformClick();
                }
            }
        }

        private void b_Settings_Click(object sender, EventArgs e)
        {
            b_Settings.BackgroundImage = Properties.Resources.ActiveButton_Settings;
            b_Settings.Refresh();

            WindowVSR_Settings SettingsWindow = new WindowVSR_Settings();
            SettingsWindow.ShowDialog();

            SettingsWindow.Dispose();

            b_Settings.BackgroundImage =
Properties.Resources.NotActiveButton_Settings;
            b_Settings.Refresh();
        }
    }
}
```

WindowVSR_Main.Designer.cs

```
namespace VarietyScreenRecorder
{
    partial class WindowVSR_Main
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
```

```
        private void InitializeComponent()
        {
            this.components = new System.ComponentModel.Container();
            this.b_PhotoCam = new System.Windows.Forms.Button();
            this.b_PhotoCamSized = new System.Windows.Forms.Button();
            this.b_TimeLapse = new System.Windows.Forms.Button();
            this.b_VideoCam = new System.Windows.Forms.Button();
            this.b_Settings = new System.Windows.Forms.Button();
            this.b_Info = new System.Windows.Forms.Button();
            this.l_VideoCamTime = new System.Windows.Forms.Label();
            this.t_TimeLapseTimer = new System.Windows.Forms.Timer(this.components);
            this.t_VideoCamTimer = new System.Windows.Forms.Timer(this.components);
            this.SuspendLayout();
            //
            // b_PhotoCam
            //
            this.b_PhotoCam.BackColor = System.Drawing.Color.Transparent;
            this.b_PhotoCam.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.NotActiveButton_PhotoCam;
            this.b_PhotoCam.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.Center;
            this.b_PhotoCam.Cursor = System.Windows.Forms.Cursors.Default;
            this.b_PhotoCam.FlatAppearance.BorderColor = System.Drawing.Color.Black;
            this.b_PhotoCam.FlatAppearance.BorderSize = 0;
            this.b_PhotoCam.FlatAppearance.MouseDownBackColor =
System.Drawing.Color.Transparent;
            this.b_PhotoCam.FlatAppearance.MouseOverBackColor =
System.Drawing.Color.Transparent;
            this.b_PhotoCam.FlatStyle = System.Windows.Forms.FlatStyle.Flat;
            this.b_PhotoCam.ForeColor = System.Drawing.Color.Transparent;
            this.b_PhotoCam.Location = new System.Drawing.Point(10, 10);
            this.b_PhotoCam.Margin = new System.Windows.Forms.Padding(0);
            this.b_PhotoCam.Name = "b_PhotoCam";
            this.b_PhotoCam.Size = new System.Drawing.Size(64, 64);
            this.b_PhotoCam.TabIndex = 0;
            this.b_PhotoCam.UseVisualStyleBackColor = false;
            this.b_PhotoCam.Click += new System.EventHandler(this.b_PhotoCam_Click);
            //
            // b_PhotoCamSized
            //
            this.b_PhotoCamSized.BackColor = System.Drawing.Color.Transparent;
            this.b_PhotoCamSized.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.NotActiveButton_PhotoCamSized;
            this.b_PhotoCamSized.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.Center;
            this.b_PhotoCamSized.Cursor = System.Windows.Forms.Cursors.Default;
            this.b_PhotoCamSized.FlatAppearance.BorderColor =
System.Drawing.Color.Black;
            this.b_PhotoCamSized.FlatAppearance.BorderSize = 0;
            this.b_PhotoCamSized.FlatAppearance.MouseDownBackColor =
System.Drawing.Color.Transparent;
            this.b_PhotoCamSized.FlatAppearance.MouseOverBackColor =
System.Drawing.Color.Transparent;
            this.b_PhotoCamSized.FlatStyle = System.Windows.Forms.FlatStyle.Flat;
            this.b_PhotoCamSized.ForeColor = System.Drawing.Color.Transparent;
            this.b_PhotoCamSized.Location = new System.Drawing.Point(84, 10);
            this.b_PhotoCamSized.Margin = new System.Windows.Forms.Padding(0);
            this.b_PhotoCamSized.Name = "b_PhotoCamSized";
            this.b_PhotoCamSized.Size = new System.Drawing.Size(64, 64);
            this.b_PhotoCamSized.TabIndex = 1;
            this.b_PhotoCamSized.UseVisualStyleBackColor = false;
```

```
            this.b_PhotoCamSized.Click += new
System.EventHandler(this.b_PhotoCamSized_Click);
            //
            // b_TimeLapse
            //
            this.b_TimeLapse.BackColor = System.Drawing.Color.Transparent;
            this.b_TimeLapse.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.NotActiveButton_TimeLapse;
            this.b_TimeLapse.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.Center;
            this.b_TimeLapse.Cursor = System.Windows.Forms.Cursors.Default;
            this.b_TimeLapse.FlatAppearance.BorderColor = System.Drawing.Color.Black;
            this.b_TimeLapse.FlatAppearance.BorderSize = 0;
            this.b_TimeLapse.FlatAppearance.MouseDownBackColor =
System.Drawing.Color.Transparent;
            this.b_TimeLapse.FlatAppearance.MouseOverBackColor =
System.Drawing.Color.Transparent;
            this.b_TimeLapse.FlatStyle = System.Windows.Forms.FlatStyle.Flat;
            this.b_TimeLapse.ForeColor = System.Drawing.Color.Transparent;
            this.b_TimeLapse.Location = new System.Drawing.Point(158, 10);
            this.b_TimeLapse.Margin = new System.Windows.Forms.Padding(0);
            this.b_TimeLapse.Name = "b_TimeLapse";
            this.b_TimeLapse.Size = new System.Drawing.Size(64, 64);
            this.b_TimeLapse.TabIndex = 2;
            this.b_TimeLapse.UseVisualStyleBackColor = false;
            this.b_TimeLapse.Click += new
System.EventHandler(this.b_TimeLapse_Click);
            //
            // b_VideoCam
            //
            this.b_VideoCam.BackColor = System.Drawing.Color.Transparent;
            this.b_VideoCam.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.NotActiveButton_VideoCam;
            this.b_VideoCam.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.Center;
            this.b_VideoCam.Cursor = System.Windows.Forms.Cursors.Default;
            this.b_VideoCam.FlatAppearance.BorderColor = System.Drawing.Color.Black;
            this.b_VideoCam.FlatAppearance.BorderSize = 0;
            this.b_VideoCam.FlatAppearance.MouseDownBackColor =
System.Drawing.Color.Transparent;
            this.b_VideoCam.FlatAppearance.MouseOverBackColor =
System.Drawing.Color.Transparent;
            this.b_VideoCam.FlatStyle = System.Windows.Forms.FlatStyle.Flat;
            this.b_VideoCam.ForeColor = System.Drawing.Color.Transparent;
            this.b_VideoCam.Location = new System.Drawing.Point(232, 10);
            this.b_VideoCam.Margin = new System.Windows.Forms.Padding(0);
            this.b_VideoCam.Name = "b_VideoCam";
            this.b_VideoCam.Size = new System.Drawing.Size(64, 64);
            this.b_VideoCam.TabIndex = 3;
            this.b_VideoCam.UseVisualStyleBackColor = false;
            this.b_VideoCam.Click += new System.EventHandler(this.b_VideoCam_Click);
            //
            // b_Settings
            //
            this.b_Settings.BackColor = System.Drawing.Color.Transparent;
            this.b_Settings.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.NotActiveButton_Settings;
            this.b_Settings.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.Center;
            this.b_Settings.Cursor = System.Windows.Forms.Cursors.Default;
            this.b_Settings.FlatAppearance.BorderColor = System.Drawing.Color.Black;
            this.b_Settings.FlatAppearance.BorderSize = 0;
```

```
            this.b_Settings.FlatAppearance.MouseDownBackColor =
System.Drawing.Color.Transparent;
            this.b_Settings.FlatAppearance.MouseOverBackColor =
System.Drawing.Color.Transparent;
            this.b_Settings.FlatStyle = System.Windows.Forms.FlatStyle.Flat;
            this.b_Settings.ForeColor = System.Drawing.Color.Transparent;
            this.b_Settings.Location = new System.Drawing.Point(432, 10);
            this.b_Settings.Margin = new System.Windows.Forms.Padding(0);
            this.b_Settings.Name = "b_Settings";
            this.b_Settings.Size = new System.Drawing.Size(64, 64);
            this.b_Settings.TabIndex = 4;
            this.b_Settings.UseVisualStyleBackColor = false;
            this.b_Settings.Click += new System.EventHandler(this.b_Settings_Click);
            //
            // b_Info
            //
            this.b_Info.BackColor = System.Drawing.Color.Transparent;
            this.b_Info.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.NotActiveButton_Info;
            this.b_Info.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.Center;
            this.b_Info.Cursor = System.Windows.Forms.Cursors.Default;
            this.b_Info.FlatAppearance.BorderColor = System.Drawing.Color.Black;
            this.b_Info.FlatAppearance.BorderSize = 0;
            this.b_Info.FlatAppearance.MouseDownBackColor =
System.Drawing.Color.Transparent;
            this.b_Info.FlatAppearance.MouseOverBackColor =
System.Drawing.Color.Transparent;
            this.b_Info.FlatStyle = System.Windows.Forms.FlatStyle.Flat;
            this.b_Info.ForeColor = System.Drawing.Color.Transparent;
            this.b_Info.Location = new System.Drawing.Point(506, 10);
            this.b_Info.Margin = new System.Windows.Forms.Padding(0);
            this.b_Info.Name = "b_Info";
            this.b_Info.Size = new System.Drawing.Size(64, 64);
            this.b_Info.TabIndex = 5;
            this.b_Info.UseVisualStyleBackColor = false;
            this.b_Info.Click += new System.EventHandler(this.b_Info_Click);
            //
            // l_VideoCamTime
            //
            this.l_VideoCamTime.AutoSize = true;
            this.l_VideoCamTime.BackColor = System.Drawing.Color.Transparent;
            this.l_VideoCamTime.Font = new System.Drawing.Font("Consolas", 18F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.l_VideoCamTime.Location = new System.Drawing.Point(306, 28);
            this.l_VideoCamTime.Margin = new System.Windows.Forms.Padding(0);
            this.l_VideoCamTime.Name = "l_VideoCamTime";
            this.l_VideoCamTime.Size = new System.Drawing.Size(116, 28);
            this.l_VideoCamTime.TabIndex = 6;
            this.l_VideoCamTime.Text = "--:--:--";
            this.l_VideoCamTime.TextAlign =
System.Drawing.ContentAlignment.MiddleLeft;
            //
            // t_TimeLapseTimer
            //
            this.t_TimeLapseTimer.Tick += new
System.EventHandler(this.t_TimeLapseTimer_Tick);
            //
            // t_VideoCamTimer
            //
            this.t_VideoCamTimer.Tick += new
System.EventHandler(this.t_VideoCamTimer_Tick);
```

```
            //
            // WindowVSR_Main
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.BackColor = System.Drawing.Color.White;
            this.ClientSize = new System.Drawing.Size(579, 85);
            this.Controls.Add(this.l_VideoCamTime);
            this.Controls.Add(this.b_Info);
            this.Controls.Add(this.b_Settings);
            this.Controls.Add(this.b_VideoCam);
            this.Controls.Add(this.b_TimeLapse);
            this.Controls.Add(this.b_PhotoCamSized);
            this.Controls.Add(this.b_PhotoCam);
            this.ForeColor = System.Drawing.Color.Black;
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;
            this.KeyPreview = true;
            this.MaximizeBox = false;
            this.Name = "WindowVSR_Main";
            this.RightToLeft = System.Windows.Forms.RightToLeft.No;
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Variety Screen Recorder";
            this.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.WindowVSR_Main_KeyDown);
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.Button b_PhotoCam;
        private System.Windows.Forms.Button b_PhotoCamSized;
        private System.Windows.Forms.Button b_TimeLapse;
        private System.Windows.Forms.Button b_VideoCam;
        private System.Windows.Forms.Button b_Settings;
        private System.Windows.Forms.Button b_Info;
        private System.Windows.Forms.Label l_VideoCamTime;
        private System.Windows.Forms.Timer t_TimeLapseTimer;
        private System.Windows.Forms.Timer t_VideoCamTimer;
    }
}
```

WindowVSR_ScreenshotCrop.cs

```
using System;
using System.Drawing;
using System.Windows.Forms;

namespace VarietyScreenRecorder.WindowVSR
{
    public partial class WindowVSR_ScreenshotCrop : Form
    {
        private static readonly Size MIN_SIZE = new Size(50, 50);

        public Image ResultImage;

        bool isMousePressed = false;
```

```
        Rectangle ScreenshotSelectedRectangle;
        Point MousePressedPoint;

        public WindowVSR_ScreenshotCrop(Image image)
        {
            this.Icon = Properties.Resources.Logo;
            this.BackgroundImage = image;
            this.SetStyle(ControlStyles.OptimizedDoubleBuffer |
ControlStyles.AllPaintingInWmPaint, true);

            InitializeComponent();
        }

        protected override void OnPaint(PaintEventArgs e)
        {
            base.OnPaint(e);

            DrawSelection(e);
        }

        private void WindowVSR_ScreenshotCrop_MouseDown(object sender, MouseEventArgs
e)
        {
            if(e.Button == MouseButtons.Left)
            {
                isMousePressed = true;

                MousePressedPoint = e.Location;
                ScreenshotSelectedRectangle = new Rectangle(MousePressedPoint, new
Size(0, 0));
            }
        }

        private void WindowVSR_ScreenshotCrop_MouseMove(object sender, MouseEventArgs
e)
        {
            if(isMousePressed)
            {
                int X1 = Math.Min(MousePressedPoint.X, e.X);
                int Y1 = Math.Min(MousePressedPoint.Y, e.Y);
                int X2 = Math.Max(MousePressedPoint.X, e.X);
                int Y2 = Math.Max(MousePressedPoint.Y, e.Y);

                ScreenshotSelectedRectangle = new Rectangle(X1, Y1, X2 - X1 + 1, Y2 -
Y1 + 1);
                Invalidate();
            }
        }

        private void WindowVSR_ScreenshotCrop_MouseUp(object sender, MouseEventArgs
e)
        {
            if(e.Button == MouseButtons.Left)
            {
                isMousePressed = false;
                Invalidate();

                if(IsLegalSize(ScreenshotSelectedRectangle.Size))
                {
                    ResultImage = new Bitmap(ScreenshotSelectedRectangle.Width,
ScreenshotSelectedRectangle.Height);
```

```
                    Graphics.FromImage(ResultImage).DrawImage(BackgroundImage, 0, 0,
ScreenshotSelectedRectangle, GraphicsUnit.Pixel);

                    this.Close();
                }
            }
        }

        private void DrawSelection(PaintEventArgs e)
        {
            Region ScreenshotArea = new Region(new Rectangle(0, 0, this.Width,
this.Height));
            if(isMousePressed)
                ScreenshotArea.Exclude(ScreenshotSelectedRectangle);

            Brush RegionBrush = new SolidBrush(Color.FromArgb(191, Color.Black));
            Pen BorderPen = new Pen(new SolidBrush(Color.FromArgb(255, 64, 255,
64)));

            e.Graphics.FillRegion(RegionBrush, ScreenshotArea);
            if(isMousePressed)
                e.Graphics.DrawRectangle(BorderPen, ScreenshotSelectedRectangle.X,
ScreenshotSelectedRectangle.Y,
                                                    ScreenshotSelectedRectangle.Width
- 1, ScreenshotSelectedRectangle.Height - 1);
        }

        private bool IsLegalSize(Size Nominal)
        {
            if (MIN_SIZE.Width > Nominal.Width || MIN_SIZE.Height > Nominal.Height)
                return false;
            else
                return true;
        }
    }
}
```

WindowVSR_ScreenshotCrop.Designer.cs

```
namespace VarietyScreenRecorder.WindowVSR
{
    partial class WindowVSR_ScreenshotCrop
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
```

```
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.SuspendLayout();
            //
            // WindowVSR_ScreenshotCrop
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(800, 450);
            this.Cursor = System.Windows.Forms.Cursors.Cross;
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.None;
            this.Name = "WindowVSR_ScreenshotCrop";
            this.Text = "Variety Screen Recorder";
            this.WindowState = System.Windows.Forms.FormWindowState.Maximized;
            this.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.WindowVSR_ScreenshotCrop_MouseDown);
            this.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.WindowVSR_ScreenshotCrop_MouseMove);
            this.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.WindowVSR_ScreenshotCrop_MouseUp);
            this.ResumeLayout(false);

        }

        #endregion
    }
}
```

WindowVSR_Settings.cs

```
using System;
using System.Windows.Forms;
using VarietyScreenRecorder.ExtraClass;

namespace VarietyScreenRecorder.WindowVSR
{
    public partial class WindowVSR_Settings : Form
    {
        public WindowVSR_Settings()
        {
            this.Icon = Properties.Resources.Logo;

            InitializeComponent();

            GetMainSettings();
            GetHotKeys();
        }
```

```
        private void SetDefaultMainSettings()
        {
            Properties.Settings.Default.FullSizeScreenshot_Path = "";
            Properties.Settings.Default.FullSizeScreenshot_Extension = "png";

            Properties.Settings.Default.SizedScreenshot_Path = "";
            Properties.Settings.Default.SizedScreenshot_Extension = "png";
            Properties.Settings.Default.SizedScreenshot_IsEdit = true;

            Properties.Settings.Default.TimeLapseScreenshot_Path = "";
            Properties.Settings.Default.TimeLapseScreenshot_Extension = "png";
            Properties.Settings.Default.TimeLapseScreenshot_Interval = 5;

            Properties.Settings.Default.VideoRecord_Path = "";

            Properties.Settings.Default.Save();
        }

        private void SetDefaultHotKeys()
        {
            Properties.Settings.Default.VSRHotKey_PhotoCam = "Ctrl+P";
            Properties.Settings.Default.VSRHotKey_PhotoCamSized = "Ctrl+C";
            Properties.Settings.Default.VSRHotKey_TimeLapse = "Ctrl+T";
            Properties.Settings.Default.VSRHotKey_VideoCam = "Ctrl+V";
            Properties.Settings.Default.VSRHotKey_Setting = "Ctrl+S";
            Properties.Settings.Default.VSRHotKey_Info = "Ctrl+I";

