Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего профессионального образования
«Севастопольский государственный университет»

ИССЛЕДОВАНИЕ СПОСОБОВ РЕАЛИЗАЦИИ СЕТЕВЫХ ЗАПРОСОВ В QT-ПРИЛОЖЕНИЯХ

Методические указания

к лабораторной работе по дисциплине

«Кроссплатформенное программирование»

для студентов, обучающихся по направлению

09.03.02 “Информационные системы и технологии”

очной и заочной форм обучения

Севастополь
2018

УДК 004.415.2
Исследование способов реализации сетевых запросов в Qt-приложениях. Методические указания/Сост. Строганов В.А. – Севастополь: Изд-во СевГУ, 2018.–16 с.

Методические указания предназначены для оказания помощи студентам при выполнении лабораторных работ по дисциплине «Кроссплатформенное программирование».

Методические указания составлены в соответствии с требованиями программы дисциплины «Кроссплатформенное программирование» для студентов направления 09.03.02 и утверждены на заседании кафедры «Информационные системы»,
протокол № от « »_____ 2018 г.

Содержание

1. ЦЕЛЬ РАБОТЫ

Исследование способов работы Qt-приложения с HTTP GET/POST запросами. Приобретение навыков разработки простейших сетевых приложений в Qt.

2. ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ ПОЛОЖЕНИЯ

Qt Framework ожидаемо содержит классы для написания клиентов и серверов TCP/IP. Данные классы содержаться в модуле QtNetwork.

Модуль работы с сетью предоставляет классы, которые помогают сделать сетевое программирование проще и переносимей. Он предлагает классы, такие как QHttp и QFtp, которые реализуют специфические протоколы уровня приложения (application-level), низкоуровневые классы, такие как QTcpSocket, QTcpServer и QUdpSocket, которые представляют низкоуровневые сетевые концепции, и высокоуровневые классы, такие как QNetworkRequest, QNetworkReply и QNetworkAccessManager для представления сетевых операций с использованием распространенных протоколов.

Модуль QtNetwork является частью Выпуска Qt Full Framework и Версий Open Source Qt.

В данной работе будут рассмотрена работа Qt фреймворка с HTTP GET/POST запросами с использованием класса QNetworkAccessManager.

HTTP (англ. HyperText Transfer Protocol — «протокол передачи гипертекста») — протокол прикладного уровня передачи данных (изначально — в виде гипертекстовых документов). Основой HTTP является технология «клиент-сервер», то есть предполагается существование потребителей (клиентов), которые инициируют соединение и посылают запрос, и поставщиков (серверов), которые ожидают соединения для получения запроса, производят необходимые действия и возвращают обратно сообщение с результатом.

HTTP в настоящее время повсеместно используется во Всемирной паутине для получения информации с веб-сайтов.

Метод HTTP (англ. HTTP Method) — последовательность из любых символов, кроме управляющих и разделителей, указывающая на основную операцию над ресурсом. Обычно метод представляет собой короткое английское слово, записанное заглавными буквами. Обратите внимание, что название метода чувствительно к регистру.

Каждый сервер обязан поддерживать как минимум методы GET и HEAD. Если сервер не распознал указанный клиентом метод, то он должен вернуть статус 501 (Not Implemented). Если серверу метод известен, но он неприменим к конкретному ресурсу, то возвращается сообщение с кодом 405 (Method Not Allowed). В обоих случаях серверу следует включить в сообщение ответа

заголовок Allow со списком поддерживаемых методов.

Кроме методов GET и HEAD, часто применяется метод POST.

GET

Используется для запроса содержимого указанного ресурса. С помощью метода GET можно также начать какой-либо процесс. В этом случае в тело ответного сообщения следует включить информацию о ходе выполнения процесса.

Клиент может передавать параметры выполнения запроса в URI целевого ресурса после символа «?»:

GET /path/resource?param1=value1&param2=value2 HTTP/1.1

Согласно стандарту HTTP, запросы типа GET считаются идемпотентными.

Идемпотентность — термин, означающий свойство математического объекта, которое проявляется в том, что повторное действие над объектом не изменяет его.

Кроме обычного метода GET, различают ещё ***условный GET*** и ***частичный GET***. Условные запросы GET содержат заголовки If-Modified-Since, If-Match, If-Range и подобные. Частичные GET содержат в запросе Range. Порядок выполнения подобных запросов определён стандартами отдельно.

