### Описание проекта "Клиент-Серверная Игра Memory"

#### Введение

Проект представляет собой клиент-серверную игру "Memory" или "Найди пару", в которой два клиента подключаются к серверу, чтобы играть в игру, состоящую из 10 карточек пяти разных цветов. Цель игры - найти больше пар карточек, чем соперник. Игра реализована на языке C++ с использованием MFC (Microsoft Foundation Classes) и Windows Sockets (Winsock) для сетевого взаимодействия. Клиент-серверная архитектура позволяет двум игрокам играть в реальном времени, обмениваясь данными через сервер.

#### Компоненты проекта

1. \*\*Сервер (Server)\*\*

- Принимает подключения от клиентов.

- Обрабатывает игровые действия клиентов.

- Отправляет обновления состояния игры клиентам.

- Управляет логикой игры, включая проверку пар карточек и определение победителя.

2. \*\*Клиент (Client)\*\*

- Подключается к серверу.

- Отправляет действия игрока на сервер (например, выбор карточки).

- Получает и отображает обновления состояния игры от сервера.

- Обеспечивает графический интерфейс для взаимодействия игрока с игрой.

#### Основные компоненты интерфейса клиента

- \*\*Кнопка "Connect"\*\* - Подключение к серверу.

- \*\*Кнопка "Send"\*\* - Отправка игрового действия на сервер.

- \*\*Кнопка "Disconnect"\*\* - Отключение от сервера.

- \*\*Поле "Log"\*\* - Отображение сообщений и статуса игры.

- \*\*10 кнопок-карточек\*\* - Представляют собой карточки, которые игроки могут выбирать.

#### Описание работы

1. \*\*Подключение\*\*

- Клиент подключается к серверу, нажав кнопку "Connect".

- Сервер принимает соединение и уведомляет другого клиента о новом подключении.

2. \*\*Игра\*\*

- Игроки по очереди выбирают карточки.

- Клиент отправляет серверу сообщение о выбранной карточке.

- Сервер проверяет, является ли карточка частью пары.

- Если карточка составляет пару с ранее открытой, сервер уведомляет оба клиента и обновляет состояние игры.

- Если нет, сервер отправляет клиенту сообщение о том, что пара не найдена, и игра продолжается.

3. \*\*Отключение\*\*

- Игрок может отключиться от сервера, нажав кнопку "Disconnect".

- Сервер обрабатывает отключение и уведомляет другого игрока.

#### Основные классы и методы

\*\*Серверная часть\*\*

- `CServerDlg`: Основной диалоговый класс сервера, который управляет сетевыми соединениями и игровой логикой.

- `OnBnClickedStartButton()`: Запускает сервер и начинает прослушивание входящих соединений.

- `HandleClientConnection(SOCKET clientSocket)`: Обрабатывает подключение клиента в отдельном потоке.

- `ProcessClientMessage(SOCKET clientSocket, const std::string& message)`: Обрабатывает сообщения от клиентов.

\*\*Клиентская часть\*\*

- `CClientDlg`: Основной диалоговый класс клиента, который управляет подключением к серверу и интерфейсом игры.

- `OnBnClickedConnectButton()`: Подключается к серверу.

- `OnBnClickedSendButton()`: Отправляет сообщение серверу.

- `OnBnClickedDisconnectButton()`: Отключается от сервера.

- `OnBnClickedCardButton(UINT nID)`: Обрабатывает нажатие на карточки.

- `UpdateCardState(int cardIndex, bool revealed, int player)`: Обновляет состояние карточек в интерфейсе.

#### Используемые технологии

- \*\*MFC (Microsoft Foundation Classes)\*\*: Для создания графического интерфейса пользователя.

- \*\*Winsock\*\*: Для реализации сетевого взаимодействия.

- \*\*Многопоточность\*\*: Для обработки клиентских соединений на сервере и чтения данных в клиенте.

#### Заключение

Этот проект демонстрирует базовые принципы клиент-серверного взаимодействия с использованием MFC и Winsock, а также основы многопоточного программирования. Игра "Memory" предоставляет интересный способ изучения сетевого программирования и работы с графическим интерфейсом в Windows.