            Properties.Settings.Default.Save();
        }

        private void GetMainSettings()
        {
            tb_FullSizeScreenshot_Path.Text =
Properties.Settings.Default.FullSizeScreenshot_Path == "" ?
Environment.CurrentDirectory + "\\Screenshots" :
Properties.Settings.Default.FullSizeScreenshot_Path;
            cb_FullSizeScreenshot_Extension.SelectedItem =
Properties.Settings.Default.FullSizeScreenshot_Extension;

            tb_SizedScreenshot_Path.Text =
Properties.Settings.Default.SizedScreenshot_Path == "" ? Environment.CurrentDirectory
+ "\\SizedScreenshots" : Properties.Settings.Default.SizedScreenshot_Path;
            cb_SizedScreenshot_Extension.SelectedItem =
Properties.Settings.Default.SizedScreenshot_Extension;
            cb_SizedScreenshot_IsEdit.Checked =
Properties.Settings.Default.SizedScreenshot_IsEdit;

            tb_TimeLapseScreenshot_Path.Text =
Properties.Settings.Default.TimeLapseScreenshot_Path == "" ?
Environment.CurrentDirectory + "\\TimeLapses" :
Properties.Settings.Default.TimeLapseScreenshot_Path;
            cb_TimeLapseScreenshot_Extension.SelectedItem =
Properties.Settings.Default.TimeLapseScreenshot_Extension;
            tb_TimeLapseScreenshot_Interval.Value =
Properties.Settings.Default.TimeLapseScreenshot_Interval;
            nud_TimeLapseScreenshot_Interval.Value =
Properties.Settings.Default.TimeLapseScreenshot_Interval;

            tb_VideoRecord_Path.Text = Properties.Settings.Default.VideoRecord_Path
== "" ? Environment.CurrentDirectory + "\\VideoRecords" :
Properties.Settings.Default.VideoRecord_Path;
        }
```

```
        private void GetHotKeys()
        {
            tb_VSRHotKey_PhotoCam.Text =
Properties.Settings.Default.VSRHotKey_PhotoCam;
            tb_VSRHotKey_PhotoCamSized.Text =
Properties.Settings.Default.VSRHotKey_PhotoCamSized;
            tb_VSRHotKey_TimeLapse.Text =
Properties.Settings.Default.VSRHotKey_TimeLapse;
            tb_VSRHotKey_VideoCam.Text =
Properties.Settings.Default.VSRHotKey_VideoCam;
            tb_VSRHotKey_Setting.Text =
Properties.Settings.Default.VSRHotKey_Setting;
            tb_VSRHotKey_Info.Text = Properties.Settings.Default.VSRHotKey_Info;
        }

        private void SaveMainSettings()
        {
            Properties.Settings.Default.FullSizeScreenshot_Path =
tb_FullSizeScreenshot_Path.Text == Environment.CurrentDirectory + "\\Screenshots" ?
"" : tb_FullSizeScreenshot_Path.Text;
            Properties.Settings.Default.FullSizeScreenshot_Extension =
(string)cb_FullSizeScreenshot_Extension.SelectedItem;

            Properties.Settings.Default.SizedScreenshot_Path =
tb_SizedScreenshot_Path.Text == Environment.CurrentDirectory + "\\SizedScreenshots" ?
"" : tb_SizedScreenshot_Path.Text;
            Properties.Settings.Default.SizedScreenshot_Extension =
(string)cb_SizedScreenshot_Extension.SelectedItem;
            Properties.Settings.Default.SizedScreenshot_IsEdit =
cb_SizedScreenshot_IsEdit.Checked;

            Properties.Settings.Default.TimeLapseScreenshot_Path =
tb_TimeLapseScreenshot_Path.Text == Environment.CurrentDirectory + "\\TimeLapses" ?
"" : tb_TimeLapseScreenshot_Path.Text;
            Properties.Settings.Default.TimeLapseScreenshot_Extension =
(string)cb_TimeLapseScreenshot_Extension.SelectedItem;
            Properties.Settings.Default.TimeLapseScreenshot_Interval =
tb_TimeLapseScreenshot_Interval.Value;
            Properties.Settings.Default.VideoRecord_Path = tb_VideoRecord_Path.Text
== Environment.CurrentDirectory + "\\VideoRecords" ? "" : tb_VideoRecord_Path.Text;

            Properties.Settings.Default.Save();
        }

        private void SaveHotKeys()
        {
            Properties.Settings.Default.VSRHotKey_PhotoCam =
tb_VSRHotKey_PhotoCam.Text;
            Properties.Settings.Default.VSRHotKey_PhotoCamSized =
tb_VSRHotKey_PhotoCamSized.Text;
            Properties.Settings.Default.VSRHotKey_TimeLapse =
tb_VSRHotKey_TimeLapse.Text;
            Properties.Settings.Default.VSRHotKey_VideoCam =
tb_VSRHotKey_VideoCam.Text;
            Properties.Settings.Default.VSRHotKey_Setting =
tb_VSRHotKey_Setting.Text;
            Properties.Settings.Default.VSRHotKey_Info = tb_VSRHotKey_Info.Text;

            Properties.Settings.Default.Save();
        }

        private string GetNewPath(string CurrentPath)
```

```
        {
            string NewPath = "";

            FolderBrowserDialog NewFolderWindow = new FolderBrowserDialog();
            NewFolderWindow.SelectedPath = CurrentPath;
            NewFolderWindow.Description = "Choose folder...";

            if (NewFolderWindow.ShowDialog() == DialogResult.OK)
                NewPath = NewFolderWindow.SelectedPath;
            else
                NewPath = CurrentPath;

            return NewPath;
        }

        private void b_FullSizeScreenshot_Path_Click(object sender, EventArgs e)
        {
            tb_FullSizeScreenshot_Path.Text =
GetNewPath(tb_FullSizeScreenshot_Path.Text);
        }

        private void b_SizedScreenshot_Path_Click(object sender, EventArgs e)
        {
            tb_SizedScreenshot_Path.Text = GetNewPath(tb_SizedScreenshot_Path.Text);
        }

        private void b_TimeLapseScreenshot_Path_Click(object sender, EventArgs e)
        {
            tb_TimeLapseScreenshot_Path.Text =
GetNewPath(tb_TimeLapseScreenshot_Path.Text);
        }

        private void b_VideoRecord_Path_Click(object sender, EventArgs e)
        {
            tb_VideoRecord_Path.Text = GetNewPath(tb_VideoRecord_Path.Text);
        }

        private void tb_TimeLapseScreenshot_Interval_Scroll(object sender, EventArgs
e)
        {
            nud_TimeLapseScreenshot_Interval.Value =
tb_TimeLapseScreenshot_Interval.Value;
        }

        private void nud_TimeLapseScreenshot_Interval_ValueChanged(object sender,
EventArgs e)
        {
            tb_TimeLapseScreenshot_Interval.Value =
(int)nud_TimeLapseScreenshot_Interval.Value;
        }

        private void b_DefaultMainSetting_Click(object sender, EventArgs e)
        {
            SetDefaultMainSettings();

            GetMainSettings();
        }

        private void b_CancelMainSetting_Click(object sender, EventArgs e)
        {
            GetMainSettings();
        }
```

```
        private void b_SaveMainSettings_Click(object sender, EventArgs e)
        {
            SaveMainSettings();

            GetMainSettings();
        }


        private void DiscardHotKey(string HotKey)
        {
            if (tb_VSRHotKey_PhotoCam.Text == HotKey) tb_VSRHotKey_PhotoCam.Text =
"";
            if (tb_VSRHotKey_PhotoCamSized.Text == HotKey)
tb_VSRHotKey_PhotoCamSized.Text = "";
            if (tb_VSRHotKey_TimeLapse.Text == HotKey) tb_VSRHotKey_TimeLapse.Text =
"";
            if (tb_VSRHotKey_VideoCam.Text == HotKey) tb_VSRHotKey_VideoCam.Text =
"";
            if (tb_VSRHotKey_Setting.Text == HotKey) tb_VSRHotKey_Setting.Text = "";
            if (tb_VSRHotKey_Info.Text == HotKey) tb_VSRHotKey_Info.Text = "";
        }


        private string GetNewHotKey(KeyEventArgs e)
        {
            string NewHotKey = "";

            if (HotKeyManager.isHotKey(e))
            {
                NewHotKey = HotKeyManager.GetHotKey(e);
                DiscardHotKey(NewHotKey);
            }

            return NewHotKey;
        }

        private void tb_VSRHotKey_PhotoCam_KeyDown(object sender, KeyEventArgs e)
        {
            tb_VSRHotKey_PhotoCam.Text = GetNewHotKey(e);
        }

        private void tb_VSRHotKey_PhotoCamSized_KeyDown(object sender, KeyEventArgs
e)
        {
            tb_VSRHotKey_PhotoCamSized.Text = GetNewHotKey(e);
        }

        private void tb_VSRHotKey_TimeLapse_KeyDown(object sender, KeyEventArgs e)
        {
            tb_VSRHotKey_TimeLapse.Text = GetNewHotKey(e);
        }

        private void tb_VSRHotKey_VideoCam_KeyDown(object sender, KeyEventArgs e)
        {
            tb_VSRHotKey_VideoCam.Text = GetNewHotKey(e);
        }

        private void tb_VSRHotKey_Setting_KeyDown(object sender, KeyEventArgs e)
        {
            tb_VSRHotKey_Setting.Text = GetNewHotKey(e);
        }
```

```
        private void tb_VSRHotKey_Info_KeyDown(object sender, KeyEventArgs e)
        {
            tb_VSRHotKey_Info.Text = GetNewHotKey(e);
        }

        private void b_DefaultHotKeys_Click(object sender, EventArgs e)
        {
            SetDefaultHotKeys();

            GetHotKeys();
        }

        private void b_CancelHotKeys_Click(object sender, EventArgs e)
        {
            GetHotKeys();
        }

        private void b_SaveHotKeys_Click(object sender, EventArgs e)
        {
            SaveHotKeys();

            GetHotKeys();
        }
    }
}
```

WindowVSR_Settings.Designer.cs

```
namespace VarietyScreenRecorder.WindowVSR
{
    partial class WindowVSR_Settings
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
```

```
            this.tlp_AllContentAlignment = new
System.Windows.Forms.TableLayoutPanel();
            this.tc_SettingsMenu = new System.Windows.Forms.TabControl();
            this.tp_MainSetting = new System.Windows.Forms.TabPage();
            this.tlp_MainSettings = new System.Windows.Forms.TableLayoutPanel();
            this.b_FullSizeScreenshot_Path = new System.Windows.Forms.Button();
            this.tb_FullSizeScreenshot_Path = new System.Windows.Forms.TextBox();
            this.cb_FullSizeScreenshot_Extension = new
System.Windows.Forms.ComboBox();
            this.tb_SizedScreenshot_Path = new System.Windows.Forms.TextBox();
            this.cb_SizedScreenshot_Extension = new System.Windows.Forms.ComboBox();
            this.cb_SizedScreenshot_IsEdit = new System.Windows.Forms.CheckBox();
            this.tb_TimeLapseScreenshot_Path = new System.Windows.Forms.TextBox();
            this.cb_TimeLapseScreenshot_Extension = new
System.Windows.Forms.ComboBox();
            this.tb_TimeLapseScreenshot_Interval = new
System.Windows.Forms.TrackBar();
            this.nud_TimeLapseScreenshot_Interval = new
System.Windows.Forms.NumericUpDown();
            this.tb_VideoRecord_Path = new System.Windows.Forms.TextBox();
            this.b_SizedScreenshot_Path = new System.Windows.Forms.Button();
            this.b_TimeLapseScreenshot_Path = new System.Windows.Forms.Button();
            this.b_VideoRecord_Path = new System.Windows.Forms.Button();
            this.b_CancelMainSetting = new System.Windows.Forms.Button();
            this.b_SaveMainSettings = new System.Windows.Forms.Button();
            this.b_DefaultMainSetting = new System.Windows.Forms.Button();
            this.l_FullSizeScreenshot_Path = new System.Windows.Forms.Label();
            this.l_FullSizeScreenshot_Extension = new System.Windows.Forms.Label();
            this.l_SizedScreenshot_Path = new System.Windows.Forms.Label();
            this.l_SizedScreenshot_Extension = new System.Windows.Forms.Label();
            this.l_SizedScreenshot_IsEdit = new System.Windows.Forms.Label();
            this.l_TimeLapseScreenshot_Path = new System.Windows.Forms.Label();
            this.l_TimeLapseScreenshot_Extension = new System.Windows.Forms.Label();
            this.l_TimeLapseScreenshot_Interval = new System.Windows.Forms.Label();
            this.l_VideoRecord_Path = new System.Windows.Forms.Label();
            this.tp_HotKeys = new System.Windows.Forms.TabPage();
            this.tlp_HotKeys = new System.Windows.Forms.TableLayoutPanel();
            this.l_VSRHotKey_PhotoCam = new System.Windows.Forms.Label();
            this.l_VSRHotKey_PhotoCamSized = new System.Windows.Forms.Label();
            this.l_VSRHotKey_TimeLapse = new System.Windows.Forms.Label();
            this.l_VSRHotKey_VideoCam = new System.Windows.Forms.Label();
            this.l_VSRHotKey_Setting = new System.Windows.Forms.Label();
            this.l_VSRHotKey_Info = new System.Windows.Forms.Label();
            this.tb_VSRHotKey_PhotoCam = new System.Windows.Forms.TextBox();
            this.tb_VSRHotKey_PhotoCamSized = new System.Windows.Forms.TextBox();
            this.tb_VSRHotKey_TimeLapse = new System.Windows.Forms.TextBox();
            this.tb_VSRHotKey_VideoCam = new System.Windows.Forms.TextBox();
            this.tb_VSRHotKey_Setting = new System.Windows.Forms.TextBox();
            this.tb_VSRHotKey_Info = new System.Windows.Forms.TextBox();
            this.b_SaveHotKeys = new System.Windows.Forms.Button();
            this.b_CancelHotKeys = new System.Windows.Forms.Button();
            this.b_DefaultHotKeys = new System.Windows.Forms.Button();
            this.tlp_AllContentAlignment.SuspendLayout();
            this.tc_SettingsMenu.SuspendLayout();
            this.tp_MainSetting.SuspendLayout();
            this.tlp_MainSettings.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.tb_TimeLapseScreenshot_Interval)).Be
ginInit();

((System.ComponentModel.ISupportInitialize)(this.nud_TimeLapseScreenshot_Interval)).B
eginInit();
```

```
            this.tp_HotKeys.SuspendLayout();
            this.tlp_HotKeys.SuspendLayout();
            this.SuspendLayout();
            //
            // tlp_AllContentAlignment
            //
            this.tlp_AllContentAlignment.ColumnCount = 3;
            this.tlp_AllContentAlignment.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_AllContentAlignment.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 100F));
            this.tlp_AllContentAlignment.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_AllContentAlignment.Controls.Add(this.tc_SettingsMenu, 1, 1);
            this.tlp_AllContentAlignment.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tlp_AllContentAlignment.Location = new System.Drawing.Point(0, 0);
            this.tlp_AllContentAlignment.Name = "tlp_AllContentAlignment";
            this.tlp_AllContentAlignment.RowCount = 3;
            this.tlp_AllContentAlignment.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_AllContentAlignment.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 100F));
            this.tlp_AllContentAlignment.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_AllContentAlignment.Size = new System.Drawing.Size(734, 461);
            this.tlp_AllContentAlignment.TabIndex = 0;
            //
            // tc_SettingsMenu
            //
            this.tc_SettingsMenu.Controls.Add(this.tp_MainSetting);
            this.tc_SettingsMenu.Controls.Add(this.tp_HotKeys);
            this.tc_SettingsMenu.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tc_SettingsMenu.Location = new System.Drawing.Point(13, 13);
            this.tc_SettingsMenu.Name = "tc_SettingsMenu";
            this.tc_SettingsMenu.SelectedIndex = 0;
            this.tc_SettingsMenu.Size = new System.Drawing.Size(708, 435);
            this.tc_SettingsMenu.TabIndex = 0;
            //
            // tp_MainSetting
            //
            this.tp_MainSetting.Controls.Add(this.tlp_MainSettings);
            this.tp_MainSetting.Location = new System.Drawing.Point(4, 22);
            this.tp_MainSetting.Name = "tp_MainSetting";
            this.tp_MainSetting.Padding = new System.Windows.Forms.Padding(3);
            this.tp_MainSetting.Size = new System.Drawing.Size(700, 409);
            this.tp_MainSetting.TabIndex = 0;
            this.tp_MainSetting.Text = "Main settings";
            this.tp_MainSetting.UseVisualStyleBackColor = true;
            //
            // tlp_MainSettings
            //
            this.tlp_MainSettings.AutoScroll = true;
            this.tlp_MainSettings.ColumnCount = 9;
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 40F));
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 20F));
```

