**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

О. В. Чопорова, С. В. Чопоров

**ЛЮДИНО-МАШИННИЙ ІНТЕРФЕЙС**

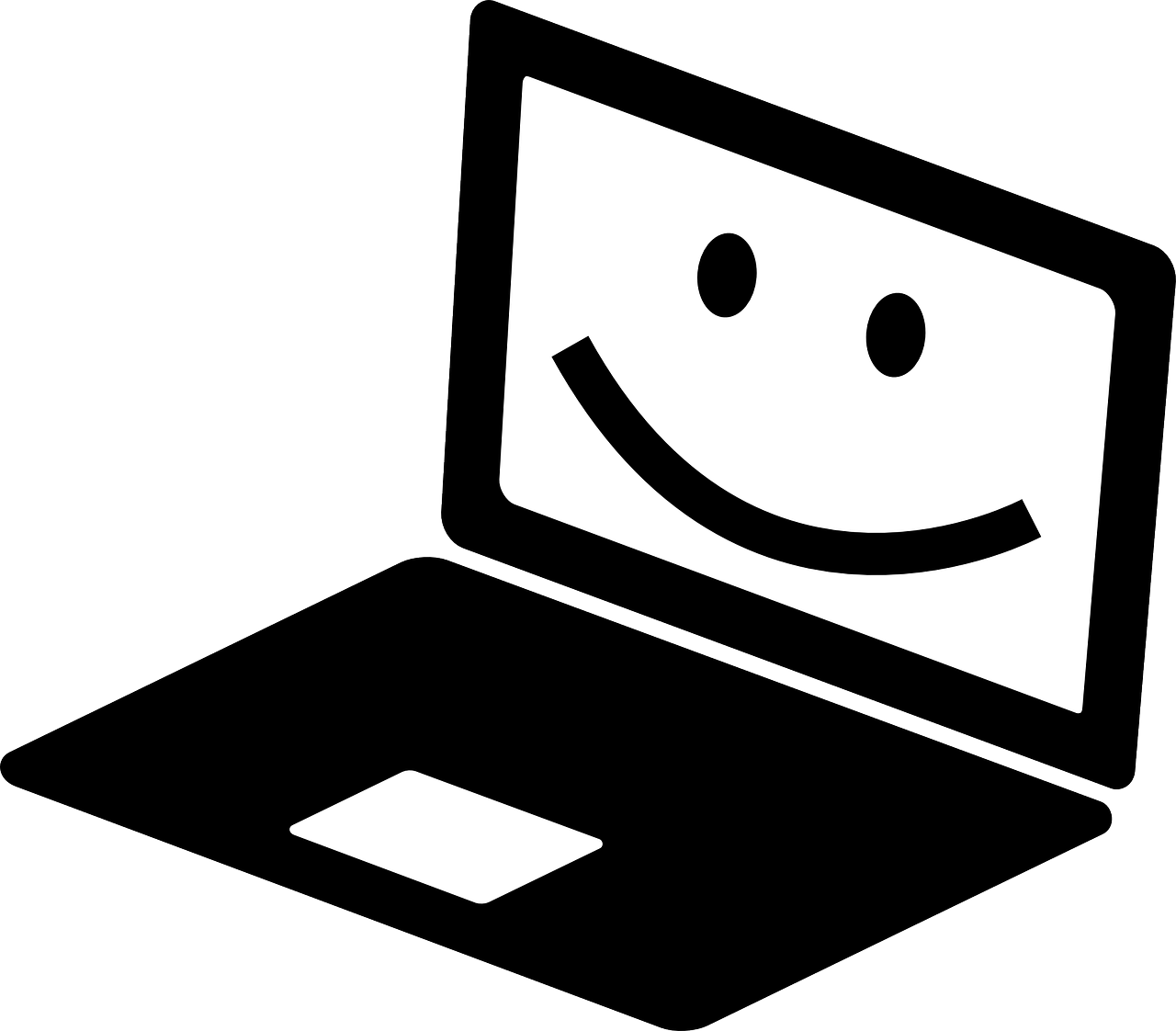
Методичні рекомендації до лабораторних робіт

для здобувачів ступеня вищої освіти бакалавра  
спеціальності “Інженерія програмного забезпечення”  
освітньо-професійної програми “Програмна інженерія”

Затверджено

Вченою радою ЗНУ

Протокол № 13 від 28.06.2022 р.



Запоріжжя

2022

УДК: 004.5(076.5)

Ч-754

Чопорова О. В., Чопоров С. В. Людино-машинний інтерфейс: методичні рекомендації до лабораторних робіт для здобувачів ступеня вищої освіти бакалавра спеціальності “Інженерія програмного забезпечення” освітньо-професійної програми “Програмна інженерія”. Запоріжжя: ЗНУ, 2022. 28 с.

Методичні рекомендації до виконання лабораторних робіт з дисципліни «Людино-машинний інтерфейс» пропонується студентам, що навчаються за програмою освітньо-кваліфікаційного рівня «бакалавр» за напрямом підготовки «Програмна інженерія» та рекомендується для закріплення навчального матеріалу з дисципліни циклу професійної та практичної підготовки.

Вказівки включають теоретичний матеріал з кожної теми дисципліни та зміст основних лабораторних робіт, що пропонуються при вивченні даного курсу, а також контрольні питання. Увага студентів акцентується на набутті практичних навичок розробки людино-машинних інтерфейсів.

Методичні вказівки призначені для студентів, викладачів, а також усіх, хто цікавиться питаннями зазначеної тематики.