POST

Применяется для передачи пользовательских данных заданному ресурсу. Например, в блогах посетители обычно могут вводить свои комментарии к записям в HTML-форму, после чего они передаются серверу методом POST и он помещает их на страницу. При этом передаваемые данные (в примере с блогами — текст комментария) включаются в тело запроса. Аналогично с помощью метода POST обычно загружаются файлы на сервер.

В отличие от метода GET, метод POST не считается идемпотентным, то есть многократное повторение одних и тех же запросов POST может возвращать разные результаты (например, после каждой отправки комментария будет появляться одна копия этого комментария).

При результате выполнения 200 (Ok) в тело ответа следует включить сообщение об итоге выполнения запроса. Если был создан ресурс, то серверу следует вернуть ответ 201 (Created) с указанием URI нового ресурса в заголовке Location.

Сообщение ответа сервера на выполнение метода POST не кэшируется.

Класс QNetworkAccessManager позволяет приложению отправлять

сетевые запросы и получать ответы.

API сетевого доступа создано вокруг объекта QNetworkAccessManager, который содержит общую конфигурацию и настройки для посылаемых запросов. Он содержит прокси и кэш, а также сигналы, связанные с ними, и сигналы ответов, которые могут быть использованы для контроля за прогрессом сетевой операции. Одного объекта QNetworkAccessManager будет достаточно для всего приложения Qt.

После создания объекта QNetworkAccessManager, приложение может посылать запросы по сети. Поставляется группа стандартных функций, которые принимают запрос и необязательные данные, и каждая возвращает объект QNetworkReply. Возвращаемый объект используется для получения любых данных, возвращаемых в ответ на соответствующий запрос.

Документация QNetworkAccessManager:
http://www.doc.crossplatform.ru/qt/4.7.x/qnetworkaccessmanager.html

Класс QNetworkRequest представляет собой тело запроса в объектной форме, который будет послан классом QNetworkAccessManager. QNetworkRequset является частью Network Access API и содержит в себе данные, необходимые для отправки сетевого запроса.

Документация QNetworkRequest:
http://www.doc.crossplatform.ru/qt/4.7.x/qnetworkreply.html

Класс QNetworkReply содержит данные и заголовки для запроса, отправленного с помощью QNetworkAccessManager.

Класс QNetworkReply содержит данные и метаданные, связанные с размещённым с помощью QNetworkAccessManager запросом. Как QNetworkRequest, он содержит URL и заголовки (как в обработанном, так и исходном виде), некоторую информацию о состоянии ответа и содержимое самого ответа.

Документация QNetworkReply:
http://www.doc.crossplatform.ru/qt/4.7.x/qnetworkreply.html

Для работы Qt приложения с классами модуля QNetwork необходимо добавить в файл проекта .pro следующую строку:

QT += network

3. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

3.1. Изучить основные возможности модуля QtNetwork для реализации HTTP-запросов (выполняется в ходе самостоятельной подготовки к лабораторной работе).

3.2. Создать Qt GUI приложение

3.3. В дизайнере создать форму со следующим расположением элементов

управления:

