1 Реализация обмена сообщениями в среде QT

1.1 Сигналы и слоты

Сигналы и слоты используются для коммуникации между объектами в Qt:

— **сигнал**: вырабатывается когда происходит определенное событие;

— **слот**: функция, которая вызывается в ответ на определенный сигнал.

На рис. **1** показан механизм работы сигналов и слотов. Здесь

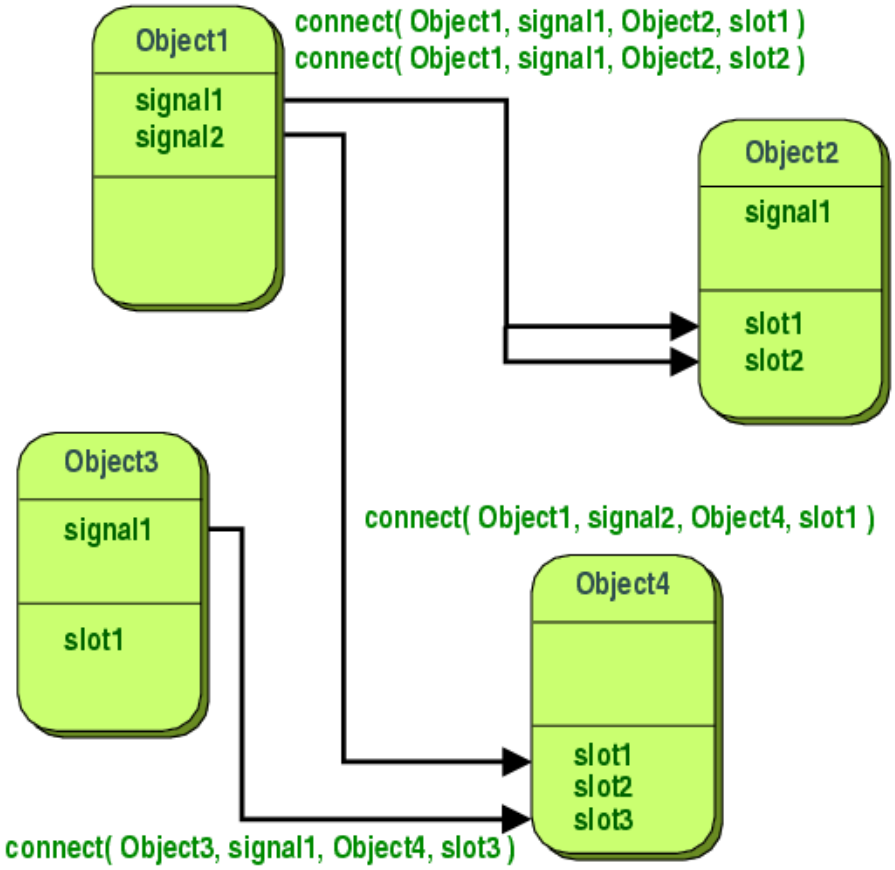


Рис. 1. Схема действия сигналов и слотов

— $Object_1$, $Object_2$ – объекты, имеющие в себе сигналы и слоты;

— *connect (args)* – функция, соединяющая сигналы и слоты в первом и втором объекте

1.2 Концепция программы

Концепция программы – реализовать пересылку сообщений между двумя объектами одного класса.

Первый объект будет отправлять второму сигнал *ping*, а второй первому – сигнал *pong*.

1.3 Реализация кода на языке Qt

1.3.1 Структура класса

В листинге **1** приведён отрывок программы *ping - pong*.

```
1  public  slots:
2    void processPing(const QString& msg);
3    void processPong(const QString& msg);
4    void sendPing();
5
6  signals:
7    void ping(QString = "", int = 0);
8    void pong(QString);
9    void msgPinged(QString);
10   void msgPonged(QString);
```

Листинг 1. Сигналы и слоты класса

Здесь содержатся 4 сигнала и 3 слота:

— **Сигналы:**

1. *ping* – сигнал, отправляемый слотом *sendPing*;
2. *pong* – сигнал, отправляемый слотом *processPing* (см. листинг **2**);
3. *msgPinged* – сигнал графического интерфейса, отправляемый слотом *sendPing* (см. листинг **4**);

4. *msgPonged* – сигнал графического интерфейса, отправляемый слотом *processPong* (см. листинг **3**).

— **Слоты:**

1. *processPing* – получает сигнал *ping* и отсылает сигнал *pong*:

```
1  void MainClass::processPing(const QString& msg)
2  {
3    emit pong(msg);
4  }
```

Листинг 2. Реализация слота processPing

2. *processPong* – получает сигнал *pong* и приступает к его обработке: записывает в строку номер сообщения и время его получения:

```
1  void MainClass::processPong(const QString& msg)
2  {
3    // ? Display pong process
4    QString m;
5    m.append("Message ponged # ");
6    m.append(QString::number(a));
7    m.append(". Time: ");
8    m.append(QTime::currentTime().toString("hh:mm:ss
       "));
9    emit msgPonged(m);
10
11 }
```