Рецензент *С. М. Гребенюк*

Відповідальний за випуск *С. В. Чопоров*

**Зміст**

[Вступ 4](#_Toc128987792)

[Лабораторна робота № 1 Розробка інтерфейсу програми для значення функції 5](#_Toc128987793)

[Лабораторна робота № 2 Розробка інтерфейсу програми для обчислення таблиці значень функції 6](#_Toc128987794)

[Лабораторна робота № 3 Розробка графічного компонента 8](#_Toc128987795)

[Лабораторна робота № 4 Створення додатку на базі технологій Qt Quick 9](#_Toc128987796)

[Лабораторна робота № 5 QML: графічні елементи та прямокутники 11](#_Toc128987797)

[Лабораторна робота № 6 Текст у графічних елементах 14](#_Toc128987798)

[Лабораторна робота № 7 Використання MouseArea для роботи з мишею 17](#_Toc128987799)

[Лабораторна робота № 8 Рух об'єктів інтерфейсу 19](#_Toc128987800)

[Лабораторна робота № 9 Анімація 20](#_Toc128987801)

[Лабораторна робота № 10 Робота з таймером 21](#_Toc128987802)

[Індивідуальне завдання 22](#_Toc128987803)

[Завдання для самостійної роботи 24](#_Toc128987804)

[Глосарій 25](#_Toc128987805)

[Список літератури 26](#_Toc128987806)

# Вступ

Людино-машинний інтерфейс – це область комп’ютерної науки і технології, яка займається вивченням особливостей проектування та реалізації інтерфейсів для управління програмно-технічними засобами.

Дисципліна, що має справу з розробкою, розвитком і застосуванням інтерактивних комп’ютерних систем з точки зору вимог користувача, а також з вивченням явищ, що їх оточують. Ця дисципліна призначена забезпечити вивчення комп’ютерних технологій з акцентом на розробку і розвиток користувальницького інтерфейсу.

**Мета курсу:** надання майбутнім фахівцям компетенцій сучасних методів та засобів створення інтерфейсів прикладного програмного забезпечення автоматизованих інформаційних систем на базі використання різноманітних сучасних програмних засобів. Вивчення моделей інтерфейсів користувача, правил проектування інтерфейсів користувача, стандартів інтерфейсів, розгляд особливостей взаємодії людини та комп’ютера, а також формування теоретичних та практичних знань, які можуть бути використані при подальшому навчанні, професійній, виробничій та науковій діяльності випускника вузу. Дисципліна «Людино-машинний інтерфейс» базується на знаннях дисциплін «Методи та засоби комп’ютерних інформаційних технологій», «Основи програмування та інформаційна культура студентів», «Архітектура комп’ютера», «Об’єктно-орієнтоване програмування», «Основи програмної інженерії» та інших. Методичні рекомендації містять 10 лабораторних робіт, у яких потрібно розробити програмні засоби, які реалізують графічні інтерфейси для взаємодії людини та комп’ютера. Кожна лабораторна робота оцінюється у 6 білів (2 бали за належне оформлення звіту, 3 бали за виконання суті роботи і 1 бал за своєчасне виконання роботи).

**Завдання навчальної дисципліни:** оволодіти основними принципами та методами проектування людино-машинних інтерфейсів. Дати необхідну практичну підготовку та знання для подальшого їх застосування при вивченні спеціальних дисциплін та професійній підготовці.

# Розробка інтерфейсу програми для значення функції

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи рекомендується використовувати такі компоненти фреймворка Qt:

* QLabel, <https://doc.qt.io/qt-6/qlabel.html>
* QLayout, <https://doc.qt.io/qt-6/qlayout.html>
* QPushButton, <https://doc.qt.io/qt-6/qpushbutton.html>
* QDoubleSpinBox, <https://doc.qt.io/qt-6/qdoublespinbox.html>
* QSpinBox, <https://doc.qt.io/qt-6/qspinbox.html>
* QRadioButton, <https://doc.qt.io/qt-6/qradiobutton.html>
* QMessageBox, <https://doc.qt.io/qt-6/qmessagebox.html>

## ፨ Завдання для лабораторної роботи

Розробити інтерфейс програми для обчислення значення обраної користувачем функції у заданій точці. Користувач має обирати одну з можливих альтернатив, вводити точку для обчислення цієї функції. Компонування елементів інтерфейсу повинно бути адаптивним до розмірів вікна. Про результат обчислення за допомогою окремого повідомлення та шляхом збереження його біля рядка з функцію. Повідомлення повинне адекватно подавати про функцію, точку, у якій виконані обчислення, та результат.

## ❓ Контрольні запитання

1. Дайте визначення людино-машинного інтерфейсу.
2. Чим відрізняється людино-машинний інтерфейс від користувальницького?
3. Які стилі доступні для кнопок?
4. Які можливі режими компонування об’єктів інтерфейсу?
5. Які основні етапи проектування і реалізації користувальницького інтерфейсу?

