МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего профессионального образования  
«Национальный исследовательский университет  «МИЭТ»

Кафедра: ПКИМС

**Отчет по производственной   
 практике**

Выполнил:  
ст.гр. ЭКТ-43  
Мазурик Е.С.

Руководитель от кафедры:   
Гусев С. В.

Руководитель от организации:  
Вернер О. В.

Москва   
2016

Оглавление

[Об организации 3](#_Toc469586128)

[Основные направления деятельности компании «ЭЛВИС-ПЛЮС» 3](#_Toc469586129)

[Базовый доверенный модуль 4](#_Toc469586130)

[О продукте 4](#_Toc469586131)

[Как работает БДМ 7](#_Toc469586132)

[Изучение Diskpart и Dism (разметка диска и работа с образами) 8](#_Toc469586133)

[Основные понятия: 8](#_Toc469586134)

[Виды разметок диска 9](#_Toc469586135)

[Программы для управления диском 10](#_Toc469586136)

[Индивидуальное задание 12](#_Toc469586137)

[Выбор инструментария 12](#_Toc469586138)

[Теоретические сведения 12](#_Toc469586139)

[Графический интерфейс программы 14](#_Toc469586140)

[Структура программы 16](#_Toc469586141)

[Принцип работы программы 17](#_Toc469586142)

[Список источников 20](#_Toc469586143)

Об организации

Открытое акционерное общество «ЭЛВИС-ПЛЮС» — российская компания, системный интегратор информационной безопасности. На рисунке 1 представлен логотип компании.



Рисунок 1 – Логотип ОАО «ЭЛВИС ПЛЮС»

Компания была образована в 1991 году. За время нашего существования мы успели занять прочное положение на рынке системной интеграции в области телекоммуникационных, сетевых и информационных технологий, а также в области сетевой информационной безопасности. Нашими клиентами стали ЦБ РФ, ГК «АРКО», НК «ЛУКОЙЛ», ОАО «Русский алюминий», РАО «ЕЭС России», ОАО «Компания «Сухой», ОАО «Ростелеком» и т.д.

Помимо высокого профессионального уровня предлагаемых нами решений, один из основных принципов нашей деятельности — построение длительных и доверительных отношений с нашими клиентами и партнерами.

## **Основные направления деятельности компании «ЭЛВИС-ПЛЮС»**

***1)*** Реализация крупномасштабных интеграционных проектов по созданию защищённых информационных систем (ЗИС), таких как комплексная система информационной безопасности корпоративной сети НК «ЛУКОЙЛ», интегрированная коммуникационная банковская сеть ЦБ РФ и других.

2) Разработка интеграционных решений по созданию и поддержке защищенных информационных систем в гетерогенных сетях любых организаций (в том числе имеющих территориально-удаленные объекты). Компания разрабатывает решения на базе продуктов по защите информации, в том числе и собственных.

3) Консалтинг в области построения защищённых информационных систем, включая проектный консалтинг и Аттестацию объектов информации.

Продуманная стратегия компании в области развития бизнеса и верные тактические решения позволяют ЭЛВИС-ПЛЮС всегда оставаться в гуще событий, происходящих на мировом и отечественном рынках информационных технологий, участвовать в формировании и развитии перспективных направлений этого рынка. Это подтверждается всей историей существования компании и наличием в нашем активе целого ряда регалий.

В 2016 году ЭЛВИС-ПЛЮС отпраздновала 25 юбилей.

## Базовый доверенный модуль

Логотип Базового доверенного модуля (БДМ) представлен на рисунке 2.



Рисунок 2 – Логотип БДМ

## О продукте

Компания ЭЛВИС-ПЛЮС разработала технологию БДМ в соответствии с требованиями российских регуляторов в области защиты информации и современными мировыми стандартами. Технология предназначена для построения доверенной вычислительной среды на мобильных компьютерах, планшетах и смартфонах, базирующихся на архитектуре Intel x64.

Технология БДМ не противопоставляется реализованным в архитектуре Intel x64 мобильного устройства механизмам доверенной загрузки (PTT, Secure Boot и др.), а может применяться наряду с ними. Они используются для повышения уровня защищённости и выстраивания многоуровневой защиты от несанкционированного воздействия на доверенную вычислительную среду.