```
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 20F));
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 20F));
            this.tlp_MainSettings.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 14F));
            this.tlp_MainSettings.Controls.Add(this.b_FullSizeScreenshot_Path, 7, 1);
            this.tlp_MainSettings.Controls.Add(this.tb_FullSizeScreenshot_Path, 3,
1);
            this.tlp_MainSettings.Controls.Add(this.cb_FullSizeScreenshot_Extension,
3, 3);
            this.tlp_MainSettings.Controls.Add(this.tb_SizedScreenshot_Path, 3, 5);
            this.tlp_MainSettings.Controls.Add(this.cb_SizedScreenshot_Extension, 3,
7);
            this.tlp_MainSettings.Controls.Add(this.cb_SizedScreenshot_IsEdit, 3, 9);
            this.tlp_MainSettings.Controls.Add(this.tb_TimeLapseScreenshot_Path, 3,
11);
            this.tlp_MainSettings.Controls.Add(this.cb_TimeLapseScreenshot_Extension,
3, 13);
            this.tlp_MainSettings.Controls.Add(this.tb_TimeLapseScreenshot_Interval,
3, 15);
            this.tlp_MainSettings.Controls.Add(this.nud_TimeLapseScreenshot_Interval,
7, 15);
            this.tlp_MainSettings.Controls.Add(this.tb_VideoRecord_Path, 3, 17);
            this.tlp_MainSettings.Controls.Add(this.b_SizedScreenshot_Path, 7, 5);
            this.tlp_MainSettings.Controls.Add(this.b_TimeLapseScreenshot_Path, 7,
11);
            this.tlp_MainSettings.Controls.Add(this.b_VideoRecord_Path, 7, 17);
            this.tlp_MainSettings.Controls.Add(this.b_CancelMainSetting, 5, 19);
            this.tlp_MainSettings.Controls.Add(this.b_SaveMainSettings, 3, 19);
            this.tlp_MainSettings.Controls.Add(this.b_DefaultMainSetting, 7, 19);
            this.tlp_MainSettings.Controls.Add(this.l_FullSizeScreenshot_Path, 1, 1);
            this.tlp_MainSettings.Controls.Add(this.l_FullSizeScreenshot_Extension,
1, 3);
            this.tlp_MainSettings.Controls.Add(this.l_SizedScreenshot_Path, 1, 5);
            this.tlp_MainSettings.Controls.Add(this.l_SizedScreenshot_Extension, 1,
7);
            this.tlp_MainSettings.Controls.Add(this.l_SizedScreenshot_IsEdit, 1, 9);
            this.tlp_MainSettings.Controls.Add(this.l_TimeLapseScreenshot_Path, 1,
11);
            this.tlp_MainSettings.Controls.Add(this.l_TimeLapseScreenshot_Extension,
1, 13);
            this.tlp_MainSettings.Controls.Add(this.l_TimeLapseScreenshot_Interval,
1, 15);
            this.tlp_MainSettings.Controls.Add(this.l_VideoRecord_Path, 1, 17);
            this.tlp_MainSettings.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tlp_MainSettings.Location = new System.Drawing.Point(3, 3);
            this.tlp_MainSettings.Name = "tlp_MainSettings";
            this.tlp_MainSettings.Padding = new System.Windows.Forms.Padding(0, 0,
20, 0);
            this.tlp_MainSettings.RowCount = 21;
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
```

```
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 36F));
            this.tlp_MainSettings.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainSettings.Size = new System.Drawing.Size(694, 403);
            this.tlp_MainSettings.TabIndex = 0;
            //
            // b_FullSizeScreenshot_Path
            //
            this.b_FullSizeScreenshot_Path.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.b_FullSizeScreenshot_Path.Location = new System.Drawing.Point(539,
13);
            this.b_FullSizeScreenshot_Path.Name = "b_FullSizeScreenshot_Path";
            this.b_FullSizeScreenshot_Path.Size = new System.Drawing.Size(118, 20);
            this.b_FullSizeScreenshot_Path.TabIndex = 0;
            this.b_FullSizeScreenshot_Path.Text = "Browse";
            this.b_FullSizeScreenshot_Path.UseVisualStyleBackColor = true;
            this.b_FullSizeScreenshot_Path.Click += new
System.EventHandler(this.b_FullSizeScreenshot_Path_Click);
            //
            // tb_FullSizeScreenshot_Path
            //
            this.tlp_MainSettings.SetColumnSpan(this.tb_FullSizeScreenshot_Path, 3);
            this.tb_FullSizeScreenshot_Path.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.tb_FullSizeScreenshot_Path.Location = new System.Drawing.Point(271,
13);
            this.tb_FullSizeScreenshot_Path.Name = "tb_FullSizeScreenshot_Path";
            this.tb_FullSizeScreenshot_Path.ReadOnly = true;
```

```
            this.tb_FullSizeScreenshot_Path.Size = new System.Drawing.Size(252, 20);
            this.tb_FullSizeScreenshot_Path.TabIndex = 1;
            //
            // cb_FullSizeScreenshot_Extension
            //
            this.cb_FullSizeScreenshot_Extension.BackColor =
System.Drawing.Color.White;
            this.cb_FullSizeScreenshot_Extension.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.cb_FullSizeScreenshot_Extension.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cb_FullSizeScreenshot_Extension.FormattingEnabled = true;
            this.cb_FullSizeScreenshot_Extension.Items.AddRange(new object[] {
            "png",
            "jpeg",
            "bmp",
            "gif"});
            this.cb_FullSizeScreenshot_Extension.Location = new
System.Drawing.Point(271, 49);
            this.cb_FullSizeScreenshot_Extension.Name =
"cb_FullSizeScreenshot_Extension";
            this.cb_FullSizeScreenshot_Extension.Size = new System.Drawing.Size(118,
21);
            this.cb_FullSizeScreenshot_Extension.TabIndex = 2;
            //
            // tb_SizedScreenshot_Path
            //
            this.tlp_MainSettings.SetColumnSpan(this.tb_SizedScreenshot_Path, 3);
            this.tb_SizedScreenshot_Path.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_SizedScreenshot_Path.Location = new System.Drawing.Point(271,
85);
            this.tb_SizedScreenshot_Path.Name = "tb_SizedScreenshot_Path";
            this.tb_SizedScreenshot_Path.ReadOnly = true;
            this.tb_SizedScreenshot_Path.Size = new System.Drawing.Size(252, 20);
            this.tb_SizedScreenshot_Path.TabIndex = 3;
            //
            // cb_SizedScreenshot_Extension
            //
            this.cb_SizedScreenshot_Extension.BackColor = System.Drawing.Color.White;
            this.cb_SizedScreenshot_Extension.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.cb_SizedScreenshot_Extension.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cb_SizedScreenshot_Extension.FormattingEnabled = true;
            this.cb_SizedScreenshot_Extension.Items.AddRange(new object[] {
            "png",
            "jpeg",
            "bmp",
            "gif"});
            this.cb_SizedScreenshot_Extension.Location = new
System.Drawing.Point(271, 121);
            this.cb_SizedScreenshot_Extension.Name = "cb_SizedScreenshot_Extension";
            this.cb_SizedScreenshot_Extension.Size = new System.Drawing.Size(118,
21);
            this.cb_SizedScreenshot_Extension.TabIndex = 4;
            //
            // cb_SizedScreenshot_IsEdit
            //
            this.cb_SizedScreenshot_IsEdit.AutoSize = true;
            this.cb_SizedScreenshot_IsEdit.Dock =
System.Windows.Forms.DockStyle.Fill;
```

```
            this.cb_SizedScreenshot_IsEdit.Location = new System.Drawing.Point(271,
157);
            this.cb_SizedScreenshot_IsEdit.Name = "cb_SizedScreenshot_IsEdit";
            this.cb_SizedScreenshot_IsEdit.Size = new System.Drawing.Size(118, 20);
            this.cb_SizedScreenshot_IsEdit.TabIndex = 5;
            this.cb_SizedScreenshot_IsEdit.UseVisualStyleBackColor = true;
            //
            // tb_TimeLapseScreenshot_Path
            //
            this.tlp_MainSettings.SetColumnSpan(this.tb_TimeLapseScreenshot_Path, 3);
            this.tb_TimeLapseScreenshot_Path.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.tb_TimeLapseScreenshot_Path.Location = new System.Drawing.Point(271,
193);
            this.tb_TimeLapseScreenshot_Path.Name = "tb_TimeLapseScreenshot_Path";
            this.tb_TimeLapseScreenshot_Path.ReadOnly = true;
            this.tb_TimeLapseScreenshot_Path.Size = new System.Drawing.Size(252, 20);
            this.tb_TimeLapseScreenshot_Path.TabIndex = 6;
            //
            // cb_TimeLapseScreenshot_Extension
            //
            this.cb_TimeLapseScreenshot_Extension.BackColor =
System.Drawing.Color.White;
            this.cb_TimeLapseScreenshot_Extension.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.cb_TimeLapseScreenshot_Extension.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
            this.cb_TimeLapseScreenshot_Extension.FormattingEnabled = true;
            this.cb_TimeLapseScreenshot_Extension.Items.AddRange(new object[] {
            "png",
            "jpeg",
            "bmp",
            "gif"});
            this.cb_TimeLapseScreenshot_Extension.Location = new
System.Drawing.Point(271, 229);
            this.cb_TimeLapseScreenshot_Extension.Name =
"cb_TimeLapseScreenshot_Extension";
            this.cb_TimeLapseScreenshot_Extension.Size = new System.Drawing.Size(118,
21);
            this.cb_TimeLapseScreenshot_Extension.TabIndex = 7;
            //
            // tb_TimeLapseScreenshot_Interval
            //
            this.tlp_MainSettings.SetColumnSpan(this.tb_TimeLapseScreenshot_Interval,
3);
            this.tb_TimeLapseScreenshot_Interval.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.tb_TimeLapseScreenshot_Interval.LargeChange = 10;
            this.tb_TimeLapseScreenshot_Interval.Location = new
System.Drawing.Point(271, 265);
            this.tb_TimeLapseScreenshot_Interval.Maximum = 3600;
            this.tb_TimeLapseScreenshot_Interval.Minimum = 1;
            this.tb_TimeLapseScreenshot_Interval.Name =
"tb_TimeLapseScreenshot_Interval";
            this.tb_TimeLapseScreenshot_Interval.Size = new System.Drawing.Size(252,
20);
            this.tb_TimeLapseScreenshot_Interval.TabIndex = 8;
            this.tb_TimeLapseScreenshot_Interval.Value = 1;
            this.tb_TimeLapseScreenshot_Interval.Scroll += new
System.EventHandler(this.tb_TimeLapseScreenshot_Interval_Scroll);
            //
            // nud_TimeLapseScreenshot_Interval
```

```
            //
            this.nud_TimeLapseScreenshot_Interval.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.nud_TimeLapseScreenshot_Interval.Location = new
System.Drawing.Point(539, 265);
            this.nud_TimeLapseScreenshot_Interval.Maximum = new decimal(new int[] {
            3600,
            0,
            0,
            0});
            this.nud_TimeLapseScreenshot_Interval.Minimum = new decimal(new int[] {
            1,
            0,
            0,
            0});
            this.nud_TimeLapseScreenshot_Interval.Name =
"nud_TimeLapseScreenshot_Interval";
            this.nud_TimeLapseScreenshot_Interval.Size = new System.Drawing.Size(118,
20);
            this.nud_TimeLapseScreenshot_Interval.TabIndex = 9;
            this.nud_TimeLapseScreenshot_Interval.Value = new decimal(new int[] {
            1,
            0,
            0,
            0});
            this.nud_TimeLapseScreenshot_Interval.ValueChanged += new
System.EventHandler(this.nud_TimeLapseScreenshot_Interval_ValueChanged);
            //
            // tb_VideoRecord_Path
            //
            this.tlp_MainSettings.SetColumnSpan(this.tb_VideoRecord_Path, 3);
            this.tb_VideoRecord_Path.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_VideoRecord_Path.Location = new System.Drawing.Point(271, 301);
            this.tb_VideoRecord_Path.Name = "tb_VideoRecord_Path";
            this.tb_VideoRecord_Path.ReadOnly = true;
            this.tb_VideoRecord_Path.Size = new System.Drawing.Size(252, 20);
            this.tb_VideoRecord_Path.TabIndex = 10;
            //
            // b_SizedScreenshot_Path
            //
            this.b_SizedScreenshot_Path.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_SizedScreenshot_Path.Location = new System.Drawing.Point(539, 85);
            this.b_SizedScreenshot_Path.Name = "b_SizedScreenshot_Path";
            this.b_SizedScreenshot_Path.Size = new System.Drawing.Size(118, 20);
            this.b_SizedScreenshot_Path.TabIndex = 11;
            this.b_SizedScreenshot_Path.Text = "Browse";
            this.b_SizedScreenshot_Path.UseVisualStyleBackColor = true;
            this.b_SizedScreenshot_Path.Click += new
System.EventHandler(this.b_SizedScreenshot_Path_Click);
            //
            // b_TimeLapseScreenshot_Path
            //
            this.b_TimeLapseScreenshot_Path.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.b_TimeLapseScreenshot_Path.Location = new System.Drawing.Point(539,
193);
            this.b_TimeLapseScreenshot_Path.Name = "b_TimeLapseScreenshot_Path";
            this.b_TimeLapseScreenshot_Path.Size = new System.Drawing.Size(118, 20);
            this.b_TimeLapseScreenshot_Path.TabIndex = 12;
            this.b_TimeLapseScreenshot_Path.Text = "Browse";
            this.b_TimeLapseScreenshot_Path.UseVisualStyleBackColor = true;
```

```
            this.b_TimeLapseScreenshot_Path.Click += new
System.EventHandler(this.b_TimeLapseScreenshot_Path_Click);
            //
            // b_VideoRecord_Path
            //
            this.b_VideoRecord_Path.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_VideoRecord_Path.Location = new System.Drawing.Point(539, 301);
            this.b_VideoRecord_Path.Name = "b_VideoRecord_Path";
            this.b_VideoRecord_Path.Size = new System.Drawing.Size(118, 20);
            this.b_VideoRecord_Path.TabIndex = 13;
            this.b_VideoRecord_Path.Text = "Browse";
            this.b_VideoRecord_Path.UseVisualStyleBackColor = true;
            this.b_VideoRecord_Path.Click += new
System.EventHandler(this.b_VideoRecord_Path_Click);
            //
            // b_CancelMainSetting
            //
            this.b_CancelMainSetting.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_CancelMainSetting.Location = new System.Drawing.Point(405, 353);
            this.b_CancelMainSetting.Name = "b_CancelMainSetting";
            this.b_CancelMainSetting.Size = new System.Drawing.Size(118, 30);
            this.b_CancelMainSetting.TabIndex = 15;
            this.b_CancelMainSetting.Text = "Cancel";
            this.b_CancelMainSetting.UseVisualStyleBackColor = true;
            this.b_CancelMainSetting.Click += new
System.EventHandler(this.b_CancelMainSetting_Click);
            //
            // b_SaveMainSettings
            //
            this.b_SaveMainSettings.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_SaveMainSettings.Location = new System.Drawing.Point(271, 353);
            this.b_SaveMainSettings.Name = "b_SaveMainSettings";
            this.b_SaveMainSettings.Size = new System.Drawing.Size(118, 30);
            this.b_SaveMainSettings.TabIndex = 14;
            this.b_SaveMainSettings.Text = "Save";
            this.b_SaveMainSettings.UseVisualStyleBackColor = true;
            this.b_SaveMainSettings.Click += new
System.EventHandler(this.b_SaveMainSettings_Click);
            //
            // b_DefaultMainSetting
            //
            this.b_DefaultMainSetting.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_DefaultMainSetting.Location = new System.Drawing.Point(539, 353);
            this.b_DefaultMainSetting.Name = "b_DefaultMainSetting";
            this.b_DefaultMainSetting.Size = new System.Drawing.Size(118, 30);
            this.b_DefaultMainSetting.TabIndex = 16;
            this.b_DefaultMainSetting.Text = "Default";
            this.b_DefaultMainSetting.UseVisualStyleBackColor = true;
            this.b_DefaultMainSetting.Click += new
System.EventHandler(this.b_DefaultMainSetting_Click);
            //
            // l_FullSizeScreenshot_Path
            //
            this.l_FullSizeScreenshot_Path.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.l_FullSizeScreenshot_Path.Location = new System.Drawing.Point(13,
10);
            this.l_FullSizeScreenshot_Path.Name = "l_FullSizeScreenshot_Path";
            this.l_FullSizeScreenshot_Path.Size = new System.Drawing.Size(242, 26);
            this.l_FullSizeScreenshot_Path.TabIndex = 0;
            this.l_FullSizeScreenshot_Path.Text = "Folder path for saving quick
screenshots";
```

```
            this.l_FullSizeScreenshot_Path.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_FullSizeScreenshot_Extension
            //
            this.l_FullSizeScreenshot_Extension.AutoSize = true;
            this.l_FullSizeScreenshot_Extension.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.l_FullSizeScreenshot_Extension.Location = new
System.Drawing.Point(13, 46);
            this.l_FullSizeScreenshot_Extension.Name =
"l_FullSizeScreenshot_Extension";
            this.l_FullSizeScreenshot_Extension.Size = new System.Drawing.Size(242,
26);
            this.l_FullSizeScreenshot_Extension.TabIndex = 17;
            this.l_FullSizeScreenshot_Extension.Text = "Quick screenshots extension";
            this.l_FullSizeScreenshot_Extension.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_SizedScreenshot_Path
            //
            this.l_SizedScreenshot_Path.AutoSize = true;
            this.l_SizedScreenshot_Path.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_SizedScreenshot_Path.Location = new System.Drawing.Point(13, 82);
            this.l_SizedScreenshot_Path.Name = "l_SizedScreenshot_Path";
            this.l_SizedScreenshot_Path.Size = new System.Drawing.Size(242, 26);
            this.l_SizedScreenshot_Path.TabIndex = 18;
            this.l_SizedScreenshot_Path.Text = "Folder path for custom screenshots";
            this.l_SizedScreenshot_Path.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_SizedScreenshot_Extension
            //
            this.l_SizedScreenshot_Extension.AutoSize = true;
            this.l_SizedScreenshot_Extension.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.l_SizedScreenshot_Extension.Location = new System.Drawing.Point(13,
118);
            this.l_SizedScreenshot_Extension.Name = "l_SizedScreenshot_Extension";
            this.l_SizedScreenshot_Extension.Size = new System.Drawing.Size(242, 26);
            this.l_SizedScreenshot_Extension.TabIndex = 19;
            this.l_SizedScreenshot_Extension.Text = "Custom screenshots extension";
            this.l_SizedScreenshot_Extension.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_SizedScreenshot_IsEdit
            //
            this.l_SizedScreenshot_IsEdit.AutoSize = true;
            this.l_SizedScreenshot_IsEdit.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_SizedScreenshot_IsEdit.Location = new System.Drawing.Point(13,
154);
            this.l_SizedScreenshot_IsEdit.Name = "l_SizedScreenshot_IsEdit";
            this.l_SizedScreenshot_IsEdit.Size = new System.Drawing.Size(242, 26);
            this.l_SizedScreenshot_IsEdit.TabIndex = 20;
            this.l_SizedScreenshot_IsEdit.Text = "Run the editing tool for custom
screenshots";
            this.l_SizedScreenshot_IsEdit.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_TimeLapseScreenshot_Path
            //
            this.l_TimeLapseScreenshot_Path.AutoSize = true;
```