# Розробка інтерфейсу програми для обчислення таблиці значень функції

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи рекомендовано використовувати наступні компоненти фреймворку Qt:

* QLabel, <https://doc.qt.io/qt-6/qlabel.html>
* QLayout, <https://doc.qt.io/qt-6/qlayout.html>
* QPushButton, <https://doc.qt.io/qt-6/qpushbutton.html>
* QDoubleSpinBox, <https://doc.qt.io/qt-6/qdoublespinbox.html>
* QSpinBox, <https://doc.qt.io/qt-6/qspinbox.html>
* QRadioButton, <https://doc.qt.io/qt-6/qradiobutton.html>
* QMessageBox, <https://doc.qt.io/qt-6/qmessagebox.html>
* QTableWidget, <https://doc.qt.io/qt-6/qtablewidget.html>

## ፨ Завдання для лабораторної роботи

Розробити інтерфейс програми для обчислення таблиці значень обраної користувачем функції. Користувач має обирати одну з можливих альтернатив функції для обчислення, вводити інтервал (початок і кінець) і кількість точок на інтервалі.

Область вибору функції

Область вибору функції

Область параметрів

Таблиця

Постійна висота

Адаптивна висота

Ширина усіх компонентів адаптивна

Програма повинна з рівномірним кроком обчислити значення функції та заповнити ними таблицю на екранній формі. Якщо у деякі точці виникне виключна ситуація, пов’язана з розривом функції, то за допомогою механізму обробки таких ситуації вивести відповідне повідомлення. Необхідно забезпечити контроль за коректністю введених даних: ліва границя менша за праву границю інтервалу (дійсні числа), кількість точок більша або рівна 2. Інтерфейс повинен бути адаптивним до розмірів екранної форми за наступною схемою:

## ❓ Контрольні запитання

1. Яким чином можна забезпечити «прилипання» контролерів до певного краю?
2. Назвіть основні параметри компонента QTableWidget?
3. Порівняйте QTableWidget з QTableView.
4. Яким чином зробити автоматичну зміну кількості рядків QTableWidget при зміні значення компонента QSpinBox?
5. Яким чином заборонити зміни значень в таблиці QTableWidget?

# Розробка графічного компонента

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи рекомендується використовувати такі класи фреймворка Qt:

* QPainter <https://doc.qt.io/qt-5/qpainter.html>
* QWidget <https://doc.qt.io/qt-6/qwidget.html>

## ፨ Завдання для лабораторної роботи

Розробити інтерфейс програми для побудови графіку обраної користувачем функції. Користувач має обирати одну з можливих альтернатив функції для візуалізації, вводити інтервал (початок і кінець) і кількість точок на інтервалі. Програма повинна з рівномірним кроком обчислити значення функції та заповнити ними таблицю на екранній формі та побудувати її графік. Якщо у деякі точці виникне виключна ситуація, пов’язана з розривом функції, то за допомогою механізму обробки таких ситуації вивести відповідне повідомлення. Необхідно забезпечити контроль за коректністю введених даних: ліва границя менша за праву границю інтервалу (дійсні числа), кількість точок більша або рівна 2. Інтерфейс повинен бути адаптивним до розмірів екранної форми. Для візуалізації графіку розробити власний графічний компонент використовуючи QPainter і QWidget. Для розробленого віджета запропонувати інтерфейс для налаштування параметрів візуалізації (кольору, товщини й нарису лінії).

## ❓ Контрольні запитання

1. Яким чином можна забезпечити «прилипання» контролерів до певного краю?
2. Назвіть основні параметри компонента QTableWidget?
3. Порівняйте QTableWidget з QTableView.
4. Яким чином зробити автоматичну зміну кількості рядків QTableWidget при зміні значення компонента QSpinBox?
5. Яким чином заборонити зміни значень в таблиці QTableWidget?

# Створення додатку на базі технологій Qt Quick

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи рекомендовано використовувати інтегроване середовище Qt Creator([https://www.qt.io/product/ development-tools](https://www.qt.io/product/development-tools)), що спроектоване таким чином, щоб процес проектування був інтуїтивно зрозумілим. Програмісту-проектувальнику надаються можливості як «у ручному режимі» проектувати інтерфейс та логіку програмного засобу, так і використовуючи засоби візуального редактору та принципи RAD[[1]](#footnote-1). Окремою принциповою можливістю середовища розробки є можливість проектування адаптивних інтерфейсів на базі декларативної мови QML[[2]](#footnote-2), що дозволяє відокремити роботи зі створення інтерфейсів та програмування поведінки програмного засобу.

QML – декларативна мова програмування, заснована на JavaScript[[3]](#footnote-3), призначена для дизайну інтерфейсу користувача додатків.

Для створення проекту на базі технологій QML необхідно обрати пункт New File or Project меню File. Далі у вікні New Qt Quick Application на кроці Location у поле Name ввести назву проекту, а у поле Create in адресу директорії на диску для створення проекту (рис. 1). На кроці Component Set необхідно обрати множину візуальних компонентів (як правило доступно версій версій множин компонентів, рекомендується використовувати Qt Quick 2.0). На кроці Summary можна встановити систему контролю версій.

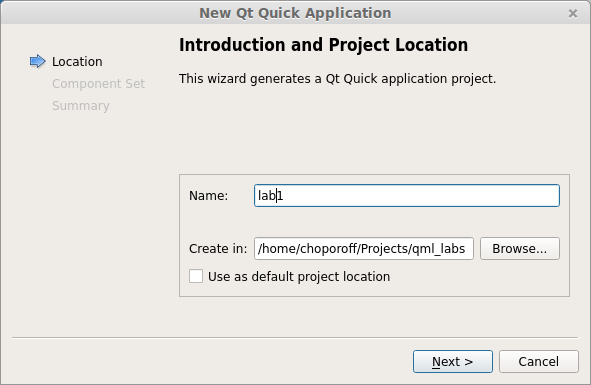


Рисунок 1 – Вікно New Qt Quick Application

За замовчуванням у віртуальній директорії qml створеного проекту буде знаходитися файл main.qml, до я кого можна дістатися використавши дерево проекту в режимі Edit на боковій панелі середовища Qt Creator. При цьому в редакторі буде доступний для редагування qml-код головного вікна проекту (лістинг 1).