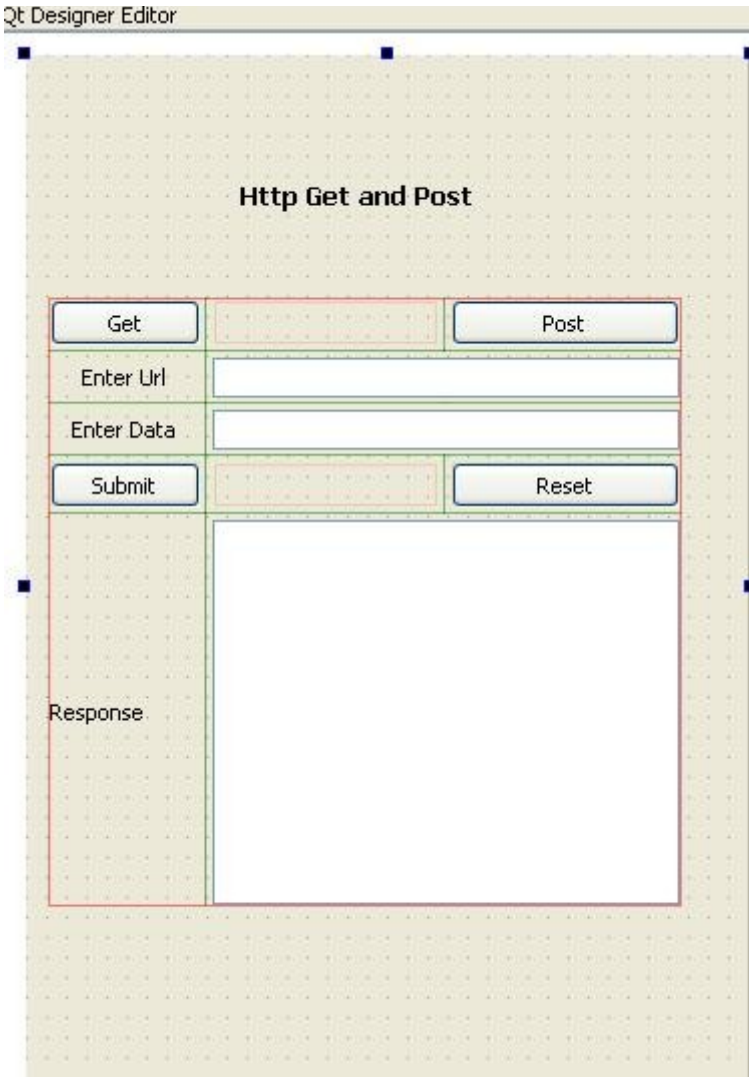

Рисунок 3.1 — Расположение виджетов в окне приложения

3.3. Реализовать следующую логику приложения (см. Приложение А): Все элементы, кроме кнопок выбора типа запроса изначально скрыты.

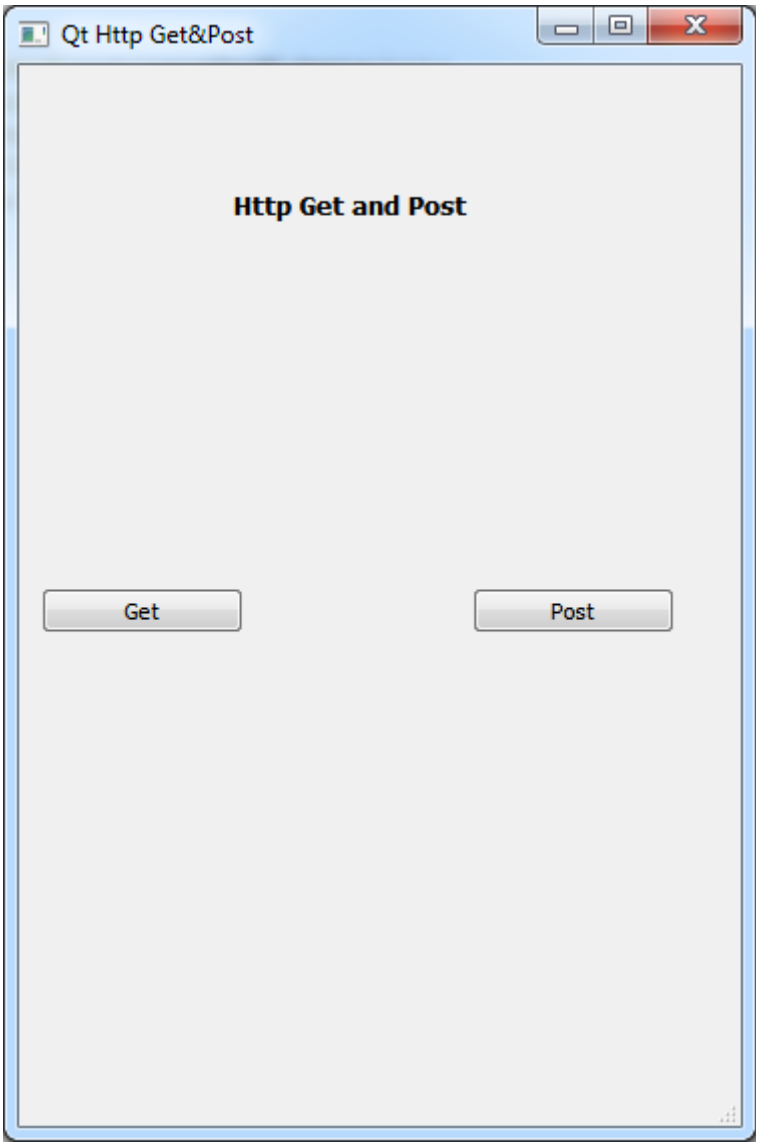

Рисунок 3.2 — Исходный вид окна приложения

После выбора метода отобразить скрытые элементы. При этому кнопки выбора метода – скрыть. QLabel с текстом “Enter Data:” и соответствующее поле ввода должно отображаться только для POST запроса.