Технология БДМ реализуется при помощи программного обеспечения и не требует никаких дополнительных наложенных аппаратных средств, за исключением внешнего носителя ключа (токен либо флеш-память).

Первым сертифицированным ФСБ России продуктом с технологией БДМ является специальное программное обеспечение (СПО) «Мобильное защищённое автоматизированное рабочее место Базовый доверенный модуль».

Функции безопасности БДМ

Действенные меры защиты информации на ноутбуках, ультрабуках, планшетах и смартфонах от современных и будущих угроз:

* многофакторная аутентификация доступа,
* контроль целостности среды обработки информации,
* шифрование всех данных на жёстком диске мобильного устройства.

Реализовать эти меры на мобильных устройствах отечественными средствами защиты информации до сих пор была невозможно из-за несовместимости интерфейсов и форм-факторов аппаратных модулей доверенной загрузки и недостаточной производительности криптоопераций при работе с дисковой памятью. А применение решений иностранного происхождения для криптографической защиты конфиденциальной информации ограничивается или запрещается нормативно-правовыми актами.

Электронный замок

Технология БДМ поддерживает двухфакторную строгую аутентификацию для доверенной загрузки. Поэтому, чтобы получить доступ к доверенной среде функционирования (ОС), недостаточно знать пароль для входа в операционную систему. Пользователь должен иметь ключ (на носителе) и пароль к нему, необходимые для запуска проверки целостности аппаратной и программной части. В сочетании с механизмом контроля целостности это реализует функции электронного замка.

Контроль целостности

Ещё до старта ОС контролируются целостность и неизменность аппаратной конфигурации и BIOS, а после этого с помощью уже проверенных компонент, получив доверие к «железу», — начального загрузчика, СПО БДМ, критичных файлов и настроек операционной системы. Такой подход предупреждает несанкционированные изменения аппаратной части и ОС, которые могли бы повлиять на безопасность данных.

Прозрачное шифрование

БДМ шифрует весь диск (Full Disk Encryption), т.е. пользовательские данные, системный раздел, временные файлы, файлы подкачки, файлы-журналы приложений, дампы памяти и образы рабочей станции, сохраняемые на диске при переходе компьютера в спящий режим.

Утеря или кража компьютера не угрожают конфиденциальности хранимых данных. Раздел жесткого диска, на котором установлена доверенная операционная система, шифруется с использованием алгоритма ГОСТ 28147-89. Пользователь не получит доступ к данным на зашифрованном разделе без прохождения контроля целостности и процедуры двухфакторной аутентификации. Каждый сектор диска шифруется отдельным ключом, что делает практически неразрешимой задачу расшифрования информации злоумышленником в случае кражи или утери мобильного устройства.

Операции шифрования/расшифрования проходят в режиме реального времени в момент обращения к диску прозрачно для приложений и пользователя и не вносят видимых временных задержек, так как производительность криптографических операций сравнима с производительностью операций чтения/записи диска (для Core i5 на одном процессорном ядре скорость шифрования составляет порядка 300 мбайт/с).

## Как работает БДМ

Для запуска доверенной ОС пользователь предъявляет токен и пароль. На токене размещаются ключи для доступа к доверенной операционной системе. При этом СПО БДМ выполняет следующие функции:

1) Доверенная начальная загрузка компьютера с контролем целостности аппаратной конфигурации, BIOS (UEFI), начального сектора диска, СПО БДМ, файлов и настроек операционной системы.

2) Обеспечение конфиденциальности хранимых данных при утере или краже компьютера за счёт прозрачного для приложений шифрования доверенного раздела жёсткого диска по алгоритму ГОСТ 28147-89. Рисунок 3 демонстрирует принцип работы БДМ.