```
            this.l_TimeLapseScreenshot_Path.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.l_TimeLapseScreenshot_Path.Location = new System.Drawing.Point(13,
190);
            this.l_TimeLapseScreenshot_Path.Name = "l_TimeLapseScreenshot_Path";
            this.l_TimeLapseScreenshot_Path.Size = new System.Drawing.Size(242, 26);
            this.l_TimeLapseScreenshot_Path.TabIndex = 21;
            this.l_TimeLapseScreenshot_Path.Text = "Folder path for time lapse
screenshots";
            this.l_TimeLapseScreenshot_Path.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_TimeLapseScreenshot_Extension
            //
            this.l_TimeLapseScreenshot_Extension.AutoSize = true;
            this.l_TimeLapseScreenshot_Extension.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.l_TimeLapseScreenshot_Extension.Location = new
System.Drawing.Point(13, 226);
            this.l_TimeLapseScreenshot_Extension.Name =
"l_TimeLapseScreenshot_Extension";
            this.l_TimeLapseScreenshot_Extension.Size = new System.Drawing.Size(242,
26);
            this.l_TimeLapseScreenshot_Extension.TabIndex = 22;
            this.l_TimeLapseScreenshot_Extension.Text = "Time lapse screenshots
extension";
            this.l_TimeLapseScreenshot_Extension.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_TimeLapseScreenshot_Interval
            //
            this.l_TimeLapseScreenshot_Interval.AutoSize = true;
            this.l_TimeLapseScreenshot_Interval.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.l_TimeLapseScreenshot_Interval.Location = new
System.Drawing.Point(13, 262);
            this.l_TimeLapseScreenshot_Interval.Name =
"l_TimeLapseScreenshot_Interval";
            this.l_TimeLapseScreenshot_Interval.Size = new System.Drawing.Size(242,
26);
            this.l_TimeLapseScreenshot_Interval.TabIndex = 23;
            this.l_TimeLapseScreenshot_Interval.Text = "Save interval for time lapse
screenshots (sec)";
            this.l_TimeLapseScreenshot_Interval.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_VideoRecord_Path
            //
            this.l_VideoRecord_Path.AutoSize = true;
            this.l_VideoRecord_Path.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_VideoRecord_Path.Location = new System.Drawing.Point(13, 298);
            this.l_VideoRecord_Path.Name = "l_VideoRecord_Path";
            this.l_VideoRecord_Path.Size = new System.Drawing.Size(242, 26);
            this.l_VideoRecord_Path.TabIndex = 24;
            this.l_VideoRecord_Path.Text = "Folder path for screen records";
            this.l_VideoRecord_Path.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // tp_HotKeys
            //
            this.tp_HotKeys.Controls.Add(this.tlp_HotKeys);
            this.tp_HotKeys.Location = new System.Drawing.Point(4, 22);
```

```
            this.tp_HotKeys.Name = "tp_HotKeys";
            this.tp_HotKeys.Padding = new System.Windows.Forms.Padding(3);
            this.tp_HotKeys.Size = new System.Drawing.Size(700, 409);
            this.tp_HotKeys.TabIndex = 1;
            this.tp_HotKeys.Text = "Hotkeys";
            this.tp_HotKeys.UseVisualStyleBackColor = true;
            //
            // tlp_HotKeys
            //
            this.tlp_HotKeys.AutoScroll = true;
            this.tlp_HotKeys.ColumnCount = 9;
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 25F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 25F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 25F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 25F));
            this.tlp_HotKeys.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.Controls.Add(this.l_VSRHotKey_PhotoCam, 1, 1);
            this.tlp_HotKeys.Controls.Add(this.l_VSRHotKey_PhotoCamSized, 1, 3);
            this.tlp_HotKeys.Controls.Add(this.l_VSRHotKey_TimeLapse, 1, 5);
            this.tlp_HotKeys.Controls.Add(this.l_VSRHotKey_VideoCam, 1, 7);
            this.tlp_HotKeys.Controls.Add(this.l_VSRHotKey_Setting, 1, 9);
            this.tlp_HotKeys.Controls.Add(this.l_VSRHotKey_Info, 1, 11);
            this.tlp_HotKeys.Controls.Add(this.tb_VSRHotKey_PhotoCam, 5, 1);
            this.tlp_HotKeys.Controls.Add(this.tb_VSRHotKey_PhotoCamSized, 5, 3);
            this.tlp_HotKeys.Controls.Add(this.tb_VSRHotKey_TimeLapse, 5, 5);
            this.tlp_HotKeys.Controls.Add(this.tb_VSRHotKey_VideoCam, 5, 7);
            this.tlp_HotKeys.Controls.Add(this.tb_VSRHotKey_Setting, 5, 9);
            this.tlp_HotKeys.Controls.Add(this.tb_VSRHotKey_Info, 5, 11);
            this.tlp_HotKeys.Controls.Add(this.b_SaveHotKeys, 3, 13);
            this.tlp_HotKeys.Controls.Add(this.b_CancelHotKeys, 5, 13);
            this.tlp_HotKeys.Controls.Add(this.b_DefaultHotKeys, 7, 13);
            this.tlp_HotKeys.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tlp_HotKeys.Location = new System.Drawing.Point(3, 3);
            this.tlp_HotKeys.Name = "tlp_HotKeys";
            this.tlp_HotKeys.Padding = new System.Windows.Forms.Padding(0, 0, 20, 0);
            this.tlp_HotKeys.RowCount = 15;
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
```

```
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 26F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 36F));
            this.tlp_HotKeys.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_HotKeys.Size = new System.Drawing.Size(694, 403);
            this.tlp_HotKeys.TabIndex = 0;
            //
            // l_VSRHotKey_PhotoCam
            //
            this.l_VSRHotKey_PhotoCam.AutoSize = true;
            this.tlp_HotKeys.SetColumnSpan(this.l_VSRHotKey_PhotoCam, 3);
            this.l_VSRHotKey_PhotoCam.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_VSRHotKey_PhotoCam.Location = new System.Drawing.Point(13, 10);
            this.l_VSRHotKey_PhotoCam.Name = "l_VSRHotKey_PhotoCam";
            this.l_VSRHotKey_PhotoCam.Size = new System.Drawing.Size(316, 26);
            this.l_VSRHotKey_PhotoCam.TabIndex = 0;
            this.l_VSRHotKey_PhotoCam.Text = "Keyboard shortcuts for quick
screenshot";
            this.l_VSRHotKey_PhotoCam.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_VSRHotKey_PhotoCamSized
            //
            this.l_VSRHotKey_PhotoCamSized.AutoSize = true;
            this.tlp_HotKeys.SetColumnSpan(this.l_VSRHotKey_PhotoCamSized, 3);
            this.l_VSRHotKey_PhotoCamSized.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.l_VSRHotKey_PhotoCamSized.Location = new System.Drawing.Point(13,
46);
            this.l_VSRHotKey_PhotoCamSized.Name = "l_VSRHotKey_PhotoCamSized";
            this.l_VSRHotKey_PhotoCamSized.Size = new System.Drawing.Size(316, 26);
            this.l_VSRHotKey_PhotoCamSized.TabIndex = 1;
            this.l_VSRHotKey_PhotoCamSized.Text = "Keyboard shortcuts for custom
screenshot";
            this.l_VSRHotKey_PhotoCamSized.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_VSRHotKey_TimeLapse
            //
            this.l_VSRHotKey_TimeLapse.AutoSize = true;
            this.tlp_HotKeys.SetColumnSpan(this.l_VSRHotKey_TimeLapse, 3);
            this.l_VSRHotKey_TimeLapse.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_VSRHotKey_TimeLapse.Location = new System.Drawing.Point(13, 82);
            this.l_VSRHotKey_TimeLapse.Name = "l_VSRHotKey_TimeLapse";
            this.l_VSRHotKey_TimeLapse.Size = new System.Drawing.Size(316, 26);
            this.l_VSRHotKey_TimeLapse.TabIndex = 2;
            this.l_VSRHotKey_TimeLapse.Text = "Keyboard shortcuts for time lapse
screenshot";
```

```
            this.l_VSRHotKey_TimeLapse.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_VSRHotKey_VideoCam
            //
            this.l_VSRHotKey_VideoCam.AutoSize = true;
            this.tlp_HotKeys.SetColumnSpan(this.l_VSRHotKey_VideoCam, 3);
            this.l_VSRHotKey_VideoCam.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_VSRHotKey_VideoCam.Location = new System.Drawing.Point(13, 118);
            this.l_VSRHotKey_VideoCam.Name = "l_VSRHotKey_VideoCam";
            this.l_VSRHotKey_VideoCam.Size = new System.Drawing.Size(316, 26);
            this.l_VSRHotKey_VideoCam.TabIndex = 3;
            this.l_VSRHotKey_VideoCam.Text = "Keyboard shortcuts for start screen
record";
            this.l_VSRHotKey_VideoCam.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_VSRHotKey_Setting
            //
            this.l_VSRHotKey_Setting.AutoSize = true;
            this.tlp_HotKeys.SetColumnSpan(this.l_VSRHotKey_Setting, 3);
            this.l_VSRHotKey_Setting.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_VSRHotKey_Setting.Location = new System.Drawing.Point(13, 154);
            this.l_VSRHotKey_Setting.Name = "l_VSRHotKey_Setting";
            this.l_VSRHotKey_Setting.Size = new System.Drawing.Size(316, 26);
            this.l_VSRHotKey_Setting.TabIndex = 4;
            this.l_VSRHotKey_Setting.Text = "Keyboard shortcuts for settings";
            this.l_VSRHotKey_Setting.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // l_VSRHotKey_Info
            //
            this.l_VSRHotKey_Info.AutoSize = true;
            this.tlp_HotKeys.SetColumnSpan(this.l_VSRHotKey_Info, 3);
            this.l_VSRHotKey_Info.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_VSRHotKey_Info.Location = new System.Drawing.Point(13, 190);
            this.l_VSRHotKey_Info.Name = "l_VSRHotKey_Info";
            this.l_VSRHotKey_Info.Size = new System.Drawing.Size(316, 26);
            this.l_VSRHotKey_Info.TabIndex = 5;
            this.l_VSRHotKey_Info.Text = "Keyboard shortcuts for information";
            this.l_VSRHotKey_Info.TextAlign =
System.Drawing.ContentAlignment.MiddleRight;
            //
            // tb_VSRHotKey_PhotoCam
            //
            this.tb_VSRHotKey_PhotoCam.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_VSRHotKey_PhotoCam.Location = new System.Drawing.Point(345, 13);
            this.tb_VSRHotKey_PhotoCam.Name = "tb_VSRHotKey_PhotoCam";
            this.tb_VSRHotKey_PhotoCam.ReadOnly = true;
            this.tb_VSRHotKey_PhotoCam.Size = new System.Drawing.Size(150, 20);
            this.tb_VSRHotKey_PhotoCam.TabIndex = 6;
            this.tb_VSRHotKey_PhotoCam.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.tb_VSRHotKey_PhotoCam_KeyDown);
            //
            // tb_VSRHotKey_PhotoCamSized
            //
            this.tb_VSRHotKey_PhotoCamSized.Dock =
System.Windows.Forms.DockStyle.Fill;
            this.tb_VSRHotKey_PhotoCamSized.Location = new System.Drawing.Point(345,
49);
            this.tb_VSRHotKey_PhotoCamSized.Name = "tb_VSRHotKey_PhotoCamSized";
            this.tb_VSRHotKey_PhotoCamSized.ReadOnly = true;
```

```
            this.tb_VSRHotKey_PhotoCamSized.Size = new System.Drawing.Size(150, 20);
            this.tb_VSRHotKey_PhotoCamSized.TabIndex = 7;
            this.tb_VSRHotKey_PhotoCamSized.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.tb_VSRHotKey_PhotoCamSized_KeyDown);
            //
            // tb_VSRHotKey_TimeLapse
            //
            this.tb_VSRHotKey_TimeLapse.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_VSRHotKey_TimeLapse.Location = new System.Drawing.Point(345, 85);
            this.tb_VSRHotKey_TimeLapse.Name = "tb_VSRHotKey_TimeLapse";
            this.tb_VSRHotKey_TimeLapse.ReadOnly = true;
            this.tb_VSRHotKey_TimeLapse.Size = new System.Drawing.Size(150, 20);
            this.tb_VSRHotKey_TimeLapse.TabIndex = 8;
            this.tb_VSRHotKey_TimeLapse.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.tb_VSRHotKey_TimeLapse_KeyDown);
            //
            // tb_VSRHotKey_VideoCam
            //
            this.tb_VSRHotKey_VideoCam.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_VSRHotKey_VideoCam.Location = new System.Drawing.Point(345, 121);
            this.tb_VSRHotKey_VideoCam.Name = "tb_VSRHotKey_VideoCam";
            this.tb_VSRHotKey_VideoCam.ReadOnly = true;
            this.tb_VSRHotKey_VideoCam.Size = new System.Drawing.Size(150, 20);
            this.tb_VSRHotKey_VideoCam.TabIndex = 9;
            this.tb_VSRHotKey_VideoCam.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.tb_VSRHotKey_VideoCam_KeyDown);
            //
            // tb_VSRHotKey_Setting
            //
            this.tb_VSRHotKey_Setting.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_VSRHotKey_Setting.Location = new System.Drawing.Point(345, 157);
            this.tb_VSRHotKey_Setting.Name = "tb_VSRHotKey_Setting";
            this.tb_VSRHotKey_Setting.ReadOnly = true;
            this.tb_VSRHotKey_Setting.Size = new System.Drawing.Size(150, 20);
            this.tb_VSRHotKey_Setting.TabIndex = 10;
            this.tb_VSRHotKey_Setting.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.tb_VSRHotKey_Setting_KeyDown);
            //
            // tb_VSRHotKey_Info
            //
            this.tb_VSRHotKey_Info.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_VSRHotKey_Info.Location = new System.Drawing.Point(345, 193);
            this.tb_VSRHotKey_Info.Name = "tb_VSRHotKey_Info";
            this.tb_VSRHotKey_Info.ReadOnly = true;
            this.tb_VSRHotKey_Info.Size = new System.Drawing.Size(150, 20);
            this.tb_VSRHotKey_Info.TabIndex = 11;
            this.tb_VSRHotKey_Info.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.tb_VSRHotKey_Info_KeyDown);
            //
            // b_SaveHotKeys
            //
            this.b_SaveHotKeys.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_SaveHotKeys.Location = new System.Drawing.Point(179, 245);
            this.b_SaveHotKeys.Name = "b_SaveHotKeys";
            this.b_SaveHotKeys.Size = new System.Drawing.Size(150, 30);
            this.b_SaveHotKeys.TabIndex = 12;
            this.b_SaveHotKeys.Text = "Save";
            this.b_SaveHotKeys.UseVisualStyleBackColor = true;
            this.b_SaveHotKeys.Click += new
System.EventHandler(this.b_SaveHotKeys_Click);
            //
            // b_CancelHotKeys
```

```
            //
            this.b_CancelHotKeys.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_CancelHotKeys.Location = new System.Drawing.Point(345, 245);
            this.b_CancelHotKeys.Name = "b_CancelHotKeys";
            this.b_CancelHotKeys.Size = new System.Drawing.Size(150, 30);
            this.b_CancelHotKeys.TabIndex = 13;
            this.b_CancelHotKeys.Text = "Cancel";
            this.b_CancelHotKeys.UseVisualStyleBackColor = true;
            this.b_CancelHotKeys.Click += new
System.EventHandler(this.b_CancelHotKeys_Click);
            //
            // b_DefaultHotKeys
            //
            this.b_DefaultHotKeys.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_DefaultHotKeys.Location = new System.Drawing.Point(511, 245);
            this.b_DefaultHotKeys.Name = "b_DefaultHotKeys";
            this.b_DefaultHotKeys.Size = new System.Drawing.Size(150, 30);
            this.b_DefaultHotKeys.TabIndex = 14;
            this.b_DefaultHotKeys.Text = "Default";
            this.b_DefaultHotKeys.UseVisualStyleBackColor = true;
            this.b_DefaultHotKeys.Click += new
System.EventHandler(this.b_DefaultHotKeys_Click);
            //
            // WindowVSR_Settings
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.BackColor = System.Drawing.Color.White;
            this.ClientSize = new System.Drawing.Size(734, 461);
            this.Controls.Add(this.tlp_AllContentAlignment);
            this.MinimumSize = new System.Drawing.Size(500, 250);
            this.Name = "WindowVSR_Settings";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Variety Screen Recorder";
            this.tlp_AllContentAlignment.ResumeLayout(false);
            this.tc_SettingsMenu.ResumeLayout(false);
            this.tp_MainSetting.ResumeLayout(false);
            this.tlp_MainSettings.ResumeLayout(false);
            this.tlp_MainSettings.PerformLayout();
((System.ComponentModel.ISupportInitialize)(this.tb_TimeLapseScreenshot_Interval)).En
dInit();

((System.ComponentModel.ISupportInitialize)(this.nud_TimeLapseScreenshot_Interval)).E
ndInit();
            this.tp_HotKeys.ResumeLayout(false);
            this.tlp_HotKeys.ResumeLayout(false);
            this.tlp_HotKeys.PerformLayout();
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.TableLayoutPanel tlp_AllContentAlignment;
        private System.Windows.Forms.TabControl tc_SettingsMenu;
        private System.Windows.Forms.TabPage tp_MainSetting;
        private System.Windows.Forms.TabPage tp_HotKeys;
        private System.Windows.Forms.TableLayoutPanel tlp_MainSettings;
        private System.Windows.Forms.Button b_FullSizeScreenshot_Path;
        private System.Windows.Forms.TextBox tb_FullSizeScreenshot_Path;
        private System.Windows.Forms.ComboBox cb_FullSizeScreenshot_Extension;
        private System.Windows.Forms.TextBox tb_SizedScreenshot_Path;
```