Лістинг 1 – Код головного вікна

1. import QtQuick 2.0
2. Rectangle {
3. width: 360
4. height: 360
5. Text {
6. text: *qsTr*("Hello World")
7. anchors.centerIn: *parent*
8. }
9. MouseArea {
10. anchors.fill: *parent*
11. onClicked: {
12. *Qt*.quit();
13. }
14. }
15. }

Редагування кода головного вікна можливе також у режимі дизайнера (режим Design бокової панелі).

Для компіляції та запуску програми пунктом Run (Ctrl+R) меню Build.

## ፨ Завдання для лабораторної роботи

1. Створити новий проект Qt Quick.
2. Змінити його таким чином, щоб головне вікно містило інформацію про студента, який виконав роботу: прізвище, ім’я та по-батькові – напівжирний зелений текст; номер групи – курсив фіолетового кольору.

## ❓ Контрольні запитання

1. Яке призначення мови QML?
2. Які параметри базові компоненти можна додавати в режимі дизайнера?
3. Які параметри візуальних компонентів можна змінювати в режимі дизайнера?
4. Які рядки коду відповідають встановленим у дизайнері параметрам?
5. Яким чином організоване завершення програми?

# QML: графічні елементи та прямокутники

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи рекомендовано використовувати мову QML. Декларативна мова QML надає розробнику ієрархію графічних елементів, шляхом розширення та/або композиції яких проектується інтерфейс користувача. Базовим графічним елементом є Item, який містить властивості спільні для всіх графічних елементів і дозволяє компонувати графічні елементи у нові візуальні компоненти. Основні властивості елементу Item наведено в табл. 1.

Таблиця 1 – Базові властивості елемента Item

| Властивість | Тип | Призначення |
| --- | --- | --- |
| antialiasing | bool | Вмикає або вимикає згладжування. За замовчуванням – false. |
| enabled | bool | Вмикає або вимикає обробку події пристроїв вводу інформації. Має каскадний вплив на дочірні елементи. За замовчуванням – true. |
| height | real | Визначає висоту елемента. |
| opacity | real | Визначає прозорість елемента (значення у діапазоні від 0 – абсолютно прозорий до 1 – абсолютно непрозорий). За замовчуванням – 1. |
| rotation | real | Визначає кут обертання (у градусах) елемента за годинниковою стрілкою. За замовчуванням – 0. |
| visible | bool | Вмикає або вимикає візуалізацію елемента. За замовчуванням – true. |
| width | real | Визначає ширину елемента. |
| x | real | Абсциса верхнього правого кута елемента відносно батьківського елемента. За замовчеуванням – 0. |
| y | real | Ордината верхнього правого кута елемента відносно батьківського елемента. За замовчуванням – 0. |
| z | real | Номер візуального шару елемента (керує порядком візуалізації). За замовчуванням 0. |

Базовою можливістю QML є групування (композиція) об’єктів для створення нових примітивів графічного інтерфейсу користувача. Визначення властивостей об’єкта та вкладених у нього відбувається шляхом використання фігурних дужок: { – початок опису, } – кінець опису.

Одним з найбільш часто використовуваних графічних елементів QML є елемент Rectangle, який надає замальований прямокутник з опційною границею. Цей елемент дозволяє замалювати площину певним кольором або градієнтом і часто містить у собі інші графічні елементи. Елемент Rectangle є потомком елемента Item, тому для нього характерні всі властивості, наведені у табл. 1. Додатково Rectangle має властивості, наведені у табл. 2.

Таблиця 2 – Властивості елемента Rectangle

| Властивість | Тип | Призначення |
| --- | --- | --- |
| border.width | int | Визначає товщину границі: 0 – границя відсутня, 1 – тонка, тощо. |
| border.color | color | Визначає колір границі. |
| color | color | Визначає колір замальовування прямокутника. |
| gradient | Gradient | Визначає градієнт замальовування прямокутника. |
| radius | real | Визначає радіус скруглення кутів прямокутника. За замовчуванням – 0. |

Наприклад, розглянемо інтерфейс, який складається з елементу Item (лістинг 2), що містить об’єднання двох прямокутників (рис. 2): зеленого розміру 100×150 точок (рядки 3-11) та синього розміру 100×100 точок (рядки 12-21).

Лістинг 2 – Інтерфейс на базі об’єднання двох прямокутників

1. import QtQuick 2.0
2. Item {
3. Rectangle {
4. width: 100
5. height: 150
6. color: "#00B000"
7. Text {
8. text: *qsTr*("Left Rectangle")
9. anchors.left: *parent*
10. }
11. }
12. Rectangle {
13. x: 50
14. y: 100
15. width: 100
16. height: 100
17. color: "steelblue"
18. opacity: 0.5
19. border.width: 2
20. border.color: "tomato"
21. }
22. }

Для визначення кольору можна використовувати трикомпонентні або чотирьох компонентні коди у шістнадцятирічній системі числення (наприклад, рядок 6 лістингу 2), або назви кольорів згідно зі стандартом SVG[[4]](#footnote-4). У рядках 4-5 і 15-16 визначено ширину та висоту прямокутників. У рядках 13-14 зазначене початкове зміщення верхнього правого кута другого прямокутника.