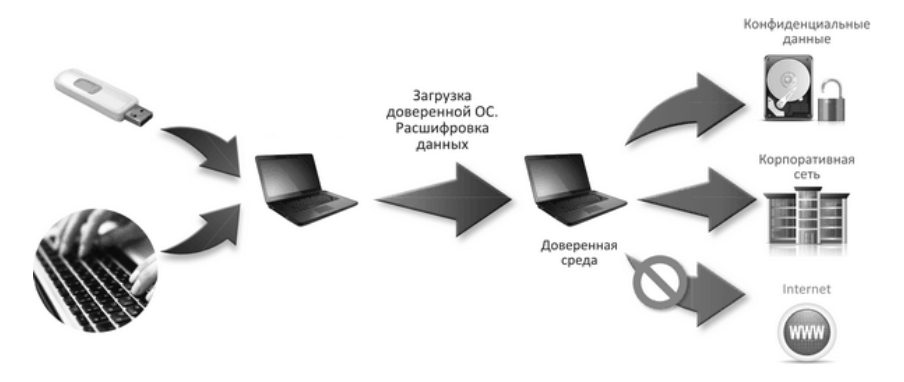


Рисунок 3 – Принцип работы БДМ

Выгоды от внедрения БДМ

1) Устранение противоречий между необходимостью защиты информации и развитием ИТ-сервисов в направлении повышения мобильности сотрудников.

3) Выполнение требований нормативных документов по защите персональных данных и государственных информационных систем.

3) Снижение риска утечек конфиденциальной информации при работе с мобильными АРМ.

4) Уверенность руководства организации в том, что конфиденциальные данные всегда под защитой.

4) Повышение эффективности работы сотрудников за счёт оперативного доступа к корпоративной сети и возможности обработки конфиденциальной информации вне пределов контролируемой зоны.

Изучение Diskpart и Dism (разметка диска и работа с образами)

## Основные понятия:

Раздел (англ. *partition*) — часть долговременной памяти накопителя данных (жёсткого диска, SSD, USB-накопителя), логически выделенная для удобства работы, и состоящая из смежных блоков. Выделение разделов обычно практикуется на внутренних загрузочных дисках компьютера, так как основной его целью является отделение файлов операционной системы от файлов пользователя и от файлов других операционных систем, находящихся на том же физическом носителе

Том (англ. *volume*)  - является аналогом системного диска. После создания структуры диска из разделов, вы можете создать из каждого раздела логический диск или том, отформатировать его и получить таким образом в операционной системе несколько отдельных дисков, так же как это выглядит при присоединении нескольких физических дисков к компьютеру. Каждый из таких томов является самостоятельным и вы можете их форматировать без страха повредить данные на других логических дисках.

## Виды разметок диска

Прежде чем использовать диск, его необходимо разбить на разделы. MBR (Главная загрузочная запись) и GPT (Таблица разделов GUID) представляют собой два различных способа хранения информации о разделах диска. Сюда входят данные о начале и конце разделов, чтобы система знала, к какому разделу принадлежит каждый сектор, и какой раздел является загрузочным.

*Ограничения MBR*

Аббревиатура MBR расшифровывается как Главная загрузочная запись. Этот стандарт был представлен в 1983 году вместе с DOS 2.0 для IBM PC.

Он называется главная загрузочная запись, так как *MBR - это специальный загрузочный сектор*, расположенный в начале диска. Этот сектор содержит загрузчик для установленной операционной системы, а так же информацию о логических разделах диска. Загрузчик - это небольшой кусок кода, который обычно используется для загрузки большого загрузчика с другого раздела или диска. MBR работает с дисками объёмом до 2 Тб., но он может справиться и с дисками большего размера. Кроме этого MBR поддерживает не более 4 основных разделов. Если вам нужно больше, придётся сделать один из основных разделов «расширенным разделом» и разместить в нём логические разделы.

*GPT означает Таблица разделов GUID.* Это новый стандарт, который постепенно приходит на смену MBR. Он является частью UEFI, а UEFI заменяет старый неудобный BIOS так же, как GPT заменяет MBR на что-то более современное. Он называется таблицей разделов GUID, поскольку каждому разделу на вашем диске присваивается «уникальный глобальный идентификатор» или GUID - случайная строка такой длины, что каждый GPT раздел на Земле, скорее всего, обладает уникальным идентификаторов.

У этой системы нет ограничений в отличии от MBR. Диски могут быть гораздо объёмнее, а ограничение на размер будет зависеть от операционной и файловой систем. GPT позволяет создавать практически неограниченное количество разделов.

На MBR диске данные о разделах и загрузочная информация хранятся в одном месте. Если эти данные повреждены или перезаписаны, у вас проблемы. GPT же хранит несколько копий этих данных по всему диска, поэтому работает гораздо быстрее и позволяет восстановить повреждённую информацию. GPT так же хранит значения циклического избыточного кода (CRC), чтобы точно знать, что данные нетронуты. Если информация повреждена, GPT замечает проблему и пытается восстановить повреждённые данные с другого места на диске. MBR не может узнать о повреждении информации. Вы увидите, что возникла проблема, только если не сможете загрузить систему или один из разделов диска исчезнет.