```
        private System.Windows.Forms.ComboBox cb_SizedScreenshot_Extension;
        private System.Windows.Forms.CheckBox cb_SizedScreenshot_IsEdit;
        private System.Windows.Forms.TextBox tb_TimeLapseScreenshot_Path;
        private System.Windows.Forms.ComboBox cb_TimeLapseScreenshot_Extension;
        private System.Windows.Forms.TrackBar tb_TimeLapseScreenshot_Interval;
        private System.Windows.Forms.NumericUpDown nud_TimeLapseScreenshot_Interval;
        private System.Windows.Forms.TextBox tb_VideoRecord_Path;
        private System.Windows.Forms.Button b_SizedScreenshot_Path;
        private System.Windows.Forms.Button b_TimeLapseScreenshot_Path;
        private System.Windows.Forms.Button b_VideoRecord_Path;
        private System.Windows.Forms.TableLayoutPanel tlp_HotKeys;
        private System.Windows.Forms.Button b_CancelMainSetting;
        private System.Windows.Forms.Button b_SaveMainSettings;
        private System.Windows.Forms.Button b_DefaultMainSetting;
        private System.Windows.Forms.Label l_FullSizeScreenshot_Path;
        private System.Windows.Forms.Label l_FullSizeScreenshot_Extension;
        private System.Windows.Forms.Label l_SizedScreenshot_Path;
        private System.Windows.Forms.Label l_SizedScreenshot_Extension;
        private System.Windows.Forms.Label l_SizedScreenshot_IsEdit;
        private System.Windows.Forms.Label l_TimeLapseScreenshot_Path;
        private System.Windows.Forms.Label l_TimeLapseScreenshot_Extension;
        private System.Windows.Forms.Label l_TimeLapseScreenshot_Interval;
        private System.Windows.Forms.Label l_VideoRecord_Path;
        private System.Windows.Forms.Label l_VSRHotKey_PhotoCam;
        private System.Windows.Forms.Label l_VSRHotKey_PhotoCamSized;
        private System.Windows.Forms.Label l_VSRHotKey_TimeLapse;
        private System.Windows.Forms.Label l_VSRHotKey_VideoCam;
        private System.Windows.Forms.Label l_VSRHotKey_Setting;
        private System.Windows.Forms.Label l_VSRHotKey_Info;
        private System.Windows.Forms.TextBox tb_VSRHotKey_PhotoCam;
        private System.Windows.Forms.TextBox tb_VSRHotKey_PhotoCamSized;
        private System.Windows.Forms.TextBox tb_VSRHotKey_TimeLapse;
        private System.Windows.Forms.TextBox tb_VSRHotKey_VideoCam;
        private System.Windows.Forms.TextBox tb_VSRHotKey_Setting;
        private System.Windows.Forms.TextBox tb_VSRHotKey_Info;
        private System.Windows.Forms.Button b_SaveHotKeys;
        private System.Windows.Forms.Button b_CancelHotKeys;
        private System.Windows.Forms.Button b_DefaultHotKeys;
    }
}
```

WindowVSR_Information.cs

```
using System.Windows.Forms;

namespace VarietyScreenRecorder.WindowVSR
{
    public partial class WindowVSR_Information : Form
    {
        public WindowVSR_Information()
        {
            this.Icon = Properties.Resources.Logo;

            InitializeComponent();
        }
    }
}
```

WindowVSR_Information.Designer.cs

```
namespace VarietyScreenRecorder.WindowVSR
{
    partial class WindowVSR_Information
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.pb_Logo = new System.Windows.Forms.PictureBox();
            this.l_ProgramName = new System.Windows.Forms.Label();
            this.l_AboutProgram = new System.Windows.Forms.Label();
            ((System.ComponentModel.ISupportInitialize)(this.pb_Logo)).BeginInit();
            this.SuspendLayout();
            //
            // pb_Logo
            //
            this.pb_Logo.Image =
global::VarietyScreenRecorder.Properties.Resources.LogoPNG;
            this.pb_Logo.Location = new System.Drawing.Point(94, 30);
            this.pb_Logo.Name = "pb_Logo";
            this.pb_Logo.Size = new System.Drawing.Size(127, 127);
            this.pb_Logo.SizeMode = System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pb_Logo.TabIndex = 0;
            this.pb_Logo.TabStop = false;
            //
            // l_ProgramName
            //
            this.l_ProgramName.AutoSize = true;
            this.l_ProgramName.Font = new System.Drawing.Font("Consolas", 15.75F,
System.Drawing.FontStyle.Bold, System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.l_ProgramName.Location = new System.Drawing.Point(18, 160);
            this.l_ProgramName.Name = "l_ProgramName";
            this.l_ProgramName.Size = new System.Drawing.Size(286, 24);
            this.l_ProgramName.TabIndex = 1;
            this.l_ProgramName.Text = "Variety Screen Recorder";
```

```
            //
            // l_AboutProgram
            //
            this.l_AboutProgram.AutoSize = true;
            this.l_AboutProgram.Font = new System.Drawing.Font("Consolas", 14.25F,
System.Drawing.FontStyle.Regular, System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.l_AboutProgram.Location = new System.Drawing.Point(31, 214);
            this.l_AboutProgram.Name = "l_AboutProgram";
            this.l_AboutProgram.Size = new System.Drawing.Size(260, 176);
            this.l_AboutProgram.TabIndex = 2;
            this.l_AboutProgram.Text = "Made by\r\nValeriy Kozlov\r\n\r\nSpecially
for \r\nSerhiy Mykytovych Serdyuk\r\n\r\nZaporizhz" +
    "hia\r\n2020";
            this.l_AboutProgram.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // WindowVSR_Information
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.BackColor = System.Drawing.Color.White;
            this.ClientSize = new System.Drawing.Size(316, 423);
            this.Controls.Add(this.l_AboutProgram);
            this.Controls.Add(this.l_ProgramName);
            this.Controls.Add(this.pb_Logo);
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;
            this.MaximizeBox = false;
            this.Name = "WindowVSR_Information";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Variety Screen Recorder";
            ((System.ComponentModel.ISupportInitialize)(this.pb_Logo)).EndInit();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.PictureBox pb_Logo;
        private System.Windows.Forms.Label l_ProgramName;
        private System.Windows.Forms.Label l_AboutProgram;
    }
}
```

WindowGE_Main.cs

```
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Text;
using System.Windows.Forms;
using VarietyScreenRecorder.ExtraClass;

namespace VarietyScreenRecorder.Window_GE
{
    public partial class WindowGE_Main : Form
    {
```

```
        private static readonly Color ActiveElementColor = Color.FromArgb(64, 255,
64);
        private static readonly Color NotActiveElementColor = SystemColors.Control;

        private ShapeCursor CursorShape;
        private Color ActiveColor = Color.Black;
        private Color ActiveFillColor = Color.White;
        private int LineWidth = 10;
        private int TextSize = 14;
        private string ActiveText = "Default";

        private string RectangleType = "Rectangle";
        private string EllipseType = "Ellipse";
        private string LineType = "Line";

        private string ActiveTool = "None";
        private IShape ActiveElement = null;

        private Bitmap Translucent;
        private Bitmap NotTransparent;
        private Graphics TranslucentGraphics;
        private Graphics NotTransparentGraphics;

        private static readonly float OnePointLineHelper = 0.01f;

        private ShapesHistory Shapes;

        private string FilePath = "";
        private string FileName = "";
        private string FileExtension = "";
        private bool IsSave = false;

        public class ShapesHistory
        {
            public int VisibleShapesCount { get; set; }
            public List<IShape> History { get; set; }

            public ShapesHistory()
            {
                History = new List<IShape>();
                VisibleShapesCount = 0;
            }

            public void Add(IShape shape)
            {
                if (VisibleShapesCount != History.Count)
                {
                    History.RemoveRange(VisibleShapesCount, History.Count -
VisibleShapesCount);
                }

                History.Add(shape);
                VisibleShapesCount++;
            }

            public void Undo()
            {
                VisibleShapesCount = VisibleShapesCount <= 0 ? 0 : VisibleShapesCount
- 1;
            }

            public bool CanUndo()
```

```
            {
                return VisibleShapesCount > 0 ? true : false;
            }

            public void Redo()
            {
                VisibleShapesCount = VisibleShapesCount >= History.Count ?
History.Count : VisibleShapesCount + 1;
            }

            public bool CanRedo()
            {
                return VisibleShapesCount == History.Count ? false : true;
            }

            public void Draw(Graphics Translucent, Graphics NotTransparent)
            {
                for (int i = 0; i < VisibleShapesCount; ++i)
                    History[i].Draw(Translucent, NotTransparent);
            }
        }

        public WindowGE_Main(Image image)
        {
            this.Icon = Properties.Resources.Logo;

            InitializeComponent();
            pb_Image.Image = image;
            this.Text = DateTime.Now.ToString("yyyyMMdd_HHmmssfff") + "*";

            Translucent = new Bitmap(image.Width, image.Height);
            NotTransparent = new Bitmap(image.Width, image.Height);

            TranslucentGraphics = Graphics.FromImage(Translucent);
            NotTransparentGraphics = Graphics.FromImage(NotTransparent);

            TranslucentGraphics.SmoothingMode = SmoothingMode.None;
            TranslucentGraphics.CompositingMode = CompositingMode.SourceCopy;

            NotTransparentGraphics.SmoothingMode = SmoothingMode.HighQuality;
            NotTransparentGraphics.TextRenderingHint = TextRenderingHint.AntiAlias;

            CursorShape = new ShapeCursor(image.Width, image.Height);
            Shapes = new ShapesHistory();
        }

        private void SetButtonsNotActiveElementColor()
        {
            tsb_Pen.BackColor = NotActiveElementColor;
            tsb_Marker.BackColor = NotActiveElementColor;
            tsb_Text.BackColor = NotActiveElementColor;
            tsb_LineArrow.BackColor = NotActiveElementColor;
            tsb_Rectangle.BackColor = NotActiveElementColor;
            tsb_Ellipse.BackColor = NotActiveElementColor;
        }

        private void SetToolStripButtonStatus(ToolStripButton Button)
        {
            if (Button.BackColor == NotActiveElementColor)
                Button.BackColor = ActiveElementColor;
            else
                Button.BackColor = NotActiveElementColor;
```

```
}

private void SetToolStripSplitButtonStatus(ToolStripSplitButton Button)
{
    if (Button.BackColor == NotActiveElementColor)
        Button.BackColor = ActiveElementColor;
    else
        Button.BackColor = NotActiveElementColor;
}

private void tsb_Cursor_Click(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripButtonStatus((ToolStripButton)sender);
    if (tsb_Cursor.BackColor == ActiveElementColor)
    {
        ActiveTool = "Cursor";
        CursorShape.IsActive = true;
    }
    else
    {
        ActiveTool = "None";
        CursorShape.IsActive = false;
    }

    if (IsSave)
    {
        IsSave = false;
        this.Text += "*";
    }

    pb_Image.Refresh();
}

private void tsb_Pen_Click(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripButtonStatus((ToolStripButton)sender);
    ActiveTool = "Pen";
}

private void tsb_Marker_Click(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripButtonStatus((ToolStripButton)sender);
    ActiveTool = "Marker";
}

private void tsb_Text_Click(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripButtonStatus((ToolStripButton)sender);
    ActiveTool = "Text";

    ActiveText = GetText("Write your text", ActiveText);
}

private void tsb_Line_Click(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripButtonStatus((ToolStripButton)sender);
    ActiveTool = "Line";
```

```
}

private void tsb_LineArrow_ButtonClick(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripSplitButtonStatus((ToolStripSplitButton)sender);
    ActiveTool = LineType;
}

private void tsmi_Line_Click(object sender, EventArgs e)
{
    tsb_LineArrow.Image = Properties.Resources.GE_Line;
    LineType = "Line";
    tsb_LineArrow.PerformButtonClick();
}

private void tsmi_Arrow_Click(object sender, EventArgs e)
{
    tsb_LineArrow.Image = Properties.Resources.GE_Arrow;
    LineType = "Arrow";
    tsb_LineArrow.PerformButtonClick();
}

private void tsb_Rectangle_ButtonClick(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripSplitButtonStatus((ToolStripSplitButton)sender);
    ActiveTool = RectangleType;
}

private void tsmi_Rectangle_Click(object sender, EventArgs e)
{
    tsb_Rectangle.Image = Properties.Resources.GE_Rectangle;
    RectangleType = "Rectangle";
    tsb_Rectangle.PerformButtonClick();
}

private void tsmi_FillRectangle_Click(object sender, EventArgs e)
{
    tsb_Rectangle.Image = Properties.Resources.GE_FillRectangle;
    RectangleType = "FillRectangle";
    tsb_Rectangle.PerformButtonClick();
}

private void tsb_Ellipse_ButtonClick(object sender, EventArgs e)
{
    SetButtonsNotActiveElementColor();
    SetToolStripSplitButtonStatus((ToolStripSplitButton)sender);
    ActiveTool = EllipseType;
}

private void tsmi_Ellipse_Click(object sender, EventArgs e)
{
    tsb_Ellipse.Image = Properties.Resources.GE_Ellipse;
    EllipseType = "Ellipse";
    tsb_Ellipse.PerformButtonClick();
}

private void tsmi_FillEllipse_Click(object sender, EventArgs e)
{
    tsb_Ellipse.Image = Properties.Resources.GE_FillEllipse;
    EllipseType = "FillEllipse";
```

```
    tsb_Ellipse.PerformButtonClick();
}

private void tsb_Color_Click(object sender, EventArgs e)
{
    ColorDialog ColorPicker = new ColorDialog();
    ColorPicker.Color = ActiveColor;
    if (ColorPicker.ShowDialog() == DialogResult.OK)
        ActiveColor = ColorPicker.Color;
}

private void tsb_FillColor_Click(object sender, EventArgs e)
{
    ColorDialog ColorPicker = new ColorDialog();
    ColorPicker.Color = ActiveFillColor;
    if (ColorPicker.ShowDialog() == DialogResult.OK)
        ActiveFillColor = ColorPicker.Color;
}

private int GetSize(string Text, int StartValue)
{
    WindowGE_GetSize GetSizeWindow = new WindowGE_GetSize(Text, StartValue);
    GetSizeWindow.ShowDialog();

    if (GetSizeWindow.isSet)
        StartValue = (int)GetSizeWindow.nud_Size.Value;

    GetSizeWindow.Dispose();
    return StartValue;
}

private void tsb_LineWidth_Click(object sender, EventArgs e)
{
    LineWidth = GetSize("Select line width", LineWidth);
}

private void tsb_TextSize_Click(object sender, EventArgs e)
{
    TextSize = GetSize("Select text size", TextSize);
}

private string GetText(string Text, string ActiveText)
{
    WindowGE_GetText GetTextWindow = new WindowGE_GetText(Text, ActiveText);
    GetTextWindow.ShowDialog();

    if (GetTextWindow.isSet)
        ActiveText = GetTextWindow.tb_TextValue.Text;

    GetTextWindow.Dispose();
    return ActiveText;
}

public interface IShape
{
    void Draw(Graphics translucent, Graphics notTransparent);
    void Draw(Graphics graphics);
    void NewCursorPoint(Point currentCursorPosition);
    void UpdateModifierKey(bool shift);
}

public class ShapeCursor : IShape
```

```
        {
            public Point CursorPosition { get; set; }
            public Point MaxPosition { get; set; }
            public bool IsActive { get; set; } = false;

            public ShapeCursor(int imageWidth, int imageHeight)
            {
                CursorPosition = new Point(0, 0);
                MaxPosition = new Point(imageWidth - 1, imageHeight - 1);
            }

            public void Draw(Graphics translucent, Graphics notTransparent)
            {
                Draw(notTransparent);
            }

            public void Draw(Graphics graphics)
            {
                if (IsActive)
                {
                    graphics.DrawImage(Properties.Resources.GE_CursorImage, new
Rectangle(CursorPosition, Properties.Resources.GE_CursorImage.Size));
                }

            }

            public void NewCursorPoint(Point currentCursorPosition)
            {
                int X = currentCursorPosition.X, Y = currentCursorPosition.Y;

                if (X < 0) X = 0;
                else if (X > MaxPosition.X) X = MaxPosition.X;

                if (Y < 0) Y = 0;
                else if (Y > MaxPosition.Y) Y = MaxPosition.Y;

                CursorPosition = new Point(X, Y);
            }

            public void UpdateModifierKey(bool shift)
            {
                ;
            }
        }

        public class ShapeText : IShape
        {
            public string Text { get; set; }
            public Font TextFont { get; set; }
            public SolidBrush TextSolidBrush { get; set; }
            public Point TextPosition { get; set; }

            public ShapeText(string text, int textSize, Point position, Color color)
            {
                Text = text;
                TextFont = new Font("Microsoft Sans Serif", textSize,
FontStyle.Regular);
                TextPosition = position;
                TextSolidBrush = new SolidBrush(color);
            }

            public void Draw(Graphics translucent, Graphics notTransparent)
```

```
            {
                Draw(notTransparent);
            }

            public void Draw(Graphics graphics)
            {
                graphics.DrawString(Text, TextFont, TextSolidBrush, TextPosition);
            }

            public void NewCursorPoint(Point currentCursorPosition)
            {
                TextPosition = currentCursorPosition;
            }

            public void UpdateModifierKey(bool shift)
            {
                ;
            }
        }

        public static void SetUpPen(Pen pen)
        {
            pen.StartCap = LineCap.Round;
            pen.EndCap = LineCap.Round;
            pen.LineJoin = LineJoin.Round;
            pen.MiterLimit = 0;
        }

        public static void SetUpPenForArrow(Pen pen)
        {
            pen.StartCap = LineCap.Round;
            pen.EndCap = LineCap.ArrowAnchor;
            pen.LineJoin = LineJoin.Round;
            pen.MiterLimit = 0;
        }

        public class ShapeMarker : IShape
        {
            public Pen MarkerPen { get; set; }
            public List<Point> MarkerPoints { get; set; }

            public ShapeMarker(int lineWidth, Point position)
            {
                MarkerPoints = new List<Point>();
                MarkerPoints.Add(position);
                MarkerPen = new Pen(Color.FromArgb(127, 64, 255, 64), lineWidth);
                SetUpPen(MarkerPen);
            }

            public void Draw(Graphics translucent, Graphics notTransparent)
            {
                Draw(translucent);
            }

            public void Draw(Graphics graphics)
            {
                if (MarkerPoints.Count == 1)
                    graphics.DrawLine(MarkerPen, MarkerPoints[0].X -
OnePointLineHelper, MarkerPoints[0].Y - OnePointLineHelper, MarkerPoints[0].X,
MarkerPoints[0].Y);
                else
                    graphics.DrawLines(MarkerPen, MarkerPoints.ToArray());
```