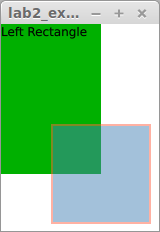


Рисунок 2 – Приклад роботи програми лістингу 2

## ፨ Завдання для лабораторної роботи

1. Створити новий проект Qt Quick.
2. Використовуючи елементи Item і Rectangle створити графічний інтерфейс за схемою, наведеною на рис. 3.

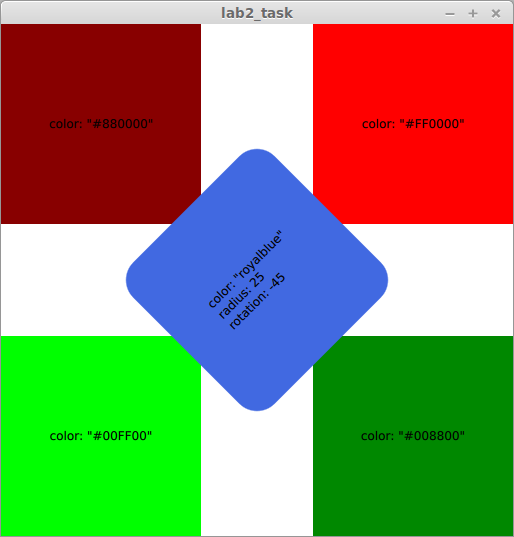


Рисунок 3 – Завдання до лабораторної роботи № 2

## ❓ Контрольні запитання

1. Поясніть, що визначає група властивостей anchor елементу Item.
2. Яким чином змінити порядок рисування графічних об’єктів?
3. У якому діапазоні можна змінювати значення радіуса кутів прямокутника?
4. Яким чином через колір можна визначити прозорість замальовування?

# Текст у графічних елементах

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи надані наступні рекомендації - для додавання відформатованого тексту до сцени в QML використовується елемент Text. За допомогою елемента Text можна використовувати у графічних елементах як звичайний текст, так і текст у “форматі збагаченого тексту”[[5]](#footnote-5) на базі тегів HTML. Елемент Text є потомком елемента Item, тому для нього характерні всі властивості, наведені у таблиці 1, а також властивості, наведені у таблиці 3.

Таблиця 3 – Властивості елемента Text

| Властивість | Тип | Призначення |
| --- | --- | --- |
| clip | bool | Індикатор того, що текст був обрізаний (якщо не вмістився у батьківський елемент). |
| color | color | Колір тексту. |
| elide | enumeration | Режим обрізання (укривання) тексту на випадок, якщо текст не вмістився у батьківський елемент. Можливі значення: Text.ElideNone, Text.ElideLeft, Text.ElideMiddle, Text.ElideRight. |
| font.bold | bool | Насиченість накреслення тексту. |
| font.capitalization | enumeration | Капіталізація тексту: Font.MixedCase – нормальна капіталізація тексту, Font.AllUppercase – капіталізація тексту великими літерами; Font.AllLowercase – капіталізація маленькими літерами; Font.SmallCaps – капіталізація великими літерами зменшеного розміру; Font.Capitalize – режим, у якому кожне слово друкується з великої літери. |
| font.family | string | Назва родини шрифтів. Наприклад, "Helvetica". |
| font.italic | bool | Курсив шрифту |
| font.pixelSize | int | Розмір шрифту в пікселях екрану |
| font.pointSize | real | Розмір шрифту в точках (незалежних роздільної здатності) |
| font.underline | bool | Вмикає режим підкреслення тексту |
| font.weight | enumeration | Режим накреслення літер тексту: Font.Light, Font.Normal (за замовченням), Font.DemiBold, Font.Bold, Font.Black |
| horizontalAligment | enumeration | Встановлюють горизонтальне вирівнювання тексту відносно ширини елементу. Можливі режими: Text.AlignLeft, Text.AlignRight, Text.AlignHCenter, Text.AlignJustify. |
| verticalAligment | enumeration | Встановлюють вертикальне вирівнювання тексту відносно висоти елементу. Можливі режими: Text.AlignTop, Text.AlignBottom, Text.AlignVCenter |
| hoveredLink | string | Посилання у форматі HTML або rich text. |
| style | enumeration | Стиль тексту. Можливі значення: Text.Normal, Text.Outline, Text.Raised, Text.Sunken |
| text | string | Текст |

Наприклад, розглянемо інтерфейс, який складається з прямокутної області, що містить текст повідомлення (рис. 4). У рядках 15-23 створено елемент Text, який центрований відносно батьківського прямокутника. Шрифт встановлено синього кольору, напівжирний, розміром 32 точки. Окремо можна відзначити використання елемента Gradient для створення градієнтного фону, який змінюється в зелених тонах.

Лістинг 3 – Елемент Text та його властивості

1. import QtQuick 2.0
2. Rectangle {
3. width: 400
4. height: 300
5. gradient: Gradient {
6. GradientStop {
7. position: 0.0;
8. color: "#005500"
9. }
10. GradientStop {
11. position: 1.0;
12. color: "#00FF00"
13. }
14. }
15. Text {
16. text: *qsTr*("Hello World")
17. color: "blue"
18. font.bold: true
19. font.pointSize: 32
20. anchors.centerIn: *parent*
21. style: Text.Sunken
22. styleColor: "white"
23. }
24. }



Рисунок 4 – Приклад роботи програми лістингу 3

## ፨ Завдання для лабораторної роботи

На базі програмного кода, розробленого у лабораторній роботі № 2, додати до інтерфейсу користувача текстові елементи за схемою, наведеною на рис. 5.