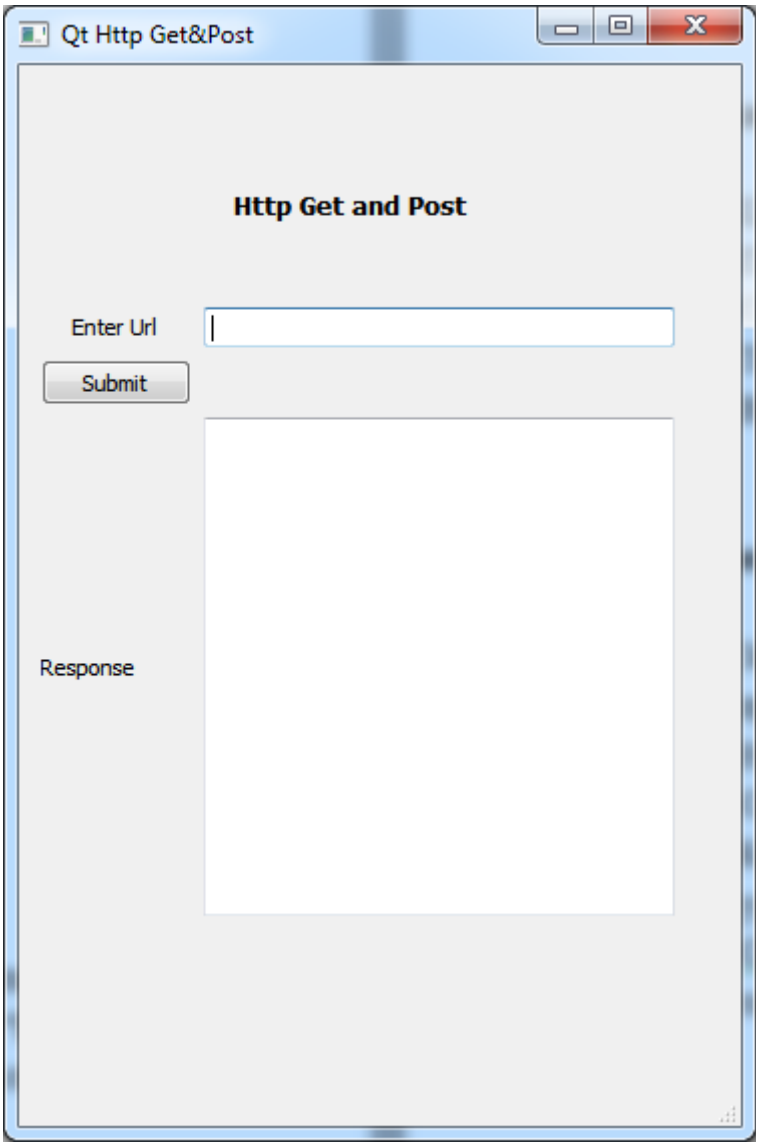

Рисунок 3.3 — Окно приложения после нажатия на кнопку «Get»