```
            }

            public void NewCursorPoint(Point currentCursorPosition)
            {
                MarkerPoints.Add(currentCursorPosition);
            }
            public void UpdateModifierKey(bool shift)
            {
                ;
            }
        }

        public class ShapePen : IShape
        {
            public Pen PenPen { get; set; }
            public List<Point> PenPoints { get; set; }

            public ShapePen(int lineWidth, Color color, Point position)
            {
                PenPoints = new List<Point>();
                PenPoints.Add(position);
                PenPen = new Pen(color, lineWidth);
                SetUpPen(PenPen);
            }

            public void Draw(Graphics translucent, Graphics notTransparent)
            {
                Draw(notTransparent);
            }

            public void Draw(Graphics graphics)
            {
                if (PenPoints.Count == 1)
                    graphics.DrawLine(PenPen, PenPoints[0].X - OnePointLineHelper,
PenPoints[0].Y - OnePointLineHelper, PenPoints[0].X, PenPoints[0].Y);
                else
                    graphics.DrawLines(PenPen, PenPoints.ToArray());
            }

            public void NewCursorPoint(Point currentCursorPosition)
            {
                PenPoints.Add(currentCursorPosition);
            }

            public void UpdateModifierKey(bool shift)
            {
                ;
            }
        }

        public class ShapeLine : IShape
        {
            public Pen PenLine { get; set; }
            public Point StartPoint { get; set; }
            public Point EndPoint { get; set; }
            public Point SaveLineEndPoint { get; set; }
            private bool IsValidEndPoint = false;
            private bool IsShift = false;

            public ShapeLine(int lineWidth, Color color, Point position, bool
isArrow)
            {
```

```
            StartPoint = position;

            PenLine = new Pen(color, lineWidth);

            if (isArrow)
                SetUpPenForArrow(PenLine);
            else
                SetUpPen(PenLine);
        }

        public void Draw(Graphics translucent, Graphics notTransparent)
        {
            Draw(notTransparent);
        }

        public void Draw(Graphics graphics)
        {
            if (IsValidEndPoint)
                graphics.DrawLine(PenLine, StartPoint, EndPoint);
            else
                graphics.DrawLine(PenLine, StartPoint.X - OnePointLineHelper,
StartPoint.Y - OnePointLineHelper, StartPoint.X, StartPoint.Y);
        }

        public void NewCursorPoint(Point currentCursorPosition)
        {
            if (currentCursorPosition.X == StartPoint.X &&
currentCursorPosition.Y == StartPoint.Y)
                IsValidEndPoint = false;
            else
                IsValidEndPoint = true;

            EndPoint = currentCursorPosition;
            if (IsShift)
                MakeCorrectLine();
        }

        public void UpdateModifierKey(bool shift)
        {
            if (IsShift != shift)
            {
                IsShift = shift;

                if (IsShift)
                    MakeCorrectLine();
                else
                    UnMakeCorrectLine();
            }
        }

        private void MakeCorrectLine()
        {
            int X1 = StartPoint.X;
            int Y1 = StartPoint.Y;
            int X2 = EndPoint.X;
            int Y2 = EndPoint.Y;

            int SizeX = Math.Abs(X1 - EndPoint.X);
            int SizeY = Math.Abs(Y1 - EndPoint.Y);

            if (SizeX > SizeY)
                Y2 = Y1;
```

```
                else
                    X2 = X1;

                SaveLineEndPoint = EndPoint;
                EndPoint = new Point(X2, Y2);
            }

            private void UnMakeCorrectLine()
            {
                EndPoint = SaveLineEndPoint;
            }
        }

        public class ShapeRectangle : IShape
        {
            public Pen PenRectangle { get; set; }
            public SolidBrush BrushFillRectangle { get; set; }
            public Point StartPoint { get; set; }
            public Point EndPoint { get; set; }
            public Point SaveRectangleEndPoint { get; set; }
            private bool IsValidEndPoint = false;
            private bool IsShift = false;

            public ShapeRectangle(int lineWidth, Color activeColor, Point position)
            {
                StartPoint = position;

                PenRectangle = new Pen(activeColor, lineWidth);
                SetUpPen(PenRectangle);

                BrushFillRectangle = new SolidBrush(Color.Transparent);

                StartPoint = position;
            }

            public ShapeRectangle(int lineWidth, Color activeColor, Color fillColor,
Point position)
            {
                StartPoint = position;
                PenRectangle = new Pen(activeColor, lineWidth);
                SetUpPen(PenRectangle);

                BrushFillRectangle = new SolidBrush(fillColor);

                StartPoint = position;
            }

            public void Draw(Graphics translucent, Graphics notTransparent)
            {
                Draw(notTransparent);
            }

            public void Draw(Graphics graphics)
            {
                if (IsValidEndPoint)
                {
                    if(StartPoint.X == EndPoint.X || StartPoint.Y == EndPoint.Y)
                    {
                        graphics.DrawLine(PenRectangle, StartPoint, EndPoint);
                    }
                    else
                    {
```

```
                        int X1 = Math.Min(StartPoint.X, EndPoint.X);
                        int Y1 = Math.Min(StartPoint.Y, EndPoint.Y);
                        int X2 = Math.Max(StartPoint.X, EndPoint.X);
                        int Y2 = Math.Max(StartPoint.Y, EndPoint.Y);

                        int width = X2 - X1;
                        int height = Y2 - Y1;

                        if (width < PenRectangle.Width && height <
PenRectangle.Width)
                        {
                            graphics.DrawLine(PenRectangle, X1, Y1, X2, Y1);
                            graphics.DrawLine(PenRectangle, X2, Y1, X2, Y2);
                            graphics.DrawLine(PenRectangle, X2, Y2, X1, Y2);
                            graphics.DrawLine(PenRectangle, X1, Y2, X1, Y1);
                        }
                        else
                        {
                            graphics.FillRectangle(BrushFillRectangle, X1, Y1, X2 -
X1, Y2 - Y1);
                            graphics.DrawRectangle(PenRectangle, X1, Y1, X2 - X1, Y2
- Y1);
                        }
                    }
                }
                else
                {
                    graphics.DrawLine(PenRectangle, StartPoint.X -
OnePointLineHelper, StartPoint.Y - OnePointLineHelper, StartPoint.X, StartPoint.Y);
                }
            }

            public void NewCursorPoint(Point currentCursorPosition)
            {
                if (currentCursorPosition.X == StartPoint.X &&
currentCursorPosition.Y == StartPoint.Y)
                    IsValidEndPoint = false;
                else
                    IsValidEndPoint = true;
                EndPoint = currentCursorPosition;
                if (IsShift)
                    MakeSquare();
            }

            public void UpdateModifierKey(bool shift)
            {
                if (IsShift != shift)
                {
                    IsShift = shift;

                    if (IsShift)
                        MakeSquare();
                    else
                        UnMakeSquare();
                }
            }

            private void MakeSquare()
            {
                int X1 = StartPoint.X;
                int Y1 = StartPoint.Y;
                int X2 = 0;
```

```
                int Y2 = 0;

                int SizeX = Math.Abs(X1 - EndPoint.X);
                int SizeY = Math.Abs(Y1 - EndPoint.Y);
                int EqualSize = Math.Min(SizeX, SizeY);

                if (StartPoint.X > EndPoint.X)
                    X2 = X1 - EqualSize;
                else
                    X2 = X1 + EqualSize;

                if (StartPoint.Y > EndPoint.Y)
                    Y2 = Y1 - EqualSize;
                else
                    Y2 = Y1 + EqualSize;

                SaveRectangleEndPoint = EndPoint;
                EndPoint = new Point(X2, Y2);
            }

            private void UnMakeSquare()
            {
                EndPoint = SaveRectangleEndPoint;
            }
        }

        public class ShapeEllipse : IShape
        {
            public Pen PenEllipse { get; set; }
            public SolidBrush BrushFillEllipse { get; set; }
            public Point StartPoint { get; set; }
            public Point EndPoint { get; set; }
            private Point SaveEllipseEndPoint { get; set; }
            private bool IsValidEndPoint = false;
            private bool IsShift = false;

            public ShapeEllipse(int lineWidth, Color activeColor, Point position)
            {
                StartPoint = position;
                PenEllipse = new Pen(activeColor, lineWidth);
                SetUpPen(PenEllipse);

                BrushFillEllipse = new SolidBrush(Color.Transparent);

                StartPoint = position;
            }

            public ShapeEllipse(int lineWidth, Color activeColor, Color fillColor,
Point position)
            {
                StartPoint = position;

                PenEllipse = new Pen(activeColor, lineWidth);
                SetUpPen(PenEllipse);

                BrushFillEllipse = new SolidBrush(fillColor);

                StartPoint = position;
            }

            public void Draw(Graphics translucent, Graphics notTransparent)
            {
```

```
            Draw(notTransparent);
        }

        public void Draw(Graphics graphics)
        {
            if (IsValidEndPoint)
            {
                if (StartPoint.X == EndPoint.X || StartPoint.Y == EndPoint.Y)
                {
                    graphics.DrawLine(PenEllipse, StartPoint, EndPoint);
                }
                else
                {
                    int X1 = Math.Min(StartPoint.X, EndPoint.X);
                    int Y1 = Math.Min(StartPoint.Y, EndPoint.Y);
                    int X2 = Math.Max(StartPoint.X, EndPoint.X);
                    int Y2 = Math.Max(StartPoint.Y, EndPoint.Y);

                    int width = X2 - X1;
                    int height = Y2 - Y1;

                    if (width < PenEllipse.Width && height < PenEllipse.Width)
                    {
                        graphics.FillEllipse(PenEllipse.Brush, X1 -
PenEllipse.Width / 2, Y1 - PenEllipse.Width / 2, X2 - X1 + PenEllipse.Width, Y2 - Y1
+ PenEllipse.Width);
                    }
                    else
                    {
                        graphics.FillEllipse(BrushFillEllipse, X1, Y1, X2 - X1,
Y2 - Y1);
                        graphics.DrawEllipse(PenEllipse, X1, Y1, X2 - X1, Y2 -
Y1);
                    }
                }
            }
            else
            {
                graphics.DrawLine(PenEllipse, StartPoint.X - OnePointLineHelper,
StartPoint.Y - OnePointLineHelper, StartPoint.X, StartPoint.Y);
            }
        }

        public void NewCursorPoint(Point currentCursorPosition)
        {
            if (currentCursorPosition.X == StartPoint.X &&
currentCursorPosition.Y == StartPoint.Y)
                IsValidEndPoint = false;
            else
                IsValidEndPoint = true;

            EndPoint = currentCursorPosition;
            if (IsShift)
                MakeCircle();
        }

        public void UpdateModifierKey(bool shift)
        {
            if (IsShift != shift)
            {
                IsShift = shift;
```

```
                    if (IsShift)
                        MakeCircle();
                    else
                        UnMakeCircle();
                }
            }

            private void MakeCircle()
            {
                int X1 = StartPoint.X;
                int Y1 = StartPoint.Y;
                int X2 = 0;
                int Y2 = 0;

                int SizeX = Math.Abs(X1 - EndPoint.X);
                int SizeY = Math.Abs(Y1 - EndPoint.Y);
                int EqualSize = Math.Min(SizeX, SizeY);

                if (StartPoint.X > EndPoint.X)
                    X2 = X1 - EqualSize;
                else
                    X2 = X1 + EqualSize;

                if (StartPoint.Y > EndPoint.Y)
                    Y2 = Y1 - EqualSize;
                else
                    Y2 = Y1 + EqualSize;

                SaveEllipseEndPoint = EndPoint;
                EndPoint = new Point(X2, Y2);
            }

            private void UnMakeCircle()
            {
                EndPoint = SaveEllipseEndPoint;
            }
        }

        private void pb_Image_MouseDown(object sender, MouseEventArgs e)
        {
            if (e.Button == MouseButtons.Left)
            {
                if (ActiveTool == "Cursor")
                {
                    ActiveElement = CursorShape;
                    CursorShape.NewCursorPoint(e.Location);
                }
                if (ActiveTool == "Text")
                {
                    ActiveElement = new ShapeText(ActiveText, TextSize, e.Location,
ActiveColor);
                }
                else if (ActiveTool == "Pen")
                {
                    ActiveElement = new ShapePen(LineWidth, ActiveColor, e.Location);
                }
                else if (ActiveTool == "Marker")
                {
                    ActiveElement = new ShapeMarker(LineWidth, e.Location);
                }
                else if (ActiveTool == "Line")
                {
```

```
                    ActiveElement = new ShapeLine(LineWidth, ActiveColor, e.Location,
false);
                }
                else if (ActiveTool == "Arrow")
                {
                    ActiveElement = new ShapeLine(LineWidth, ActiveColor, e.Location,
true);
                }
                else if (ActiveTool == "Rectangle")
                {
                    ActiveElement = new ShapeRectangle(LineWidth, ActiveColor,
e.Location);
                }
                else if (ActiveTool == "FillRectangle")
                {
                    ActiveElement = new ShapeRectangle(LineWidth, ActiveColor,
ActiveFillColor, e.Location);
                }
                else if (ActiveTool == "Ellipse")
                {
                    ActiveElement = new ShapeEllipse(LineWidth, ActiveColor,
e.Location);
                }
                else if (ActiveTool == "FillEllipse")
                {
                    ActiveElement = new ShapeEllipse(LineWidth, ActiveColor,
ActiveFillColor, e.Location);
                }

                if (ActiveTool != "None" && IsSave)
                {
                    IsSave = false;
                    this.Text += "*";
                }

                pb_Image.Refresh();
            }
        }
        private void pb_Image_MouseMove(object sender, MouseEventArgs e)
        {
            if (e.Button == MouseButtons.Left)
            {
                if (ActiveElement != null)
                {
                    ActiveElement.NewCursorPoint(e.Location);
                    pb_Image.Refresh();
                }
            }
        }

        private void pb_Image_MouseUp(object sender, MouseEventArgs e)
        {
            if (e.Button == MouseButtons.Left)
            {
                if (ActiveElement != null)
                {
                    if (ActiveTool != "Cursor")
                    {
                        ActiveElement.Draw(TranslucentGraphics,
NotTransparentGraphics);
                        Shapes.Add(ActiveElement);
                    }
```

```
                    ActiveElement = null;
                    UpdateUndoRedoButtons();

                    pb_Image.Refresh();
                }
            }
        }

        private void pb_Image_Paint(object sender, PaintEventArgs e)
        {
            if (ActiveElement != null && ActiveTool != "Cursor")
            {
                Image CopyTranslucent = (Image)Translucent.Clone();
                Image CopyNotTransparent = (Image)NotTransparent.Clone();

                Graphics CopyTranslucentGraphics =
Graphics.FromImage(CopyTranslucent);
                Graphics CopyNotTransparentGraphics =
Graphics.FromImage(CopyNotTransparent);

                CopyTranslucentGraphics.SmoothingMode = SmoothingMode.None;
                CopyTranslucentGraphics.CompositingMode = CompositingMode.SourceCopy;
                CopyNotTransparentGraphics.SmoothingMode = SmoothingMode.HighQuality;
                CopyNotTransparentGraphics.TextRenderingHint =
TextRenderingHint.AntiAlias;

                ActiveElement.Draw(CopyTranslucentGraphics,
CopyNotTransparentGraphics);

                CursorShape.Draw(e.Graphics);
                e.Graphics.DrawImage(CopyTranslucent, new Point(0, 0));
                e.Graphics.DrawImage(CopyNotTransparent, new Point(0, 0));

                CopyTranslucent.Dispose();
                CopyNotTransparent.Dispose();
                CopyTranslucentGraphics.Dispose();
                CopyNotTransparentGraphics.Dispose();
            }
            else
            {
                CursorShape.Draw(e.Graphics);
                e.Graphics.DrawImage(Translucent, new Point(0, 0));
                e.Graphics.DrawImage(NotTransparent, new Point(0, 0));
            }
        }

        private void ReDrawImages()
        {
            TranslucentGraphics.Clear(Color.Transparent);
            NotTransparentGraphics.Clear(Color.Transparent);

            Shapes.Draw(TranslucentGraphics, NotTransparentGraphics);
        }

        private void UpdateUndoRedoButtons()
        {
            if (Shapes.CanUndo())
                tsmi_Undo.Enabled = true;
            else
                tsmi_Undo.Enabled = false;
            if (Shapes.CanRedo())
                tsmi_Redo.Enabled = true;
```

```
    else
        tsmi_Redo.Enabled = false;
}

private void tsmi_Undo_Click(object sender, EventArgs e)
{
    if (IsSave)
    {
        IsSave = false;
        this.Text += "*";
    }

    if (ActiveElement != null)
    {
        ActiveElement = null;
    }
    else
    {
        Shapes.Undo();

        ReDrawImages();
    }

    UpdateUndoRedoButtons();
    pb_Image.Refresh();
}

private void tsmi_Redo_Click(object sender, EventArgs e)
{
    if (IsSave)
    {
        IsSave = false;
        this.Text += "*";
    }

    if (ActiveElement != null)
    {
        if (ActiveTool != "Cursor")
        {
            ActiveElement.Draw(TranslucentGraphics, NotTransparentGraphics);
            Shapes.Add(ActiveElement);
        }

        ActiveElement = null;
    }
    else
    {
        Shapes.Redo();

        ReDrawImages();
    }

    UpdateUndoRedoButtons();
    pb_Image.Refresh();
}

private void SaveImage()
{
    if (ActiveElement != null)
    {
        if (ActiveTool != "Cursor")
        {
```

```
                    ActiveElement.Draw(TranslucentGraphics, NotTransparentGraphics);
                    Shapes.Add(ActiveElement);