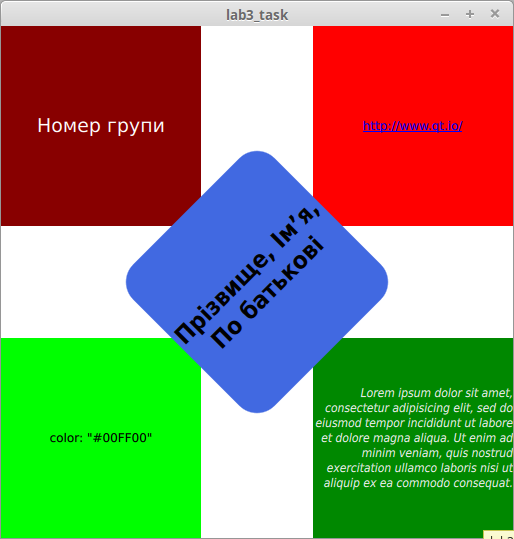


Рисунок 5 – Схема наповнення текстом елементів

## ❓ Контрольні запитання

1. Якими способами можливо задавати колір тексту?
2. Як повернути текст на кут 27°?
3. Як створити напівжирний текст?

# Використання MouseArea для роботи з мишею

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи рекомендовано використовувати елемент MouseArea([https://doc.qt.io/ qt-6/ qml-qtquick- mousearea.html](https://doc.qt.io/qt-6/qml-qtquick-mousearea.html)). Елемент MouseArea є невидимим елементом, який зазвичай використовується у поєднанні з деяким візуальним елементом для обробки подій, пов’язаних з маніпулятором миша. Елемент MouseArea є потомком елемента Item, тому для нього характерні всі властивості, наведені у таблиці 1, а також властивості, наведені у таблиці 4.

Таблиця 4 – Основні властивості елемента MouseArea

| Властивість | Тип | Призначення |
| --- | --- | --- |
| acceptedButtons | enumeration | Для визначення кнопок миші, події яких можуть оброблятися елементом. Можливі значення: Qt.LeftButton, Qt.RightButton, Qt.MiddleButton або Qt.AllButton. Значення за замовченням: Qt.LeftButton. |
| cursorShape | enumeration | Для визначення форми вказівника миші. Можливі значення: Qt.ArrowCursor (за замовченням), Qt.UpArrowCursor, Qt.CrossCursor, Qt.WaitCursor, Qt.IBeamCursor, Qt.SizeVerCursor, Qt.SizeHorCursor, Qt.SizeBDiagCursor, Qt.SizeFDiagCursor, Qt.SizeAllCursor, Qt.BlankCursor, Qt.SplitVCursor, Qt.SplitHCursor, Qt.PointingHandCursor, Qt.ForbiddenCursor, Qt.WhatsThisCursor, Qt.BusyCursor, Qt.OpenHandCursor, Qt.ClosedHandCursor, Qt.DrugCopyCursor, Qt.DragMoveCursor, Qt.DragLinkCursor. |
| hoverEnabled | bool | Керування режимом обробки подій миші. Якщо ця властивість false (за замовченням), то обробляються тільки події, пов’язані з кнопками, інакше додатково обробляються події переміщення курсору над елементом. |
| mouseX | real | Якщо hoverEnabled = true, то містить абсцису поточної координати курсору. |
| mouseY | real | Якщо hoverEnabled = true, то містить ординату поточної координати курсору. |
| pressed | bool | true, якщо натиснута одна з допустимих кнопок, інакше – false. |
| pressedButtons | enumeration | Бітова комбінація, відповідна сполученню натиснутих кнопок. Можливі значення: Qt.LeftButton, Qt.RightButton, Qt.MiddleButon. |

В основу підсистеми обробки подій QML покладено механізм сигналів й обробників. Сигнал –

## ፨ Завдання для лабораторної роботи

До інтерфейсу, розробленому в попередній лабораторній роботі, додати кнопку завершення роботи програми й палітру для зміни кольору центрального прямокутника. Палітру реалізувати як сім квадратів, відповідних кольорам веселки, розміром 15×15 точок. При натисканні лівої кнопки миші повинен встановлюватися обраний колір, а при натисканні правої – колір, на тон світліший за обраний.

## ❓ Контрольні запитання

1. Яке призначення властивості acceptedButtons?
2. Як змінити форму курсора, що потрапив до поточного простору миші?
3. Яким чином визначити клавішу миші, що була натиснута?
4. Наведіть схему визначення координат курсору миші.

# Рух об'єктів інтерфейсу

## ℹ Методичні рекомендації щодо виконання роботи

* 1. Для виконання лабораторної роботи рекомендовано використовувати наступні компоненти фреймворку Qt: QtQuick ([https://doc.qt.io/qt-6/qtquick- index.html](https://doc.qt.io/qt-6/qtquick-index.html)) та QtQuick.Controls ([https://doc.qt.io/qt-6/qtquickcontrols- index.html](https://doc.qt.io/qt-6/qtquickcontrols-index.html)).

## ፨ Завдання для лабораторної роботи

До інтерфейсу, розробленому в попередній лабораторній роботі, додати можливість переміщати об'єкти. Обраний об'єкт повинен відрізнятися від інших за допомогою маркерів вибору. Необхідно забечити контроль за коректністю переміщення (неможливість потрапляння за межі екранної форми).