## Программы для управления диском

1) *Diskpart* - это работающий в текстовом режиме командный интерпретатор в операционных системах Windows . Это средство позволяет управлять объектами (дисками, разделами или томами) с помощью сценариев или команд, вводимых из командной строки.

*Команды DiskPart*

Перед использованием команд DiskPart для диска, раздела или тома необходимо сначала составить список объектов, а затем выбрать объект для работы. После этого все введенные команды DiskPart выполняются для данного объекта.

Основные команды Diskpart представлены в таблице 1.

Таблица 1

|  |  |
| --- | --- |
| Команда | Описание |
| select | Выбрать объект |
| list disk | Отобразить все диски |
| list volume | Отобразить все тома |
| list partition. | Отобразить все разделы |
| assign | Назначение буквы диску |
| clean | Удалить все разделы или тома на диске, имеющем фокус. |
| convert gpt/mbr | Преобразовать диск в GPT/MBR |
| create partition/volume | Создать раздел/том |
| detail | Вывести детальные параметры выбранного объекта |
| format | Форматировать том или раздел |

2) *DISM.exe* (система обслуживания образов развертывания и управления ими) - это средство командной строки, которое может использоваться для обслуживания образа Windows® или для подготовки образа среды предустановки Windows (Windows PE).

Основные параметры *DISM* представлены в таблице 2*.*

Таблица 2

|  |  |
| --- | --- |
| Параметр | Описание |
| /Apply-Image | Применяет образ к указанному диску. |
| /Capture-Image | Записывает образ диска в новый WIM-файл. В записанные каталоги включаются все вложенные папки и данные. |
| /Get-ImageInfo | Отображает информацию об образах, содержащихся в WIM-файле. |
| /Mount-Image | Подключает образ из WIM-файла к указанному каталогу, чтобы сделать его доступным для обслуживания. |
| /Split-Image | Разбивает существующий WIM-файл на несколько разделенных WIM-файлов, доступных только для чтения. |

Индивидуальное задание

Разработать программу для заполнения банков памяти информацией согласно пользовательским данным.

## Выбор инструментария

Для выполнения индивидуального задания мною был выбран язык программирования Python 3.5, для создания графического интерфейса – библиотека PyQt5.

## Теоретические сведения

Выбор языка

Выбирая язык программирования, я остановила свой выбор на Python версии 3.5. Тому способствовал ряд причин.

Python – это современный, мощный и в то же время изящный язык программирования. Он относится к группе интерпретируемых, высокоуровневых языков программирования общего назначения с поддержкой целого ряда парадигм: структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное стили программирования.

На сегодняшний день Python используют практически все крупные корпорации в той или иной степени, ведь сегодня интерпретируемый не значит медленный, как это считалось раньше. Конечно, языки C/C++ или Java работают несколько быстрее, но учитываю быстроту освоения, высокий уровень абстракций, предоставляемый Python, и ряд других особенностей этот язык стал необычайно популярен в сфере создания и тестирования пользовательских программ.

Ниже перечислены некоторые, по моему мнению, самые важные преимущества данного языка:

1) Портируемость.

Наверное, самая важная особенность. Это означает, что программу , написанную под одну платформу, можно запустить на другой без каких-либо изменений.

2) Синтаксис.

Удобный для чтения и понимания синтаксис Python – одно из главных преимуществ. В сфере коммерческой разработки это означает меньшее время разработки программного продукта в группе программистов. Код пишется быстрее, простота синтаксиса минимизирует проблемы с последующим его обслуживанием, что делает процесс разработки менее затратным.

3) Интерпретируемый язык.

Как замечалось выше, здесь присутствуют свои недостатки, но в целом это дает возможность намного упростить отладку программ.

4) Поддержка ООП.

Объектно – ориентированное программирование на сегодняшний день – одна из основных парадигм программирования, позволяющая разработчику создавать высокоуровневые абстракции с понятным интерфейсом, что делает код более понятным и хорошо структурированным.

5) Динамическая типизация и автоматическая сборка мусора.