                    UpdateUndoRedoButtons();

                    pb_Image.Refresh();
                }
            }

            Image Result = (Image)pb_Image.Image.Clone();
            Graphics ResultGraphics = Graphics.FromImage(Result);
            CursorShape.Draw(ResultGraphics);
            ResultGraphics.DrawImage(Translucent, new Point(0, 0));
            ResultGraphics.DrawImage(NotTransparent, new Point(0, 0));

            Clipboard.SetImage(Result);
            ScreenshotManager.SaveScreenshot(Result, FilePath, FileName,
FileExtension);
            this.Text = FilePath + "\\" + FileName + "." + FileExtension;
            IsSave = true;

            Result.Dispose();
            ResultGraphics.Dispose();
        }

        private bool GetFilePath()
        {
            SaveFileDialog saveFileDialog = new SaveFileDialog();
            saveFileDialog.Title = "Save screenshot as...";
            saveFileDialog.Filter = "Image files(*.png)|*.png|Image
files(*.jpeg)|*.jpeg|Image files(*.bmp)|*.bmp|Image files(*.gif)|*.gif";
            saveFileDialog.OverwritePrompt = true;
            saveFileDialog.CheckPathExists = true;
            if (FilePath != "")
            {
                saveFileDialog.FileName = FilePath + "\\" + FileName + "." +
FileExtension;
                saveFileDialog.FileName = FileName;
            }


            if (saveFileDialog.ShowDialog() == DialogResult.OK)
            {
                FileExtension =
saveFileDialog.FileName.Substring(saveFileDialog.FileName.LastIndexOf('.') + 1);
                FileName =
saveFileDialog.FileName.Substring(saveFileDialog.FileName.LastIndexOf('\\') + 1,
saveFileDialog.FileName.LastIndexOf('.') - saveFileDialog.FileName.LastIndexOf('\\')
- 1);
                FilePath = saveFileDialog.FileName.Substring(0,
saveFileDialog.FileName.LastIndexOf('\\'));

                return true;
            }
            else
                return false;
        }

        private void tsmi_Save_Click(object sender, EventArgs e)
        {
            if (FilePath != "")
                SaveImage();
```

```
            else
                tsmi_SaveAs.PerformClick();
        }

        private void tsmi_SaveAs_Click(object sender, EventArgs e)
        {
            if (GetFilePath())
                SaveImage();
        }

        private void WindowGE_Main_KeyDown(object sender, KeyEventArgs e)
        {
            if(e.Shift)
                if (ActiveElement != null)
                    ActiveElement.UpdateModifierKey(true);

            pb_Image.Refresh();
        }

        private void WindowGE_Main_KeyUp(object sender, KeyEventArgs e)
        {
            if (!e.Shift)
                if (ActiveElement != null)
                    ActiveElement.UpdateModifierKey(false);

            pb_Image.Refresh();
        }

        private void WindowGE_Main_FormClosing(object sender, FormClosingEventArgs e)
        {
            if(!IsSave)
            {
                DialogResult dialogResult = MessageBox.Show("Want to save a
screenshot before closing?", "Closing...", MessageBoxButtons.YesNo);
                if (dialogResult == DialogResult.Yes)
                {
                    tsmi_Save.PerformClick();
                }
            }
        }
    }
}
```

WindowGE_Main.Designer.cs

```
namespace VarietyScreenRecorder.Window_GE
{
    partial class WindowGE_Main
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
```

```
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(WindowGE_Main));
            this.ms_Menu = new System.Windows.Forms.MenuStrip();
            this.tsmi_File = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_Save = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_SaveAs = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_Edit = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_Undo = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_Redo = new System.Windows.Forms.ToolStripMenuItem();
            this.ts_ToolsButtons = new System.Windows.Forms.ToolStrip();
            this.tsb_Cursor = new System.Windows.Forms.ToolStripButton();
            this.tsb_Pen = new System.Windows.Forms.ToolStripButton();
            this.tsb_Marker = new System.Windows.Forms.ToolStripButton();
            this.tsb_Text = new System.Windows.Forms.ToolStripButton();
            this.tsb_LineArrow = new System.Windows.Forms.ToolStripSplitButton();
            this.tsmi_Line = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_Arrow = new System.Windows.Forms.ToolStripMenuItem();
            this.tsb_Rectangle = new System.Windows.Forms.ToolStripSplitButton();
            this.tsmi_Rectangle = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_FillRectangle = new System.Windows.Forms.ToolStripMenuItem();
            this.tsb_Ellipse = new System.Windows.Forms.ToolStripSplitButton();
            this.tsmi_Ellipse = new System.Windows.Forms.ToolStripMenuItem();
            this.tsmi_FillEllipse = new System.Windows.Forms.ToolStripMenuItem();
            this.toolStripSeparator1 = new System.Windows.Forms.ToolStripSeparator();
            this.tsb_Color = new System.Windows.Forms.ToolStripButton();
            this.tsb_FillColor = new System.Windows.Forms.ToolStripButton();
            this.tsb_LineWidth = new System.Windows.Forms.ToolStripButton();
            this.tsb_TextSize = new System.Windows.Forms.ToolStripButton();
            this.tlp_MainArea = new System.Windows.Forms.TableLayoutPanel();
            this.p_PictureArea = new System.Windows.Forms.Panel();
            this.pb_Image = new System.Windows.Forms.PictureBox();
            this.ms_Menu.SuspendLayout();
            this.ts_ToolsButtons.SuspendLayout();
            this.tlp_MainArea.SuspendLayout();
            this.p_PictureArea.SuspendLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pb_Image)).BeginInit();
            this.SuspendLayout();
            //
            // ms_Menu
            //
            this.ms_Menu.BackColor = System.Drawing.SystemColors.Control;
            this.ms_Menu.Items.AddRange(new System.Windows.Forms.ToolStripItem[] {
            this.tsmi_File,
            this.tsmi_Edit});
            this.ms_Menu.Location = new System.Drawing.Point(0, 0);
```

```
            this.ms_Menu.Name = "ms_Menu";
            this.ms_Menu.RenderMode =
System.Windows.Forms.ToolStripRenderMode.System;
            this.ms_Menu.Size = new System.Drawing.Size(734, 24);
            this.ms_Menu.TabIndex = 0;
            this.ms_Menu.Text = "menuStrip1";
            //
            // tsmi_File
            //
            this.tsmi_File.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.tsmi_Save,
            this.tsmi_SaveAs});
            this.tsmi_File.Name = "tsmi_File";
            this.tsmi_File.Size = new System.Drawing.Size(37, 20);
            this.tsmi_File.Text = "File";
            //
            // tsmi_Save
            //
            this.tsmi_Save.Name = "tsmi_Save";
            this.tsmi_Save.ShortcutKeys =
((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control |
System.Windows.Forms.Keys.S)));
            this.tsmi_Save.Size = new System.Drawing.Size(193, 22);
            this.tsmi_Save.Text = "Save";
            this.tsmi_Save.Click += new System.EventHandler(this.tsmi_Save_Click);
            //
            // tsmi_SaveAs
            //
            this.tsmi_SaveAs.Name = "tsmi_SaveAs";
            this.tsmi_SaveAs.ShortcutKeys =
((System.Windows.Forms.Keys)(((System.Windows.Forms.Keys.Control |
System.Windows.Forms.Keys.Shift)
            | System.Windows.Forms.Keys.S)));
            this.tsmi_SaveAs.Size = new System.Drawing.Size(193, 22);
            this.tsmi_SaveAs.Text = "Save as...";
            this.tsmi_SaveAs.Click += new
System.EventHandler(this.tsmi_SaveAs_Click);
            //
            // tsmi_Edit
            //
            this.tsmi_Edit.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.tsmi_Undo,
            this.tsmi_Redo});
            this.tsmi_Edit.Name = "tsmi_Edit";
            this.tsmi_Edit.Size = new System.Drawing.Size(39, 20);
            this.tsmi_Edit.Text = "Edit";
            //
            // tsmi_Undo
            //
            this.tsmi_Undo.Enabled = false;
            this.tsmi_Undo.Name = "tsmi_Undo";
            this.tsmi_Undo.ShortcutKeys =
((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control |
System.Windows.Forms.Keys.Z)));
            this.tsmi_Undo.Size = new System.Drawing.Size(144, 22);
            this.tsmi_Undo.Text = "Undo";
            this.tsmi_Undo.Click += new System.EventHandler(this.tsmi_Undo_Click);
            //
            // tsmi_Redo
            //
```

```
            this.tsmi_Redo.Enabled = false;
            this.tsmi_Redo.Name = "tsmi_Redo";
            this.tsmi_Redo.ShortcutKeys =
((System.Windows.Forms.Keys)((System.Windows.Forms.Keys.Control |
System.Windows.Forms.Keys.Y)));
            this.tsmi_Redo.Size = new System.Drawing.Size(144, 22);
            this.tsmi_Redo.Text = "Redo";
            this.tsmi_Redo.Click += new System.EventHandler(this.tsmi_Redo_Click);
            //
            // ts_ToolsButtons
            //
            this.ts_ToolsButtons.BackColor = System.Drawing.SystemColors.Control;
            this.ts_ToolsButtons.Dock = System.Windows.Forms.DockStyle.Left;
            this.ts_ToolsButtons.GripStyle =
System.Windows.Forms.ToolStripGripStyle.Hidden;
            this.ts_ToolsButtons.Items.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.tsb_Cursor,
            this.tsb_Pen,
            this.tsb_Marker,
            this.tsb_Text,
            this.tsb_LineArrow,
            this.tsb_Rectangle,
            this.tsb_Ellipse,
            this.toolStripSeparator1,
            this.tsb_Color,
            this.tsb_FillColor,
            this.tsb_LineWidth,
            this.tsb_TextSize});
            this.ts_ToolsButtons.LayoutStyle =
System.Windows.Forms.ToolStripLayoutStyle.VerticalStackWithOverflow;
            this.ts_ToolsButtons.Location = new System.Drawing.Point(0, 24);
            this.ts_ToolsButtons.Name = "ts_ToolsButtons";
            this.ts_ToolsButtons.RenderMode =
System.Windows.Forms.ToolStripRenderMode.System;
            this.ts_ToolsButtons.Size = new System.Drawing.Size(44, 387);
            this.ts_ToolsButtons.TabIndex = 1;
            this.ts_ToolsButtons.Text = "toolStrip1";
            //
            // tsb_Cursor
            //
            this.tsb_Cursor.AutoSize = false;
            this.tsb_Cursor.BackColor = System.Drawing.SystemColors.Control;
            this.tsb_Cursor.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_Cursor.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Cursor;
            this.tsb_Cursor.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_Cursor.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_Cursor.Name = "tsb_Cursor";
            this.tsb_Cursor.Size = new System.Drawing.Size(28, 28);
            this.tsb_Cursor.Text = "Cursor";
            this.tsb_Cursor.Click += new System.EventHandler(this.tsb_Cursor_Click);
            //
            // tsb_Pen
            //
            this.tsb_Pen.AutoSize = false;
            this.tsb_Pen.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_Pen.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Pen;
```

```
            this.tsb_Pen.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_Pen.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_Pen.Name = "tsb_Pen";
            this.tsb_Pen.Size = new System.Drawing.Size(28, 28);
            this.tsb_Pen.Text = "Pen";
            this.tsb_Pen.Click += new System.EventHandler(this.tsb_Pen_Click);
            //
            // tsb_Marker
            //
            this.tsb_Marker.AutoSize = false;
            this.tsb_Marker.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_Marker.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Marker;
            this.tsb_Marker.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_Marker.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_Marker.Name = "tsb_Marker";
            this.tsb_Marker.Size = new System.Drawing.Size(28, 28);
            this.tsb_Marker.Text = "Marker";
            this.tsb_Marker.Click += new System.EventHandler(this.tsb_Marker_Click);
            //
            // tsb_Text
            //
            this.tsb_Text.AutoSize = false;
            this.tsb_Text.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_Text.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Text;
            this.tsb_Text.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_Text.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_Text.Name = "tsb_Text";
            this.tsb_Text.Size = new System.Drawing.Size(28, 28);
            this.tsb_Text.Text = "Text";
            this.tsb_Text.Click += new System.EventHandler(this.tsb_Text_Click);
            //
            // tsb_LineArrow
            //
            this.tsb_LineArrow.AutoSize = false;
            this.tsb_LineArrow.BackColor = System.Drawing.SystemColors.Control;
            this.tsb_LineArrow.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.EmptyImage;
            this.tsb_LineArrow.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.None;
            this.tsb_LineArrow.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_LineArrow.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.tsmi_Line,
            this.tsmi_Arrow});
            this.tsb_LineArrow.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Line;
            this.tsb_LineArrow.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_LineArrow.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_LineArrow.Name = "tsb_LineArrow";
            this.tsb_LineArrow.Size = new System.Drawing.Size(43, 28);
            this.tsb_LineArrow.Text = "Lines";
            this.tsb_LineArrow.ButtonClick += new
System.EventHandler(this.tsb_LineArrow_ButtonClick);
```

```
            //
            // tsmi_Line
            //
            this.tsmi_Line.BackColor = System.Drawing.SystemColors.Control;
            this.tsmi_Line.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Line;
            this.tsmi_Line.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsmi_Line.Name = "tsmi_Line";
            this.tsmi_Line.Size = new System.Drawing.Size(114, 30);
            this.tsmi_Line.Text = "Line";
            this.tsmi_Line.Click += new System.EventHandler(this.tsmi_Line_Click);
            //
            // tsmi_Arrow
            //
            this.tsmi_Arrow.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Arrow;
            this.tsmi_Arrow.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsmi_Arrow.Name = "tsmi_Arrow";
            this.tsmi_Arrow.Size = new System.Drawing.Size(114, 30);
            this.tsmi_Arrow.Text = "Arrow";
            this.tsmi_Arrow.Click += new System.EventHandler(this.tsmi_Arrow_Click);
            //
            // tsb_Rectangle
            //
            this.tsb_Rectangle.AutoSize = false;
            this.tsb_Rectangle.BackColor = System.Drawing.SystemColors.Control;
            this.tsb_Rectangle.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.EmptyImage;
            this.tsb_Rectangle.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.None;
            this.tsb_Rectangle.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_Rectangle.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.tsmi_Rectangle,
            this.tsmi_FillRectangle});
            this.tsb_Rectangle.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Rectangle;
            this.tsb_Rectangle.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_Rectangle.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_Rectangle.Name = "tsb_Rectangle";
            this.tsb_Rectangle.Size = new System.Drawing.Size(43, 28);
            this.tsb_Rectangle.Text = "Rectangle";
            this.tsb_Rectangle.ButtonClick += new
System.EventHandler(this.tsb_Rectangle_ButtonClick);
            //
            // tsmi_Rectangle
            //
            this.tsmi_Rectangle.BackColor = System.Drawing.SystemColors.Control;
            this.tsmi_Rectangle.Image =
((System.Drawing.Image)(resources.GetObject("tsmi_Rectangle.Image")));
            this.tsmi_Rectangle.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsmi_Rectangle.Name = "tsmi_Rectangle";
            this.tsmi_Rectangle.Size = new System.Drawing.Size(149, 30);
            this.tsmi_Rectangle.Text = "Rectangle";
            this.tsmi_Rectangle.Click += new
System.EventHandler(this.tsmi_Rectangle_Click);
            //
```

```
            // tsmi_FillRectangle
            //
            this.tsmi_FillRectangle.Image =
((System.Drawing.Image)(resources.GetObject("tsmi_FillRectangle.Image")));
            this.tsmi_FillRectangle.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsmi_FillRectangle.Name = "tsmi_FillRectangle";
            this.tsmi_FillRectangle.Size = new System.Drawing.Size(149, 30);
            this.tsmi_FillRectangle.Text = "Fill rectangle";
            this.tsmi_FillRectangle.Click += new
System.EventHandler(this.tsmi_FillRectangle_Click);
            //
            // tsb_Ellipse
            //
            this.tsb_Ellipse.AutoSize = false;
            this.tsb_Ellipse.BackgroundImage =
global::VarietyScreenRecorder.Properties.Resources.EmptyImage;
            this.tsb_Ellipse.BackgroundImageLayout =
System.Windows.Forms.ImageLayout.None;
            this.tsb_Ellipse.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_Ellipse.DropDownItems.AddRange(new
System.Windows.Forms.ToolStripItem[] {
            this.tsmi_Ellipse,
            this.tsmi_FillEllipse});
            this.tsb_Ellipse.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Ellipse;
            this.tsb_Ellipse.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_Ellipse.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_Ellipse.Name = "tsb_Ellipse";
            this.tsb_Ellipse.Size = new System.Drawing.Size(43, 28);
            this.tsb_Ellipse.Text = "Ellipse";
            this.tsb_Ellipse.ButtonClick += new
System.EventHandler(this.tsb_Ellipse_ButtonClick);
            //
            // tsmi_Ellipse
            //
            this.tsmi_Ellipse.Image =
((System.Drawing.Image)(resources.GetObject("tsmi_Ellipse.Image")));
            this.tsmi_Ellipse.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsmi_Ellipse.Name = "tsmi_Ellipse";
            this.tsmi_Ellipse.Size = new System.Drawing.Size(133, 30);
            this.tsmi_Ellipse.Text = "Ellipse";
            this.tsmi_Ellipse.Click += new
System.EventHandler(this.tsmi_Ellipse_Click);
            //
            // tsmi_FillEllipse
            //
            this.tsmi_FillEllipse.Image =
((System.Drawing.Image)(resources.GetObject("tsmi_FillEllipse.Image")));
            this.tsmi_FillEllipse.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsmi_FillEllipse.Name = "tsmi_FillEllipse";
            this.tsmi_FillEllipse.Size = new System.Drawing.Size(133, 30);
            this.tsmi_FillEllipse.Text = "Fill ellipse";
            this.tsmi_FillEllipse.Click += new
System.EventHandler(this.tsmi_FillEllipse_Click);
            //
            // toolStripSeparator1
            //
```