Листинг 3. Реализация слота processPong

3. *sendPing* – слот, соединённый с таймером, отправляет сигналы *pong* и *msgPinged*:

```
1  void MainClass::sendPing()
2  {
3    ++a;
```

```
4         emit ping(QString::number(a));
5         QString m;
6         m.append("Message pinged # ");
7         m.append(QString::number(a));
8         m.append(". Time: ");
9         m.append(QTime::currentTime().toString("hh:mm:ss
              "));
10        emit msgPinged(m);
11    }
```

Листинг 4. Реализация слота sendPing

1.3.2 Связь сигналов и слотов

Для работы с сигналами и слотами, они должны быть соединены между собой. Соединение сигналов и слотов класса между собой приведены в листинге **5**:

```
1  QObject::connect(&timer, SIGNAL(timeout()),
2      &test1, SLOT(sendPing()));
3
4  QObject::connect(&test1, SIGNAL(ping(QString)),
5      &test2, SLOT(processPing(QString)));
6
7  QObject::connect(&test2, SIGNAL(pong(QString)),
8      &test1, SLOT(processPong(QString)));
9
10 QObject::connect(&test1, SIGNAL(msgPinged(QString)),
11     ui->plainTextEdit, SLOT(appendPlainText(QString)));
12
13 QObject::connect(&test1, SIGNAL(msgPonged(QString)),
14     ui->plainTextEdit_2, SLOT(appendPlainText(QString)));
```

Листинг 5. Соединение сигналов и слотов

Принцип работы программы следующий (см. рис. **2**):

1. Вызывается таймер;
2. Таймер вызывает слот *sendPing*;
3. Слот *sendPing* отправляет сигнал *ping*, а также отправляет сигнал графическому интерфейсу *msgPinged*;
4. Сигнал *ping* вызывает слот *processPing*;
5. Слот *processPing* отправляет сигнал *pong*;
6. Сигнал *pong* вызывает слот *processPong*;
7. Слот *processPong* отправляет сигнал графическому интерфейсу *msgPonged*

1.3.3 Графический интерфейс и окно вывода

На рис. **3** показан интерфейс программы до начала работы и после её окончания.

В левом поле пишутся сообщения класса *ping*, в правом – *pong*. Нижнее поле предназначено для задания интервалов отправки сообщений в милисекундах.

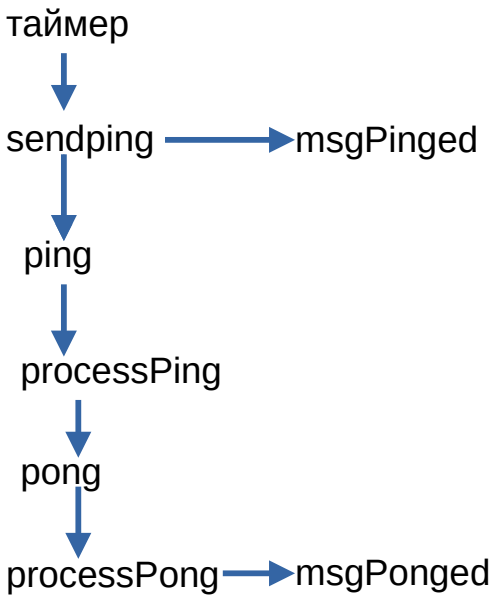
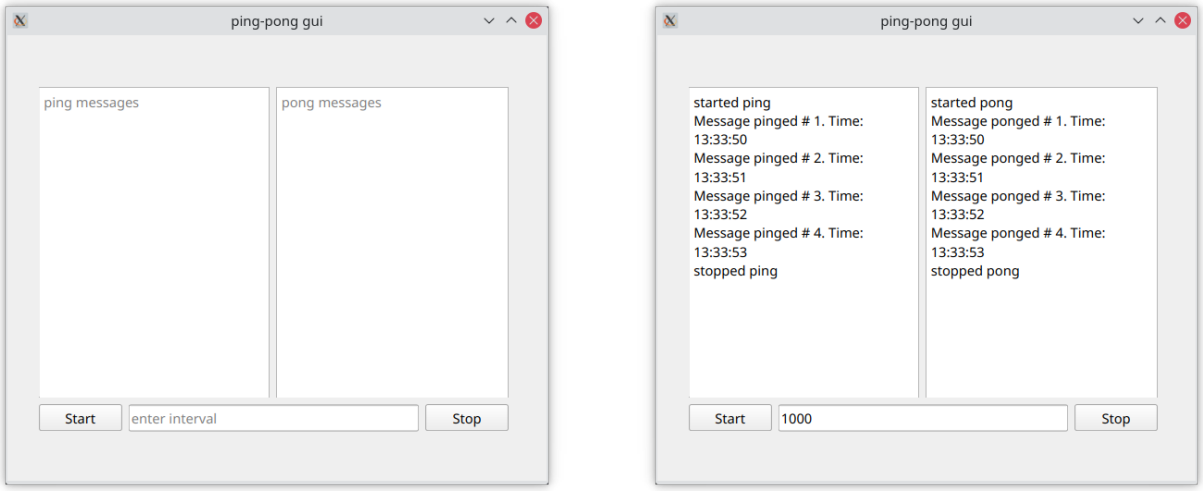


Рис. 2. Структурная схема программы



а) Интерфейс при открытии

б) Интерфейс после исполнения

Рис. 3. Графический интерфейс программы