**Санкт-Петербургский политехнический университет Петра Великого**

**Институт компьютерных наук и технологий**

Кафедра компьютерных систем и программных технологий

**Курсовой проект**

**Дисциплина**: «Проектирование ОС и компонентов»

**Тема**: Разработка инсталлятора

Санкт –Петербург

2017

# Постановка задачи

Необходимо разработать собственный инсталлятор.

**Язык программирования**: C++

**Среда разработки**: Microsoft Visual Studio 2013

**Инструментарий пользователя**: Консольное приложение

Процесс инсталляции:

* 1. получение системных данных путем сбора информации из реестра
  2. получение данных пользователя
  3. создание архива
  4. проверка свободного места на диске
  5. распаковка
  6. настройка системного реестра
  7. создание ярлыка и ссылок в меню «Пуск»
  8. размещение в каталоге установленных программ
  9. размещение в списке деинсталлируемых программ

## Используемая система

C:\Users\Helen>systeminfo

Имя узла: DESKTOP-1MBMOD7

Название ОС: Майкрософт Windows 10 Pro

Версия ОС: 10.0.10240 Н/Д построение 10240

Изготовитель ОС: Microsoft Corporation

Параметры ОС: Изолированная рабочая станция

Построение ОС: Multiprocessor Free

Зарегистрированный владелец: Helen

Зарегистрированная организация:

Код продукта: 00330-80000-00000-AA809

Дата установки: 20.01.2017, 21:49:03

Время загрузки системы: 12.06.2017, 20:20:04

Изготовитель системы: Gigabyte Technology Co., Ltd.

Модель системы: H87-HD3

Тип системы: x64-based PC

Процессор(ы): Число процессоров - 1.

[01]: Intel64 Family 6 Model 60 Stepping 3 GenuineIntel ~3201 МГц

Версия BIOS: American Megatrends Inc. F3, 09.05.2013

Папка Windows: C:\Windows

Системная папка: C:\Windows\system32

Устройство загрузки: \Device\HarddiskVolume2

Язык системы: ru;Русский

Язык ввода: ru;Русский

Часовой пояс: (UTC+03:00) Москва, Санкт-Петербург, Волгоград

Полный объем физической памяти: 3 975 МБ

Доступная физическая память: 1 463 МБ

Виртуальная память: Макс. размер: 5 652 МБ

Виртуальная память: Доступна: 2 409 МБ

Виртуальная память: Используется: 3 243 МБ

Расположение файла подкачки: C:\pagefile.sys

Домен: WORKGROUP

# Теоретический обзор

## Windows Installer

Windows Installer (установщик Windows) — подсистема Microsoft Windows, обеспечивающая установку программ (инсталлятор). Является компонентом Windows, начиная с Windows 2000; может доустанавливаться и на более ранние версии Windows. Вся необходимая для установки информация (иногда и вместе с устанавливаемыми файлами) содержится в установочных пакетах (installation packages), имеющих расширение .msi.

## Inno Setup

Inno Setup — система создания инсталляторов для Windows программ с открытым исходным кодом. Впервые выпущенный в 1997 году, Inno Setup сегодня конкурирует и даже превосходит многие коммерческие установщики по функциональности и стабильности.

## Ключевые особенности

* Поддержка всех современных версий Windows: 10, 8, 7, 2008 R2, Vista, XP, 2003, 2000, 9x, NT 4.0. Поддерживает установку 64-битных программ на 64-битных выпусках Windows Vista, Windows 7, Windows XP, и Windows Server 2003. Также поддерживает 64-разрядные процессоры. (При использовании процессоров Itanium необходимо установить Service Pack).
* Поддерживает создание одиночных EXE для облегчения установки и распространения программ через сеть Интернет.
* Стандартный интерфейс мастера установки в стиле Windows 2000/XP.
* Возможность выбора типа установки, например: Полный, Минимальный, Выборочный.
* Имеет встроенную поддержку DEFLATE, bzip2 и 7-Zip LZMA/LZMA2 сжатий. Установщик умеет сравнивать версии файлов, заменять встроенные файлы, устанавливать разделяемые файлы, регистрировать DLL/OCX библиотеки и устанавливать шрифты.
* Позволяет создавать ярлыки в меню «Пуск» и на «Рабочем столе».
* Позволяет создавать записи в реестре и .ini-файлы.
* Имеет встроенную поддержку скриптов на Object Pascal (посредством Pascal Script[en])[1].
* Поддерживает создание многоязычных инсталляторов программ.
* Поддерживает Unicode и языки с направлением письма справа налево.
* Поддерживает установку пароля и шифрование инсталляторов программ.
* Доступны исходные тексты на Borland Delphi 2.0-5.0 и 2009.

