Лабораторная работа №3 Браузер с возможностью работы с HTTP и FTP

Лабораторная работа №3 выполняется после изучения материала, посвященного описанию принципов использования C#, Java, Python, C++ и библиотек .Net, QT, WxWidgets, wxPython, javafx для управления соединениями с сервером по протоколам HTTP и FTP. Большинство данных библиотек уже имеет в своем составе готовый компонент подобный WebBrowser для работы как с FTP так и http, предлагая стандартные возможности навигации по страницам гипертекстовых документов и каталогизированной структуре ftp серверов.

Цель работы:

написать GUI приложение для ОС Windows или ОС Linux, представляющее собой простой Web – браузер и FTP – клиент, используя стандартные компоненты. Сложный вариант включает создание приложения использующего сокеты и непосредственно команды HTTP для получения контента веб-сервера и команды FTP для создания клиента. Простой вариант предполагает использование готового компонента (например, WebBrowser). Но при выполнении простого варианта для получения дополнительных баллов необходимо сделать задание по получению страницы HTTP обычным запросом через соккет, парсинг этой страницы и скачивание контента по какой-либо ссылке присутствующей на данной странице (можно выбрать один сайт и сделать «жестко заданное» приложение). Для простоты можно вывести полученную страницу отдельно в готовом компоненте или браузере.

Так же сделать приложение FTP (FTPS, rfc4217, https://tools.ietf.org/html/rfc4217) клиент, которое выполнит хотя бы получение списка файлов с использованием команд протокола. Не путайте FTPS (SSL FTP) с FTP через SSH (туннелирование обычного FTP через соединение SSH, защита реализуется только для пассивного клиентского соединения из-за особенности работы FTP в активном режиме требующего подключения сервера к клиенту). Так же есть SFTP (SHH FTP, защищенное через SHH FTP соединение, SSH2, отдельный от FTP протокол, продолжение SSH), а также с SFTP (Simple FTP, урезанная версия FTP).

Используя Visual Studio и C# сделать Браузер и FTP клиент достаточно просто, необходимо создать приложение с формой на котором необходимо разместить объект WebBrowser и объекты для редактирования и ввода текстовой строки, а так же кнопки для навигации. Реализация данного размещения интуитивна и проста, так же как и создание обработчика события на нажатие кнопки - разметив кнопку на форме, можно в режиме конструктора нажать на неё, обработчик событий в виде заголовочных кодов метода обработчика будет создан в окне редактирования кода программы. В обработчике можно написать любой код, который вам необходим, например, там может быть вызов метода объекта WebBrowser.Navigate, с указанием в качестве параметра адреса перехода, который можно взять, например, в свойстве объекта TextEdit.Text. Объект WebBrowser имеет событие NewWindow, возникающее при попытке открыть новое окно при нажатии на ссылку требующую открытия нового окна (например, при переходе на ссылки в поисковике), его необходимо обработать, так чтобы открывался ваш браузер, а не браузер по умолчанию. Рекомендуется так же изучить свойства и методы и события. Например, AllowNavigation, Navigated и т.д. Примеры можно посмотреть на:

https://docs.microsoft.com/ru-ru/dotnet/api/system.windows.forms.webbrowser.navigated?view=netframework-4.7.2

Использование QT требует бОльших затрат на изучение данной технологии, но может дать неоценимый опыт при изучении концепции использования сигналов и слотов, при реализации возможностей обработки событий. Для создания приложений на QT рекомендуется использовать QTCreator и QT Designer. Для браузера можно использовать

QWebView и более современную версию Qt WebEngine. Так же можно использовать QAxWidget для использования ActiveX объекта Webbrowser, соответственно только для Windows.

GUI приложения на Java могут использовать swing, апплеты awt, более современная библиотека javafx имеет компонент WebView – минибраузер.

Очень простой пример на Python с использованием wxPython. Так же может понадобиться установка python-wxgtk-webview3.0 под Linux. Здесь реализована одна простая функция перехода по ссылке, при этом нет обработки события открытия нового окна, назад, вперед и многих других функций.

```
import wx
import wx.html2

class TestFrame(wx.Frame):
	go = []
	address = []
	browser = []
#обработчик события нажатия кнопки
	def onGoButton(self,event):
		print("ButtonClick")
		print(self.address.GetValue())
#загрузка страницы, адрес берется из строки ввода
		self.browser.LoadURL(self.address.GetValue())
		return

	def __init__(self, parent, title):
		wx.Frame.__init__(self, parent, id=-1, title=title)
		text = wx.StaticText(self, label=title)
# компонент бокссайзер позволяет управлять размещением объектов на форме
#(в данном случае вертикально)
		sizer = wx.BoxSizer(wx.VERTICAL)
#объект для создания минибраузера
		self.browser = wx.html2.WebView.New(self)
#объект для текстовой строки ввода
		self.address = wx.TextCtrl(self, value="http://")
#объект кнопка
		self.go = wx.Button(self, label="Go", id=wx.ID_OK)
#добавляем объекты для пропорционального размещения в соотношении 90, 5, 5
		sizer.Add(self.browser, proportion = 90, flag = wx.EXPAND, border = 10)
		sizer.Add(self.address, proportion = 5, flag = wx.EXPAND, border = 10)
		sizer.Add(self.go, proportion = 5, flag = wx.EXPAND, border = 10)
		self.SetSizer(sizer)
		self.SetSize((1000, 800))
#связываем обработчик события onGoButton и кнопку go
		self.Bind(wx.EVT_BUTTON,self.onGoButton,self.go)

app = wx.App()
frame = TestFrame(None, "Simple browser")
frame.Show()
app.MainLoop()
```

Варианты заданий.

1) Создать приложение Браузер с двумя окнами отображения страниц, новая открытая страница должна отображаться во втором окне, новая страница, открытие которой инициировано во втором окошке браузера, должна отображаться в первом.
2) Создать приложение браузер с возможностями навигации, кнопками вперед, назад.
3) Создать Браузер с тремя вкладками, возможностями навигации, вперед, назад.
4) Создать Браузер, в котором любой переход по ссылке приводит к открытию нового окна.
5) Создать браузер, в котором три объекта webbrowser при этом каждый первый переход по ссылке в первом окне открывается во втором окне, а каждый второй в третьем.
6) Два объекта веб браузера, сделать кнопку обмена местами окна справа и слева.
7) Переход по ссылке в окне браузера при создании нового окна приводит к отображению нового окна во втором объекте браузера.
8) Два объекта браузера в одном окне располагаются слева и справа, для каждой есть строка ввода адреса, а так же возможность навигации вперед и назад, но при этом они меняются местами. То есть при навигации назад и вперед, все изменения происходят в другом объекте.
9) Сделать приложение, в котором введенная строка для перехода на сайт пересылается через какое-то время из другого приложения по TCP соединению.
10) Сделать так, чтобы переход на новую ссылку вызывал отображение ссылки перехода в списке.