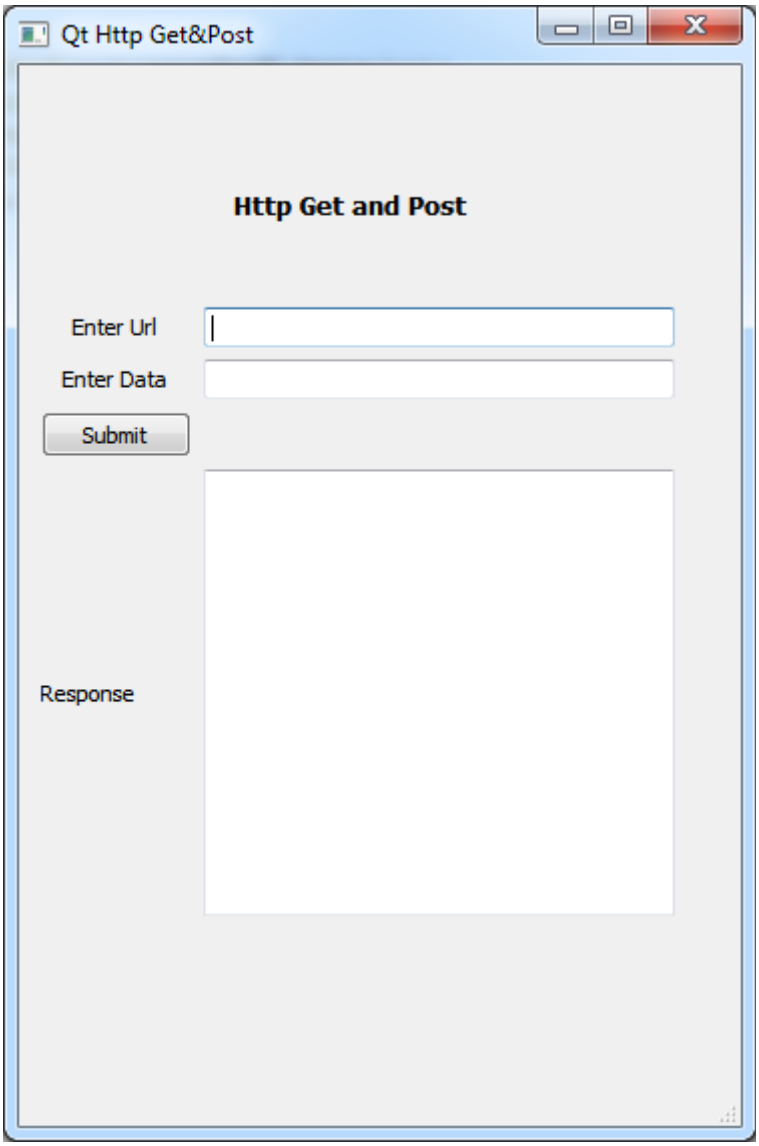

Рисунок 3.4 — Окно приложения после нажатия на кнопку «Post»

По нажатию кнопки Submit реализовать отправку соответствующего запроса и вывод результата в поле Response. Для POST запроса использовать данные из поля Enter Data.

Для тестирования работы HTTP GET запроса используйте следующий URL: http://api.openweathermap.org/data/2.5/weather?q=Sevastopol,ua

Для тестирования работы HTTP POST:

Url: http://api.forismatic.com/api/1.0/

Данные: method=getQuote&key=457653&format=xml&lang=ru

3.4. Исследовать работу программы при отправке GET и POST запросов, проанализировать формат получаемых HTTP-ответов.

4. СОДЕРЖАНИЕ ОТЧЕТА

4.1. Цель работы.

4.2. Постановка задачи.

4.3. Описание алгоритма работы приложения.

4.4. Результаты работы приложения после выполнения GET и POST запросов.

4.5. Текст программы.

4.6. Выводы.

5. КОНТРОЛЬНЫЕ ВОПРОСЫ

5.1. Какой модуль Qt Framework предназначен для работы с TCP/IP протоколом?

5.2. Какие основные классы используются для работы с сетевыми операциями?

5.3. Расскажите о назначении класса QNetworkAccessManager.

5.4. Каким образом осуществить отправку и получение результата GET запроса, используя QNetworkAccessManager?

5.5. Каким образом осуществить отправку и получение результата POST запроса, используя QNetworkAccessManager?

5.6. Какую роль играет механизм сигналов/слотов при отправке и обработке результата запроса?

5.7. Необходимо ли для осуществления отдельного запроса использовать отдельный экзмепляр QNetworkAccessManager? Если нет, каким образом вы бы реализовали работу с QNetworkAccessManager в рамках крупного проекта? Какие шаблоны проектирования позволили бы упростить эту задачу?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Ж. Бланшет. Qt 3: программирование GUI на C++ / Бланшет Ж., Саммерфилд М. — М. : КУДИЦ — ОБРАЗ, 2005. - 448 с.

2. Е.Р. Алексеев. Программирование на языке C++ в среде Qt Creator /Е. Р. Алексеев, Г. Г. Злобин, Д. А. Костюк, О. В. Чеснокова, А. С. Чмыхало.— М.:Альт Линукс, 2015. — 448 с.

3. Буч, Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++/Г. Буч. — М. : БИНОМ ; СПб. : Невский диалект, 2001. - 560 с.