Программисту не нужно заранее объявлять тип переменной, интерпретатор сам выберет нужный тип исходя из контекста ее использования.   
Автоматическая сборка мусора предотвращает утечки памяти – проблему, которую довольно сложно отловить на этапе отладки.

6) Наличие большого количества как встроенных, так и сторонних библиотек.

Большинство проблем уже решено, так зачем же собирать свой велосипед? Готовые модули есть практически для каждой области, начиная web-разработкой и заканчивая сложными многопоточными играми. Также ведутся работы над библиотеками, позволяющими эффективно работать с большими данными.

7) Интеграция с другими языками программирования.

Если возможностей Python все же недостаточно, то можно использовать удобно реализованные инструменты интеграции с C/C++ и другими языками.

8) Грамотная и подробная документация.

9) Кроме того, Python прост в изучении, прививает способность писать структурированный и понятный код.

## Графический интерфейс программы

Графический интерфейс реализованной программы представлен на рисунках 4, 5, 6.

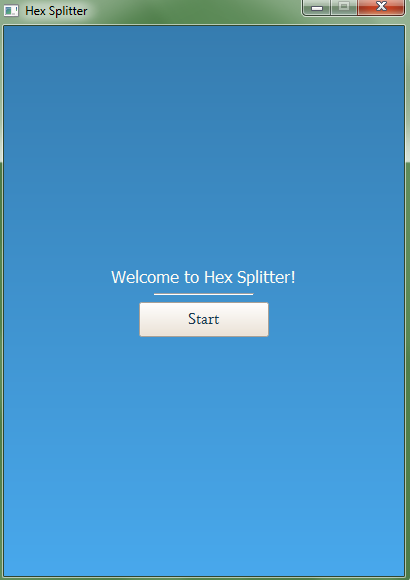


Рисунок 4 – Окно приветствия

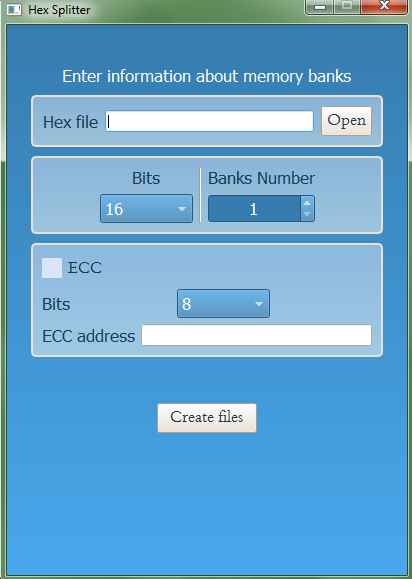


Рисунок 5 – Окно ввода параметров

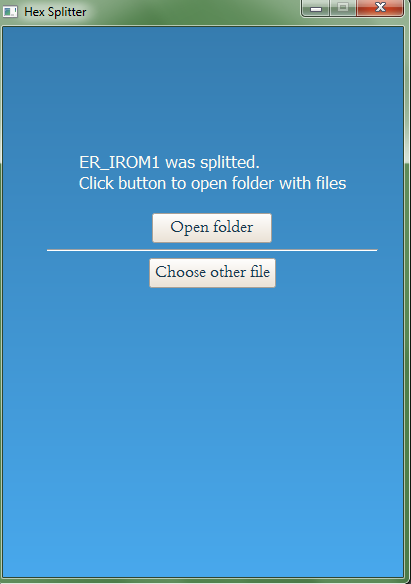


Рисунок 6 – Окно успешного распределения данных

## Структура программы

Проект разработанной программы состоит из 11 файлов, краткое описание которых приведено в таблице 3.

Таблица 3

|  |  |
| --- | --- |
| Название файла | Описание |
| \_\_main\_\_.py | Создание и запуск приложения |
| ECC.py | Создание кода коррекции ошибок для исходных данных |
| Exceptions.py | Пользовательские исключения |
| Helpers.py | Функции, необходимые для работы модулей программы |
| HexSplitter.py | Получение информации из исходного файла и ее распределение по файлам согласно данным, введенным пользователем |
| MainPage.py | Окно выбора параметров |
| Page.py | Базовый класс для всех окон |
| PageManager.py | Управление окнами |
| StartPage.py | Начальное окно приложения |
| SucsessPage.py | Окно успешного выполнения разделения файла. |
| main.ui | Описание пользовательского интерфейса программы в формате XML |