## InstallShield

InstallShield — система создания инсталляторов и пакетов программного обеспечения для Microsoft Windows и Linux с закрытыми исходными кодами.

Программный инструмент используется главным образом для установки программного обеспечения для Microsoft Windows, как на десктопные, так и серверные платформы, а также может быть использован для управления приложениями на различных портативных и мобильных устройствах.

Версия InstallShield 2011 была выпущена на рынок 19 августа 2010 года, она получила поддержку Windows 7, Windows Server 2008 R2, MSI 5, и являлась единственной установкой для поддержки Microsoft Application Virtualization, которая широко использует формат виртуализации приложений. Поддержка Visual Studio 2010, .NET Framework 4, интеграция с Team Foundation Server Build для компиляции и сборки проектов. Улучшенная поддержка 64-х битных приложений.

Последняя на данный момент версия InstallShield 2015 получила поддержку Windows 10. В SP1 было введено исправления данной поддержки. Также, в версии 2015 года была введена поддержка Visual Studio 2015 и использование цифровых сертификатов с SHA-256. В SP1 была введена поддержка Microsoft App-V 5.1 и .NET Framework 4.6.

## WiX

The Windows Installer Xml (WiX) toolset — набор инструментов, позволяющих создавать инсталляционные пакеты Windows Installer (.MSI и .MSM) на основе XML-описаний.

Программы из набора WiX предназначены для работы в командной строке, однако существует плагин Votive, который позволяет работать с исходными кодами инсталлятора как с проектом Visual Studio (проектный файл имеет расширение .wixproj). При этом поддерживаются обычные возможности проектов: подсветка синтаксиса, построение из командной строки, IntelliSense и т. д.

Windows Installer Xml был выпущен компанией Microsoft в апреле 2004 года под лицензией CPL и размещён на сайте SourceForge.net (файлы исходного кода были перемещены на codeplex.com). WiX стал первым проектом, выпущенным компанией под открытой лицензией. Создатель и ведущий разработчик WiX — Rob Mensching.

## Nullsoft Scriptable Install System

Nullsoft Scriptable Install System (NSIS) — система создания установочных программ для Microsoft Windows с открытыми исходными кодами, созданная компанией Nullsoft, основанной создателями Winamp. NSIS создан как альтернатива InstallShield, предназначенного для коммерческих продуктов.

Особенности:

* Очень маленький размер установочного блока (около 34 Кбайт)
* zlib, bzip2 и LZMA-сжатие
* Гибкая система скриптов
* Мультиязычность
* Поддержка плагинов
* Скрипт препроцессор
* Создание Web-установщиков

## Smart Install Maker

Smart Install Maker — это условно-бесплатная утилита с закрытым исходным кодом, предназначенная для создания инсталляторов в 32-битных операционных системах Microsoft Windows.

Утилита предоставляет алгоритм Cabinet для сжатия и создания компактных инсталляторов, поддерживает создание многоязычного инсталлятора (до 25 языков), а также предоставляет средства для управления внешним видом программы-установки.

Среди возможностей Smart Install Maker, можно выделить создание деинсталлятора, ярлыков в указанных каталогах системы, регистрацию новых шрифтов или расширений в операционной системе, поддержку ActiveX и переменных, проверку установленного .NET Framework, разделение инсталлятора на установочные диски, детальную настройку информации об установочном файле, системных требований (например, запретить установку программы в Windows 95) а также его внешнего вида и многое другое.

## CreateInstall

Идеология инсталлятора CreateInstall основана на двух главных принципах. Полный контроль над процессом установки и неограниченная расширяемость. Весь сценарий установки представляется в виде последовательности команд.

## IzPack

Главные особенности IzPack: XML-синтаксис, интеграция с различными инструментами компиляции, простой и гибкий API-интерфейс, возможность локализации.

Среди особенностей инсталляторов, созданных средствами IzPack, — небольшой размер и модульность. В основе дистрибутива лежит XML-документ, где указываются все необходимые спецификации, определяется функциональность продукта. Впоследствии конфигурационный файл обрабатывается любым удобным способом, для этого используется командная строка или Apache Ant. Для наращивания функциональности инсталлятора могут быть задействованы IDE среды, такие как Eclipse, IntelliJ IDEA или Netbeans. Для составления конфигурационного файла подойдет любой редактор кода.

В XML-документе указывается информация о проекте (<info>), интерфейс (<guiprefs>), локализации (<locale>), файлы (<files>), ресурсы (<resources>), панели (<panels>), содержимое (<packs> — может быть упаковано в raw, gzip, bzip2) и др. Что касается java, в IzPack есть возможность применения действий (custom acrions) и переменных — встроенных, пользовательских и переменных среды.