4. Шилдт, Г. C++: базовый курс, 3-е издание /Г. Шилдт. — М.: «Вильямс», 2012. — 624 с.

5. Шилдт, Г. Полный справочник по C++, 4-е издание /Г. Шилдт. — М.: «Вильямс», 2011. — 800 с.

ПРИЛОЖЕНИЕ — Пример программы

Заголовочный файл

```
#ifndef MAINWINDOW_H
```

```
#define MAINWINDOW_H

#include <QMainWindow>
#include <QNetworkAccessManager>
#include <QNetworkReply>
#include <QMessageBox>
#include<QByteArray>


namespace Ui {
    class MainWindow;
}

class MainWindow : public QMainWindow {
    Q_OBJECT
public:
    MainWindow(QWidget *parent = 0);
    ~MainWindow();

protected:
    void changeEvent(QEvent *e);

private slots:
    void finished(QNetworkReply *reply);
    void DoHttpGet();
    void activateGetWidgets();
    void HideWidgets();
    void clearWidgets();
    void activatePostWidgets();

private:
    Ui::MainWindow *ui;

    QNetworkAccessManager *nam;
};

#endif // MAINWINDOW_H
```

Файл mainwindow.cpp

```
#include "mainwindow.h"
#include "ui_mainwindow.h"

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    nam = new QNetworkAccessManager(this);
    HideWidgets();
    connect(ui->getButton,SIGNAL(clicked()),this,SLOT(activateGetWidgets()));
    connect(ui->submitButton,SIGNAL(clicked()),this,SLOT(DoHttpGet()));
    connect(ui->resetButton,SIGNAL(clicked()),this,SLOT(clearWidgets()));
    connect(nam,SIGNAL(finished(QNetworkReply*)),this,SLOT(finished(QNetworkReply*)));
    connect(ui->postButton,SIGNAL(clicked()),this,SLOT(activatePostWidgets()));
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::changeEvent(QEvent *e)
{
    QMainWindow::changeEvent(e);
```

```
    switch (e->type()) {
    case QEvent::LanguageChange:
        ui->retranslateUi(this);
        break;
    default:
        break;
    }
}

void MainWindow::activateGetWidgets()
        {
      ui->urlLabel->setHidden(false);
      ui->urlLine->setHidden(false);
      ui->submitButton->setHidden(false);
      ui->textBrowser->setHidden(false);
      ui->responseTitleLabel->setHidden(false);
      ui->getButton->setHidden(true);
      ui->postButton->setHidden(true);

        }

void MainWindow::activatePostWidgets()
        {
        ui->dataLabel->setHidden(false);
        ui->dataLine->setHidden(false);
        activateGetWidgets();

        }



void MainWindow::finished(QNetworkReply *reply)
{
    if(reply->error() == QNetworkReply::NoError)
    {
        ui->textBrowser->setText(reply->readAll());

    }
    else
    {
        ui->textBrowser->setText(reply->errorString());
    }
}

void MainWindow::DoHttpGet()
        {
                ui->resetButton->setHidden(false);
        QString url = ui->urlLine->text();
        QString data = ui->dataLine->text();
        QByteArray postData;
        postData.append(data.toAscii());
        if(postData.isEmpty() == true)
                {
                nam->get(QNetworkRequest(QUrl(url)));
                }
        else
                {
                nam->post(QNetworkRequest(QUrl(url)),postData);
                }

        }

void MainWindow::HideWidgets()
        {
    ui->urlLabel->setHidden(true);
    ui->urlLine->setHidden(true);
    ui->dataLabel->setHidden(true);
```

```
    ui->dataLine->setHidden(true);
    ui->submitButton->setHidden(true);
    ui->responseTitleLabel->setHidden(true);
    ui->textBrowser->setHidden(true);
    ui->resetButton->setHidden(true);

        }

void MainWindow::clearWidgets()
        {
    ui->urlLabel->setHidden(true);
    ui->urlLine->setHidden(true);
    ui->dataLabel->setHidden(true);
    ui->dataLine->setHidden(true);
    ui->submitButton->setHidden(true);
    ui->responseTitleLabel->setHidden(true);
    ui->textBrowser->setHidden(true);
    ui->resetButton->setHidden(true);
    ui->urlLine->clear();
    ui->textBrowser->clear();
    ui->dataLine->clear();
    ui->getButton->setHidden(false);
    ui->postButton->setHidden(false);

        }
```