## Принцип работы программы

При первоначальном запуске программы запускается файл *\_\_main\_\_.py,* отрывок кода c комментариями представлен ниже:

app = QApplication(sys.argv) # создание объекта приложения

app.setStyle(QStyleFactory.create("Fusion")) # применение стиля #Fusion для приложения

window = PageManager() # создание объекта класса PageManager, #который управляет переключениями окон

window.show() # отображение окна на экране

sys.exit(app.exec\_()) # запуск цикла выполнения приложения

\_\_main\_\_.py создает объект класса PageManager. Класс объявлен в модуле PageManager.py. Ниже представлен отрывок кода этого класса:

class PageManager(QMainWindow): # объявления класса; наследуется # от QMainWindow  
 def \_\_init\_\_(self): # метод для инициализации объекта класса  
 super().\_\_init\_\_() # вызов метода \_\_init\_\_() базового класса  
  
 self.initUi() # инициализация пользовательского интерфейса  
 self.all\_ui.setWindowTitle('Hex Splitter') # установка заголовка #программы  
 self.init\_pages() # инициализация окон  
 self.connect\_signals() # связывание сигналов от ui-формы со #слотами  
  
 self.open\_page('Start') # открытие приветственного окна  
  
 def initUi(self):  
 self.all\_ui = uic.loadUi('ui\\main.ui', self) # загрузка ui в проект  
  
 def init\_pages(self):

# Создание объектов классов для отображения окон  
 self.Start = StartPage(self.all\_ui)  
 self.Main = MainPage(self.all\_ui)  
 self.Sucsess = SucsessPage(self.all\_ui)  
  
 def connect\_signals(self):  
 self.Main.open\_page.connect(self.open\_page)  
 self.Sucsess.open\_page.connect(self.open\_page)  
 self.Sucsess.reset.connect(self.reset)

def open\_page(self, page):  
  
 if page == 'Start':  
 self.all\_ui.stackedWidget.setCurrentIndex(0)  
 elif page == 'Main':  
 self.all\_ui.stackedWidget.setCurrentIndex(1)  
 elif page == 'Sucsess':  
 self.all\_ui.stackedWidget.setCurrentIndex(2)  
 self.Sucsess.late\_init(self.Main.file\_path)  
 else:  
 raise NameError('Wrong page name: \"{}\"'.format(page))

Далее открывается окно приветствия. Оно имеет лишь 1 кнопку «Start», при нажатии на которую открывается окно ввода параметров.

Окно ввода параметров представлено объектом класса MainWindow. Окно содержит в себе виджеты, позволяющие пользователю ввести данные о требуемой конфигурации создаваемых файлов (или блоков памяти): выбрать исходный файл (здесь осуществляется проверка на корректность выбранного файла). Если все поля заполнены верно, становится доступна кнопка «Create files», при нажатии на которую вызывается метод инициализации класса HexSplitter, в качестве параметров которому передаются данные, введенные пользователем.

Класс HexSplitter играет основную роль в логике исполнения программы, иными словами – это ее «мозг».

Сначала метод self.\_\_create\_empty\_hex\_files() создает отдельную директорию рядом с файлом, который выбрал пользователь. Затем внутри этой директории создается требуемое количество пустых файлов, которые далее будут заполняться.

Далее метод self.\_\_write\_split\_hex\_files() производит запись информации в пустые файлы согласно параметрам, выбранным пользователем.

Если распределение данных прошло успешно, открывается третье окно, за создание которого отвечает класс SucsessPage. В этом окне пользователь видит информацию об успешном создании файлов. Также здесь присутствует 2 кнопки: первая позволяет открыть в проводнике папку с созданными файлами, вторая – вернуться на страницу выбора параметров.

Если же запись файлов не была успешной, пользователь видит сообщение об ошибке и кнопку, позволяющую вернуться к выбору параметров.

*Тестирование*

Разработанная программа была проверена на нескольких входных данных. Результаты работы корректны.

Список источников

1) **elvis**.ru/

**2)** <https://ru.wikipedia.org/wiki/Python>

**3)** <https://docs.python.org/3/>

**4)** <http://pyqt.sourceforge.net/Docs/PyQt4/classes.html>