## Excelsior Installer

Создание дистрибутива или обновления производится с помощью упаковщика (Packager). Он содержит пошаговый мастер, содержащий настройки проекта. На основе проекта очень просто скомпилировать восьмишаговый инсталлятор. Согласно описанию, это делается и вовсе двумя кликами мыши, но в обязательном порядке в упаковщике необходимо определить следующие параметры:

* Файлы и папки приложения, ярлыки
* Описание: имя компании, название продукта, версия продукта

Упаковщик состоит из разделов Files (Файлы), Settings (Настройки) и Appearance (Внешний вид). В целом, здесь содержится стандартный перечень опций, отвечающих за создание лицензионного соглашения, выбор инсталляционной директории, ярлыков, ключей реестра и другие параметры.

## Описание реестра

5-е издание словаря Microsoft Computer Dictionary содержит следующее определение реестра:

Иерархическая централизованная база данных, используемая в ОС Microsoft Windows 98, Windows CE, Windows NT и Windows 2000 для хранения сведений, необходимых для настройки операционной системы для работы с пользователями, программными продуктами и устройствами.

В реестре хранятся данные, которые необходимы для правильного функционирования Windows. К ним относятся профили всех пользователей, сведения об установленном программном обеспечении и типах документов, которые могут быть созданы каждой программой, информация о свойствах папок и значках приложений, а также установленном оборудовании и используемых портах.

Системный реестр заменяет собой большинство текстовых INI-файлов, которые использовались в Windows 3.x, а также файлы конфигурации MS-DOS, такие как Autoexec.bat и Config.sys. Версии реестра для разных версий операционных систем семейства Windows имеют определенные различия.

Куст реестра - это группа разделов, подразделов и параметров реестра с набором вспомогательных файлов, содержащих резервные копии этих данных. Вспомогательные файлы для всех кустов за исключением HKEY\_CURRENT\_USER хранятся в системах Windows NT 4.0, Windows 2000, Windows XP, Windows Server 2003 и Windows Vista в папке %SystemRoot%\System32\Config.

Вспомогательные файлы для куста HKEY\_CURRENT\_USER хранятся в папке %SystemRoot%\Profiles\Имя\_пользователя. Расширения имен файлов в этих папках указывают на тип содержащихся в них данных. Отсутствие расширения также иногда может указывать на тип содержащихся в файле данных.

|  |  |
| --- | --- |
| Куст реестра | Вспомогательные файлы |
| HKEY\_LOCAL\_MACHINE\SAM | Sam, Sam.log, Sam.sav |
| HKEY\_LOCAL\_MACHINE\Security | Security, Security.log, Security.sav |
| HKEY\_LOCAL\_MACHINE\Software | Software, Software.log, Software.sav |
| HKEY\_LOCAL\_MACHINE\System | System, System.alt, System.log, System.sav |
| HKEY\_CURRENT\_CONFIG | System, System.alt, System.log, System.sav, Ntuser.dat, Ntuser.dat.log |
| HKEY\_USERS\DEFAULT | Default, Default.log, Default.sav |

Таблица содержит перечень и краткое описание стандартных разделов. Максимальная длина имени раздела составляет 255 символов.

|  |  |
| --- | --- |
| Папка/стандартный раздел | Описание |
| HKEY\_CURRENT\_USER | Данный раздел является корневым для данных конфигурации пользователя, вошедшего в систему в настоящий момент. Здесь хранятся папки пользователя, цвета экрана и параметры панели управления. Эти сведения сопоставлены с профилем пользователя. Вместо полного имени раздела иногда используется аббревиатура HKCU. |
| HKEY\_USERS | Данный раздел содержит все активные загруженные профили пользователей компьютера. Раздел HKEY\_CURRENT\_USER является подразделом раздела HKEY\_USERS. Вместо полного имени раздела иногда используется аббревиатура HKU. |
| HKEY\_LOCAL\_MACHINE | Раздел содержит параметры конфигурации, относящиеся к данному компьютеру (для всех пользователей). Вместо полного имени раздела иногда используется аббревиатура HKLM. |
| HKEY\_CLASSES\_ROOT | Является подразделом HKEY\_LOCAL\_MACHINE\Software. Хранящиеся здесь сведения обеспечивают выполнение необходимой программы при открытии файла с использованием проводника. Вместо полного имени раздела иногда используется аббревиатура HKCR. Начиная с Windows 2000, эти сведения хранятся как в HKEY\_LOCAL\_MACHINE, так и в HKEY\_CURRENT\_USER. Раздел HKEY\_LOCAL\_MACHINE\Software\Classes содержит параметры по умолчанию, которые относятся ко всем пользователям локального компьютера. Параметры, содержащиеся в разделе HKEY\_CURRENT\_USER\Software\Classes, переопределяют принятые по умолчанию и относятся только к текущему пользователю. Раздел HKEY\_CLASSES\_ROOT включает в себя данные из обоих источников. Кроме того, раздел HKEY\_CLASSES\_ROOT предоставляет эти объединенные данные программам, разработанным для более ранних версий Windows. Изменения настроек текущего пользователя выполняются в разделе HKEY\_CURRENT\_USER\Software\Classes. Модификация параметров по умолчанию должна производиться в разделе HKEY\_LOCAL\_MACHINE\Software\Classes. Данные из разделов, добавленных в HKEY\_CLASSES\_ROOT, будут сохранены системой в разделе HKEY\_LOCAL\_MACHINE\Software\Classes. Если изменяется параметр в одном из подразделов раздела HKEY\_CLASSES\_ROOT и такой подраздел уже существует в HKEY\_CURRENT\_USER\Software\Classes, то для хранения информации будет использован раздел HKEY\_CURRENT\_USER\Software\Classes, а не HKEY\_LOCAL\_MACHINE\Software\Classes. |
| HKEY\_CURRENT\_CONFIG | Данный раздел содержит сведения о профиле оборудования, используемом локальным компьютером при запуске системы. |