## ❓ Контрольні запитання

1. Дати визначення людино-орієнтованого інтерфейсу.
2. Яке комплексне застосування законів Хіка і Фіттса?
3. Який кількісний аналіз інтерфейсу?
4. Дати визначення - вимірювання ефективності інтерфейсу.

# Анімація

## ℹ Методичні рекомендації щодо виконання роботи

* 1. Для виконання лабораторної роботи рекомендовано використовувати наступні компоненти фреймворку Qt: QtQuick ([https://doc.qt.io/qt-6/qtquick- index.html](https://doc.qt.io/qt-6/qtquick-index.html)), QtQuick.Controls ([https://doc.qt.io/qt-6/qtquickcontrols- index.html](https://doc.qt.io/qt-6/qtquickcontrols-index.html)) та MouseArea (<https://doc.qt.io/qt-6/qml-qtquick-mousearea.html>)

## ፨ Завдання для лабораторної роботи

До інтерфейсу, розробленому в попередній лабораторній роботі, додати анімацію обертання навколо власного центру для об’єкта, що переміщується.

## ❓ Контрольні запитання

1. Які існують програмні засоби для створення інтерфейсів?
2. Які існують моделі інтерфейсів?
3. Як правильно називаються елементи інтерфейсу?
4. Що таке швидкість інтерфейсу?

# Робота з таймером

## ℹ Методичні рекомендації щодо виконання роботи

Для виконання лабораторної роботи рекомендовано використовувати наступні компоненти фреймворку Qt:

* QtQuick <https://doc.qt.io/qt-6/qtquick-index.html>
* TableView <https://doc.qt.io/qt-6/qtableview.html>
* ListModel <https://doc.qt.io/qt-6/qml-qtqml-models-listmodel.html>
* Timer <https://doc.qt.io/qt-6/qtimer.html>

## ፨ Завдання для лабораторної роботи

Розробити програму яка дозволяє зберігати набір повідомлень з часом їх відображення. Необхідно забезпечити можливість нагадування як у встановлений час, так і через заданий проміжок часу.

## ❓ Контрольні запитання

1. В чому відмінність протоколу від інтерфейсу?
2. Чим клас відрізняється від інтерфейсу?
3. Чи можливо створити поля в інтерфейсі?
4. Яку мову використовують для написання інтерфейсів?

# Індивідуальне завдання

## ℹ Теоретичні відомості

Для виконання індивідуального завдання рекомендовано використовувати наступні компоненти:

* QtWidgets <https://doc.qt.io/qt-6/qtwidgets-index.html>
* QtCore <https://doc.qt.io/qt-6/qtcore-index.html>
* QtWidgets <https://doc.qt.io/qt-6/qtwidgets-index.html>
* QTextStream <https://doc.qt.io/qt-6/qtextstream.html>
* QTimer <https://doc.qt.io/qt-6/qtimer.html>

## ፨ Змістовна частина завдання

Зробити звіт про виконання індивідуального завдання, який повинен включати постановку завдання, математичну модель розв’язку задачі, блок-схему та набір тестових прикладів зі знімками екрану з роботою програми.

1. Запустити QtCreator. Створити програму Qt Widgets. Помістити на форму елементи: textEdit, Label, pushButton (4 штуки), verticalSpacer і скомпонувати.
2. Прописати процедури для кожної кнопки:

* Кнопка "Відкрити", objectName - open\_text. Текст повинен відкриватися як англійською, так і українською. Використовувати QTextStream.
* Кнопка "Зберегти", objectName - save\_text. Текст повинен зберігатися і читатися у файлі як англійською, так і українською. Використовувати QTextStream.
* Кнопка "Очистити", objectName - clear\_text.: використовуючи функцію clear() очистити вміст TextEdit.
* Кнопка "Вихід", objectName - exit\_text. Вихід із програми із попереднім збереженням вмісту textEdit на запит.

1. Реалізувати автозбереження у тимчасовий файл за допомогою таймера. Видавати інформацію про зроблене автозбереження. Після виходу з програми видаляти тимчасові файли.

Наприклад, можна зробити так:

Пишемо функцію автозбереження autosave(), в якій відбуватиметься збереження тексту з TextEdit у тому випадку, якщо в Label вже занесений шлях та ім'я файлу (за завданням у Label завжди буде вказано шлях з ім'ям поточного файлу). Ім'я тимчасового файлу формуватиме так: «ім'я\_temp». Додаємо повідомлення QMessageBox::information з інформацією про автозбереження (для перевірки роботи програми). MainWindow створюємо таймер і з'єднуємо сигнал timeout() (запускається після закінчення зазначеного проміжку часу) з нашою функцією.

1. Створити одноразовий таймер для показу логотипу вашої компанії після певного проміжку часу. Тобто, користувач відкрив програму, попрацював заданий проміжок часу і потім йому виходить повідомлення про те, що, наприклад, «Ви використовуєте нашу програму вже 10 хв. Ми раді, що вона вам сподобалася», та логотип. Також це може бути повідомлення, наприклад, про те, що термін дії пробної версії програми минув.

# Завдання для самостійної роботи

**Тестування якості людино-машинного інтерфейсу**

1. Задачі тестування користувальницького інтерфейсу;
2. Функціональне тестування інтерфейсів;
3. Зручність використання інтерфейсів.

**Тестування додатку ОС Android**

1. Основні етапи тестування додатків ОС Android.
2. Налаштування Android пристрою.
3. Тестування додатку на Android пристрої.