Файл формы mainwindow.ui

```
<?xml version="1.0" encoding="UTF-8"?>
<ui version="4.0">
 <class>MainWindow</class>
 <widget class="QMainWindow" name="MainWindow">
  <property name="geometry">
   <rect>
    <x>0</x>
    <y>0</y>
    <width>360</width>
    <height>530</height>
   </rect>
  </property>
  <property name="windowTitle">
   <string>MainWindow</string>
  </property>
  <widget class="QWidget" name="centralWidget">
   <widget class="QWidget" name="layoutWidget">
    <property name="geometry">
     <rect>
      <x>11</x>
      <y>121</y>
      <width>316</width>
      <height>304</height>
     </rect>
    </property>
    <layout class="QGridLayout" name="gridLayout">
     <item row="0" column="0">
      <widget class="QPushButton" name="getButton">
       <property name="text">
        <string>Get</string>
       </property>
      </widget>
     </item>
     <item row="0" column="2">
      <widget class="QPushButton" name="postButton">
       <property name="text">
```

```
        <string>Post</string>
       </property>
      </widget>
     </item>
     <item row="1" column="0">
      <widget class="QLabel" name="urlLabel">
       <property name="text">
        <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot;
content=&quot;1&quot; /&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8.25pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p align=&quot;center&quot; style=&quot; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;Enter Url&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
       </property>
      </widget>
     </item>
     <item row="1" column="1" colspan="2">
      <widget class="QLineEdit" name="urlLine"/>
     </item>
     <item row="2" column="0">
      <widget class="QLabel" name="dataLabel">
       <property name="text">
        <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot;
content=&quot;1&quot; /&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8.25pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p align=&quot;center&quot; style=&quot; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;span style=&quot; font-size:8pt;&quot;&gt;Enter
Data&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
       </property>
      </widget>
     </item>
     <item row="2" column="1" colspan="2">
      <widget class="QLineEdit" name="dataLine"/>
     </item>
     <item row="3" column="0">
      <widget class="QPushButton" name="submitButton">
       <property name="text">
        <string>Submit</string>
       </property>
      </widget>
     </item>
     <item row="3" column="2">
      <widget class="QPushButton" name="resetButton">
       <property name="text">
        <string>Reset</string>
       </property>
      </widget>
     </item>
     <item row="4" column="0">
      <widget class="QLabel" name="responseTitleLabel">
       <property name="text">
        <string>Response</string>
       </property>
      </widget>
```

```
     </item>
     <item row="4" column="1" colspan="2">
      <widget class="QTextBrowser" name="textBrowser"/>
     </item>
    </layout>
   </widget>
   <widget class="QLabel" name="label">
    <property name="geometry">
     <rect>
      <x>10</x>
      <y>50</y>
      <width>311</width>
      <height>41</height>
     </rect>
    </property>
    <property name="text">
     <string>&lt;!DOCTYPE HTML PUBLIC &quot;-//W3C//DTD HTML 4.0//EN&quot;
&quot;http://www.w3.org/TR/REC-html40/strict.dtd&quot;&gt;
&lt;html&gt;&lt;head&gt;&lt;meta name=&quot;qrichtext&quot;
content=&quot;1&quot; /&gt;&lt;style type=&quot;text/css&quot;&gt;
p, li { white-space: pre-wrap; }
&lt;/style&gt;&lt;/head&gt;&lt;body style=&quot; font-family:'MS Shell Dlg 2';
font-size:8.25pt; font-weight:400; font-style:normal;&quot;&gt;
&lt;p align=&quot;center&quot; style=&quot; margin-top:0px; margin-bottom:0px;
margin-left:0px; margin-right:0px; -qt-block-indent:0; text-
indent:0px;&quot;&gt;&lt;span style=&quot; font-size:10pt; font-
weight:600;&quot;&gt;Http Get and
Post&lt;/span&gt;&lt;/p&gt;&lt;/body&gt;&lt;/html&gt;</string>
    </property>
   </widget>
  </widget>
  <widget class="QStatusBar" name="statusBar"/>
 </widget>
 <layoutdefault spacing="6" margin="11"/>
 <resources/>
 <connections/>
</ui>
```

Заказ № _____ от «_____» _____ 2018 г. Тираж экз.