```
            this.toolStripSeparator1.Margin = new System.Windows.Forms.Padding(0, 10,
0, 10);
            this.toolStripSeparator1.Name = "toolStripSeparator1";
            this.toolStripSeparator1.Size = new System.Drawing.Size(41, 6);
            //
            // tsb_Color
            //
            this.tsb_Color.AutoSize = false;
            this.tsb_Color.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_Color.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_Color;
            this.tsb_Color.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_Color.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_Color.Name = "tsb_Color";
            this.tsb_Color.Size = new System.Drawing.Size(28, 28);
            this.tsb_Color.Text = "Color";
            this.tsb_Color.Click += new System.EventHandler(this.tsb_Color_Click);
            //
            // tsb_FillColor
            //
            this.tsb_FillColor.AutoSize = false;
            this.tsb_FillColor.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_FillColor.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_FillColor;
            this.tsb_FillColor.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_FillColor.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_FillColor.Name = "tsb_FillColor";
            this.tsb_FillColor.Size = new System.Drawing.Size(28, 28);
            this.tsb_FillColor.Text = "Fill color";
            this.tsb_FillColor.Click += new
System.EventHandler(this.tsb_FillColor_Click);
            //
            // tsb_LineWidth
            //
            this.tsb_LineWidth.AutoSize = false;
            this.tsb_LineWidth.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_LineWidth.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_LineWidth;
            this.tsb_LineWidth.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_LineWidth.ImageTransparentColor = System.Drawing.Color.Magenta;
            this.tsb_LineWidth.Name = "tsb_LineWidth";
            this.tsb_LineWidth.Size = new System.Drawing.Size(28, 28);
            this.tsb_LineWidth.Text = "Line width";
            this.tsb_LineWidth.Click += new
System.EventHandler(this.tsb_LineWidth_Click);
            //
            // tsb_TextSize
            //
            this.tsb_TextSize.AutoSize = false;
            this.tsb_TextSize.DisplayStyle =
System.Windows.Forms.ToolStripItemDisplayStyle.Image;
            this.tsb_TextSize.Image =
global::VarietyScreenRecorder.Properties.Resources.GE_TextSize;
            this.tsb_TextSize.ImageScaling =
System.Windows.Forms.ToolStripItemImageScaling.None;
            this.tsb_TextSize.ImageTransparentColor = System.Drawing.Color.Magenta;
```

```
            this.tsb_TextSize.Name = "tsb_TextSize";
            this.tsb_TextSize.Size = new System.Drawing.Size(28, 28);
            this.tsb_TextSize.Text = "Text size";
            this.tsb_TextSize.Click += new
System.EventHandler(this.tsb_TextSize_Click);
            //
            // tlp_MainArea
            //
            this.tlp_MainArea.BackColor = System.Drawing.Color.White;
            this.tlp_MainArea.ColumnCount = 3;
            this.tlp_MainArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 100F));
            this.tlp_MainArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainArea.Controls.Add(this.p_PictureArea, 1, 1);
            this.tlp_MainArea.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tlp_MainArea.Location = new System.Drawing.Point(44, 24);
            this.tlp_MainArea.Name = "tlp_MainArea";
            this.tlp_MainArea.RowCount = 3;
            this.tlp_MainArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 100F));
            this.tlp_MainArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_MainArea.Size = new System.Drawing.Size(690, 387);
            this.tlp_MainArea.TabIndex = 2;
            //
            // p_PictureArea
            //
            this.p_PictureArea.AutoScroll = true;
            this.p_PictureArea.BackColor = System.Drawing.Color.White;
            this.p_PictureArea.Controls.Add(this.pb_Image);
            this.p_PictureArea.Dock = System.Windows.Forms.DockStyle.Fill;
            this.p_PictureArea.Location = new System.Drawing.Point(13, 13);
            this.p_PictureArea.Name = "p_PictureArea";
            this.p_PictureArea.Size = new System.Drawing.Size(664, 361);
            this.p_PictureArea.TabIndex = 0;
            //
            // pb_Image
            //
            this.pb_Image.BorderStyle = System.Windows.Forms.BorderStyle.FixedSingle;
            this.pb_Image.Location = new System.Drawing.Point(0, 0);
            this.pb_Image.Name = "pb_Image";
            this.pb_Image.Size = new System.Drawing.Size(50, 50);
            this.pb_Image.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.AutoSize;
            this.pb_Image.TabIndex = 0;
            this.pb_Image.TabStop = false;
            this.pb_Image.Paint += new
System.Windows.Forms.PaintEventHandler(this.pb_Image_Paint);
            this.pb_Image.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.pb_Image_MouseDown);
            this.pb_Image.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.pb_Image_MouseMove);
            this.pb_Image.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.pb_Image_MouseUp);
            //
            // WindowGE_Main
            //
```

```
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.AutoScroll = true;
            this.ClientSize = new System.Drawing.Size(734, 411);
            this.Controls.Add(this.tlp_MainArea);
            this.Controls.Add(this.ts_ToolsButtons);
            this.Controls.Add(this.ms_Menu);
            this.KeyPreview = true;
            this.MainMenuStrip = this.ms_Menu;
            this.Name = "WindowGE_Main";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;
            this.Text = "Graphics Editor | Variety Screen Recorder";
            this.FormClosing += new
System.Windows.Forms.FormClosingEventHandler(this.WindowGE_Main_FormClosing);
            this.KeyDown += new
System.Windows.Forms.KeyEventHandler(this.WindowGE_Main_KeyDown);
            this.KeyUp += new
System.Windows.Forms.KeyEventHandler(this.WindowGE_Main_KeyUp);
            this.ms_Menu.ResumeLayout(false);
            this.ms_Menu.PerformLayout();
            this.ts_ToolsButtons.ResumeLayout(false);
            this.ts_ToolsButtons.PerformLayout();
            this.tlp_MainArea.ResumeLayout(false);
            this.p_PictureArea.ResumeLayout(false);
            this.p_PictureArea.PerformLayout();
            ((System.ComponentModel.ISupportInitialize)(this.pb_Image)).EndInit();
            this.ResumeLayout(false);
            this.PerformLayout();

        }

        #endregion

        private System.Windows.Forms.MenuStrip ms_Menu;
        private System.Windows.Forms.ToolStripMenuItem tsmi_File;
        private System.Windows.Forms.ToolStrip ts_ToolsButtons;
        private System.Windows.Forms.ToolStripButton tsb_Cursor;
        private System.Windows.Forms.ToolStripButton tsb_Marker;
        private System.Windows.Forms.ToolStripButton tsb_Pen;
        private System.Windows.Forms.ToolStripButton tsb_Color;
        private System.Windows.Forms.TableLayoutPanel tlp_MainArea;
        private System.Windows.Forms.Panel p_PictureArea;
        private System.Windows.Forms.PictureBox pb_Image;
        private System.Windows.Forms.ToolStripButton tsb_Text;
        private System.Windows.Forms.ToolStripSplitButton tsb_Rectangle;
        private System.Windows.Forms.ToolStripSplitButton tsb_Ellipse;
        private System.Windows.Forms.ToolStripButton tsb_FillColor;
        private System.Windows.Forms.ToolStripButton tsb_LineWidth;
        private System.Windows.Forms.ToolStripButton tsb_TextSize;
        private System.Windows.Forms.ToolStripSeparator toolStripSeparator1;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Rectangle;
        private System.Windows.Forms.ToolStripMenuItem tsmi_FillRectangle;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Ellipse;
        private System.Windows.Forms.ToolStripMenuItem tsmi_FillEllipse;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Save;
        private System.Windows.Forms.ToolStripMenuItem tsmi_SaveAs;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Edit;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Undo;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Redo;
        private System.Windows.Forms.ToolStripSplitButton tsb_LineArrow;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Line;
        private System.Windows.Forms.ToolStripMenuItem tsmi_Arrow;
```

```
    }
}
```

WindowGE_GetSize.cs

```
using System;
using System.Windows.Forms;

namespace VarietyScreenRecorder.Window_GE
{
    public partial class WindowGE_GetSize : Form
    {
        public bool isSet { get; set; } = false;

        public WindowGE_GetSize(string Text, int StartValue)
        {
            this.Icon = Properties.Resources.Logo;

            InitializeComponent();

            l_Text.Text = Text;
            nud_Size.Value = StartValue;
        }

        private void b_Set_Click(object sender, EventArgs e)
        {
            isSet = true;
            this.Close();
        }

        private void b_Cancel_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

WindowGE_GetSize.Designer.cs

```
namespace VarietyScreenRecorder.Window_GE
{
    partial class WindowGE_GetSize
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
```

```
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.l_Text = new System.Windows.Forms.Label();
            this.nud_Size = new System.Windows.Forms.NumericUpDown();
            this.tlp_FullArea = new System.Windows.Forms.TableLayoutPanel();
            this.b_Set = new System.Windows.Forms.Button();
            this.b_Cancel = new System.Windows.Forms.Button();
            ((System.ComponentModel.ISupportInitialize)(this.nud_Size)).BeginInit();
            this.tlp_FullArea.SuspendLayout();
            this.SuspendLayout();
            //
            // l_Text
            //
            this.l_Text.AutoSize = true;
            this.tlp_FullArea.SetColumnSpan(this.l_Text, 3);
            this.l_Text.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_Text.Location = new System.Drawing.Point(13, 10);
            this.l_Text.Name = "l_Text";
            this.l_Text.Size = new System.Drawing.Size(158, 27);
            this.l_Text.TabIndex = 0;
            this.l_Text.Text = "Your text";
            this.l_Text.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            //
            // nud_Size
            //
            this.tlp_FullArea.SetColumnSpan(this.nud_Size, 3);
            this.nud_Size.Dock = System.Windows.Forms.DockStyle.Fill;
            this.nud_Size.Location = new System.Drawing.Point(13, 50);
            this.nud_Size.Minimum = new decimal(new int[] {
            1,
            0,
            0,
            0});
            this.nud_Size.Name = "nud_Size";
            this.nud_Size.Size = new System.Drawing.Size(158, 20);
            this.nud_Size.TabIndex = 1;
            this.nud_Size.Value = new decimal(new int[] {
            1,
            0,
            0,
            0});
            //
            // tlp_FullArea
            //
            this.tlp_FullArea.ColumnCount = 5;
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 50F));
```

```
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 50F));
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.Controls.Add(this.nud_Size, 1, 3);
            this.tlp_FullArea.Controls.Add(this.l_Text, 1, 1);
            this.tlp_FullArea.Controls.Add(this.b_Set, 1, 5);
            this.tlp_FullArea.Controls.Add(this.b_Cancel, 3, 5);
            this.tlp_FullArea.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tlp_FullArea.Location = new System.Drawing.Point(0, 0);
            this.tlp_FullArea.Name = "tlp_FullArea";
            this.tlp_FullArea.RowCount = 7;
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 30F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 30F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 40F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.Size = new System.Drawing.Size(184, 131);
            this.tlp_FullArea.TabIndex = 2;
            //
            // b_Set
            //
            this.b_Set.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_Set.Location = new System.Drawing.Point(13, 87);
            this.b_Set.Name = "b_Set";
            this.b_Set.Size = new System.Drawing.Size(71, 30);
            this.b_Set.TabIndex = 2;
            this.b_Set.Text = "Set";
            this.b_Set.UseVisualStyleBackColor = true;
            this.b_Set.Click += new System.EventHandler(this.b_Set_Click);
            //
            // b_Cancel
            //
            this.b_Cancel.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_Cancel.Location = new System.Drawing.Point(100, 87);
            this.b_Cancel.Name = "b_Cancel";
            this.b_Cancel.Size = new System.Drawing.Size(71, 30);
            this.b_Cancel.TabIndex = 3;
            this.b_Cancel.Text = "Cancel";
            this.b_Cancel.UseVisualStyleBackColor = true;
            this.b_Cancel.Click += new System.EventHandler(this.b_Cancel_Click);
            //
            // WindowGE_GetSize
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(184, 131);
            this.Controls.Add(this.tlp_FullArea);
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;
            this.Name = "WindowGE_GetSize";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterParent;
```

```
            ((System.ComponentModel.ISupportInitialize)(this.nud_Size)).EndInit();
            this.tlp_FullArea.ResumeLayout(false);
            this.tlp_FullArea.PerformLayout();
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.Label l_Text;
        private System.Windows.Forms.TableLayoutPanel tlp_FullArea;
        private System.Windows.Forms.Button b_Set;
        private System.Windows.Forms.Button b_Cancel;
        public System.Windows.Forms.NumericUpDown nud_Size;
    }
}
```

WindowGE_GetText.cs

```
using System;
using System.Windows.Forms;

namespace VarietyScreenRecorder.Window_GE
{
    public partial class WindowGE_GetText : Form
    {
        public bool isSet { get; set; } = false;

        public WindowGE_GetText(string Text, string ActiveText)
        {
            this.Icon = Properties.Resources.Logo;

            InitializeComponent();

            l_Text.Text = Text;
            tb_TextValue.Text = ActiveText;
        }

        private void tb_TextValue_TextChanged(object sender, EventArgs e)
        {
            if (tb_TextValue.Text == "")
                b_Set.Enabled = false;
            else
                b_Set.Enabled = true;
        }

        private void b_Set_Click(object sender, EventArgs e)
        {
            isSet = true;
            this.Close();
        }

        private void b_Cancel_Click(object sender, EventArgs e)
        {
            this.Close();
        }
    }
}
```

WindowGE_GetText.Designer.cs

```
namespace VarietyScreenRecorder.Window_GE
{
    partial class WindowGE_GetText
    {
        /// <summary>
        /// Required designer variable.
        /// </summary>
        private System.ComponentModel.IContainer components = null;

        /// <summary>
        /// Clean up any resources being used.
        /// </summary>
        /// <param name="disposing">true if managed resources should be disposed;
otherwise, false.</param>
        protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
            }
            base.Dispose(disposing);
        }

        #region Windows Form Designer generated code

        /// <summary>
        /// Required method for Designer support - do not modify
        /// the contents of this method with the code editor.
        /// </summary>
        private void InitializeComponent()
        {
            this.tlp_FullArea = new System.Windows.Forms.TableLayoutPanel();
            this.l_Text = new System.Windows.Forms.Label();
            this.tb_TextValue = new System.Windows.Forms.TextBox();
            this.b_Set = new System.Windows.Forms.Button();
            this.b_Cancel = new System.Windows.Forms.Button();
            this.tlp_FullArea.SuspendLayout();
            this.SuspendLayout();
            //
            // tlp_FullArea
            //
            this.tlp_FullArea.ColumnCount = 5;
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 50F));
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Percent, 50F));
            this.tlp_FullArea.ColumnStyles.Add(new
System.Windows.Forms.ColumnStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.Controls.Add(this.l_Text, 1, 1);
            this.tlp_FullArea.Controls.Add(this.tb_TextValue, 1, 3);
            this.tlp_FullArea.Controls.Add(this.b_Set, 1, 5);
            this.tlp_FullArea.Controls.Add(this.b_Cancel, 3, 5);
            this.tlp_FullArea.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tlp_FullArea.Location = new System.Drawing.Point(0, 0);
```

```
            this.tlp_FullArea.Name = "tlp_FullArea";
            this.tlp_FullArea.RowCount = 7;
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 30F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 30F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Percent, 40F));
            this.tlp_FullArea.RowStyles.Add(new
System.Windows.Forms.RowStyle(System.Windows.Forms.SizeType.Absolute, 10F));
            this.tlp_FullArea.Size = new System.Drawing.Size(284, 131);
            this.tlp_FullArea.TabIndex = 0;
            //
            // l_Text
            //
            this.l_Text.AutoSize = true;
            this.tlp_FullArea.SetColumnSpan(this.l_Text, 3);
            this.l_Text.Dock = System.Windows.Forms.DockStyle.Fill;
            this.l_Text.Location = new System.Drawing.Point(13, 10);
            this.l_Text.Name = "l_Text";
            this.l_Text.Size = new System.Drawing.Size(258, 27);
            this.l_Text.TabIndex = 0;
            this.l_Text.Text = "label1";
            this.l_Text.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            //
            // tb_TextValue
            //
            this.tlp_FullArea.SetColumnSpan(this.tb_TextValue, 3);
            this.tb_TextValue.Dock = System.Windows.Forms.DockStyle.Fill;
            this.tb_TextValue.Location = new System.Drawing.Point(13, 50);
            this.tb_TextValue.Name = "tb_TextValue";
            this.tb_TextValue.Size = new System.Drawing.Size(258, 20);
            this.tb_TextValue.TabIndex = 1;
            this.tb_TextValue.TextChanged += new
System.EventHandler(this.tb_TextValue_TextChanged);
            //
            // b_Set
            //
            this.b_Set.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_Set.Location = new System.Drawing.Point(13, 87);
            this.b_Set.Name = "b_Set";
            this.b_Set.Size = new System.Drawing.Size(121, 30);
            this.b_Set.TabIndex = 2;
            this.b_Set.Text = "Set";
            this.b_Set.UseVisualStyleBackColor = true;
            this.b_Set.Click += new System.EventHandler(this.b_Set_Click);
            //
            // b_Cancel
            //
            this.b_Cancel.Dock = System.Windows.Forms.DockStyle.Fill;
            this.b_Cancel.Location = new System.Drawing.Point(150, 87);
            this.b_Cancel.Name = "b_Cancel";
            this.b_Cancel.Size = new System.Drawing.Size(121, 30);
            this.b_Cancel.TabIndex = 3;
            this.b_Cancel.Text = "Cancel";
            this.b_Cancel.UseVisualStyleBackColor = true;
```

```
            this.b_Cancel.Click += new System.EventHandler(this.b_Cancel_Click);
            //
            // WindowGE_GetText
            //
            this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new System.Drawing.Size(284, 131);
            this.Controls.Add(this.tlp_FullArea);
            this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedSingle;
            this.Name = "WindowGE_GetText";
            this.StartPosition = System.Windows.Forms.FormStartPosition.CenterParent;
            this.tlp_FullArea.ResumeLayout(false);
            this.tlp_FullArea.PerformLayout();
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.TableLayoutPanel tlp_FullArea;
        private System.Windows.Forms.Label l_Text;
        private System.Windows.Forms.Button b_Set;
        private System.Windows.Forms.Button b_Cancel;
        public System.Windows.Forms.TextBox tb_TextValue;
    }
}
```