# Глосарій

**Людино-машинний інтерфейс** – термін, що охоплює інженерні рішення, котрі забезпечують взаємодію оператора з керованими ним машинами.

**Інтерфейс користувача** - сукупність засобів для обробки та відображення інформації, максимально пристосованих для зручності користувача.

**Користувач** – це особа, яка використовує вже готову (діючу) систему для виконання конкретної мети.

**Проєктування** - процес створення прототипу, прообразу майбутнього об'єкта, стану та способів його виготовлення. У проєктуванні застосовують системний підхід, який полягає у встановлені структури системи, типу зв'язків, визначенні атрибутів, аналізуванні впливів зовнішнього середовища.

**Прототип** - прообраз, зразок, оригінал.

**Технічна пропозиція** — проєктна конструкторська документація, яка містить технічне і техніко-економічне обґрунтування доцільності розроблення виробу на підставі аналізу технічного завдання та опрацювання можливих варіантів конструкції виробу.

**UI** – User Interface

**HID** – Human interface device

**CUI** – Combined User Interface

**API** – Application Programming Interface

**ОСРЧ** – Операційна система реального часу

**GUI** – Graphical User Interface

# Список літератури

## Основна:

1. Логунова О.С. Людино-машинна взаємодія: теорія та практика / О.С. Логунова, І.М. Ячиков, Е.А. Ільїна. – Київ: Фенікс, 2018. – 285 с.
2. Мандел Т. Розробка користувальницького інтерфейса / Т. Мандел. – Л.: Прес, 2021. – 416 с.
3. Навчальний посібник з дисципліни “Людино-машинний інтерфейс” для студентів напряму підготовки “Програмна інженерія”// Укладач доц. Уткіна Г.А.. -КЕІ ДВНЗ “КНЕУ імені Вадима Гетьмана”. – 2019. – 162с.
4. Р.П. Шевчук // Опорний конспект лекцій з дисципліни „Сучасні інструментальні засоби розробки користувацького інтерфейсу”, для студентів за спеціальностями: "Програмне забезпечення систем", - «Програмне забезпечення систем» — Тернопіль, 2018. — 103 с.
5. Людино-машинна взаємодія в системах штучного інтелекту : навч. посіб. / Н. І. Бойко [та ін.] ; Нац. ун-т "Львів. політехніка". - Львів : Вид-во Тараса Сороки, 2018. - 247 с. : рис. - Бібліогр.: с. 238-242. - ISBN 978-617-7593-13-2

## Додаткова:

1. Кушков В.М. Людино-машинні інтерфейси [Електронний ресурс]: Конспект лекцій для студ. напряму «Автоматизація та комп’ютерно-інтегровані технології» ден. та заоч. форм навч. – К.: НУХТ. 2022. - 100 с.
2. Мандел Т. Розробка користувацького інтерфейсу: Пер. з англ. -М.: ДМК, 2021, -431 с.
3. Уотсон К. Visual C# 2008: базовый курс / К. Уотсон, К. Нейгел, Я.Х. Педерсен, Дж.Д. Рид, М. Скиннер, Э. Уайт. – K.: Профпрес, 2019. – 1216 с.
4. Lukman Adam Obomeghie, Bayo Adedeji, Sylvester Aketaumhe Unified Modeling Language Practical Guide, 2022. 59p.
5. Suriya Sundaramoorthy UML Diagramming: A Case Study Approach 1st Edition, 2022. 416p.
6. ДСТУ 3899-2013 Дизайн і ергономіка. Терміни та визначення основних понять.
7. ДСТУ 7299:2013 Дизайн і ергономіка. Робоче місце оператора. Взаємне розташування елементів робочого місця. Загальні вимоги ергономіки
8. ДСТУ ISO 11064-4:2009 Проектування центрів керування ергономічне. Частина 4. Компонування та розміри автоматизованих робочих місць (ISO 11064-4:2000, IDT)

## Інформаційні ресурси:

1. Бібліотека офіційної технічної документації для розробників під ОС Microsoft Windows – www.msdn.com
2. Usability professional’s assotiations <http://www.usability> professionals.org/
3. http://developer.android.com
4. HTML Book [Електронний ресурс]. – Режим доступу: http://htmlbook.ua/
5. Підручник HTML [Електронний ресурс]. – Режим доступу: http://ru.html.net/tutorials/html/
6. HTML справочник [Електронний ресурс]. – Режим доступу: http://html.manual.ua/
7. QtQml 5.0: Qt QML [Електронний ресурс]. – Режим доступу: http://qt-project.org/doc/qt-5.0/qtqml/qtqml-index.html

Методичне видання

(українською мовою)

Чопорова Оксана Володимирівна

Чопоров Сергій Вікторович

**ЛЮДИНО-МАШИННИЙ ІНТЕРФЕЙС**

Методичні вказівки до лабораторних робіт

для студентів освітньо-кваліфікаційного рівня «бакалавр»

напряму підготовки «Програмна інженерія»

Рецензент *С. М. Гребенюк*

Відповідальний за випуск *С. В. Чопоров*

Коректор *О. В. Чопорова*

1. Rapid Application Development [↑](#footnote-ref-1)
2. Qt Meta-Object Language [↑](#footnote-ref-2)
3. JavaScript (JS) – динамічна, об'єктно-орієнтована мова програмування сценаріїв [↑](#footnote-ref-3)
4. http://www.w3.org/TR/SVG/types.html#ColorKeywords [↑](#footnote-ref-4)
5. Rich Text [↑](#footnote-ref-5)