Следующая таблица содержит список типов данных, определенных и используемых Windows.

|  |  |  |
| --- | --- | --- |
| **Имя** | **Тип** | **Описание** |
| Двоичный параметр | REG\_BINARY | Необработанные двоичные данные. Большинство сведений об аппаратных компонентах хранится в виде двоичных данных и выводится в редакторе реестра в шестнадцатеричном формате. |
| Параметр DWORD | REG\_DWORD | Данные представлены в виде значения, длина которого составляет 4 байта (32-разрядное целое). Этот тип данных используется для хранения параметров драйверов устройств и служб. Значение отображается в окне редактора реестра в двоичном, шестнадцатеричном или десятичном формате. Эквивалентами типа DWORD являются DWORD\_LITTLE\_ENDIAN (самый младший байт хранится в памяти в первом числе) и REG\_DWORD\_BIG\_ENDIAN (самый младший байт хранится в памяти в последнем числе). |
| Расширяемая строка данных | REG\_EXPAND\_SZ | Строка данных переменной длины. Этот тип данных включает переменные, обрабатываемые при использовании данных программой или службой. |
| Многострочный параметр | REG\_MULTI\_SZ | Многострочный текст. Этот тип, как правило, имеют списки и другие записи в формате, удобном для чтения. Записи разделяются пробелами, запятыми или другими символами. |
| Строковый параметр | REG\_SZ | Текстовая строка фиксированной длины. |
| Двоичный параметр | REG\_RESOURCE\_LIST | Последовательность вложенных массивов. Служит для хранения списка ресурсов, которые используются драйвером устройства или управляемым им физическим устройством. Обнаруженные данные система сохраняет в разделе \ResourceMap. В окне редактора реестра эти данные отображаются в виде двоичного параметра в шестнадцатеричном формате. |
| Двоичный параметр | REG\_RESOURCE\_REQUIREMENTS\_LIST | Последовательность вложенных массивов. Служит для хранения списка драйверов аппаратных ресурсов, которые могут быть использованы определенным драйвером устройства или управляемым им физическим устройством. Часть этого списка система записывает в раздел \ResourceMap. Данные определяются системой. В окне редактора реестра они отображаются в виде двоичного параметра в шестнадцатеричном формате. |
| Двоичный параметр | REG\_FULL\_RESOURCE\_DESCRIPTOR | Последовательность вложенных массивов. Служит для хранения списка ресурсов, которые используются физическим устройством. Обнаруженные данные система сохраняет в разделе \HardwareDescription. В окне редактора реестра эти данные отображаются в виде двоичного параметра в шестнадцатеричном формате. |
| Отсутствует | REG\_NONE | Данные, не имеющие определенного типа. Такие данные записываются в реестр системой или приложением. В окне редактора реестра отображаются в виде двоичного параметра в шестнадцатеричном формате. |
| Ссылка | REG\_LINK | Символическая ссылка в формате Юникод. |
| Параметр QWORD | REG\_QWORD | Данные, представленные в виде 64-разрядного целого. Начиная с Windows 2000, такие данные отображаются в окне редактора реестра в виде двоичного параметра. |

В рамках индивидуального задания было написано приложение, разберем его исходных код, рассмотрим основные функции, используемые в программе.

# Функциональная структура приложения

Функциональная структура приложения приведена на рисунке 1.

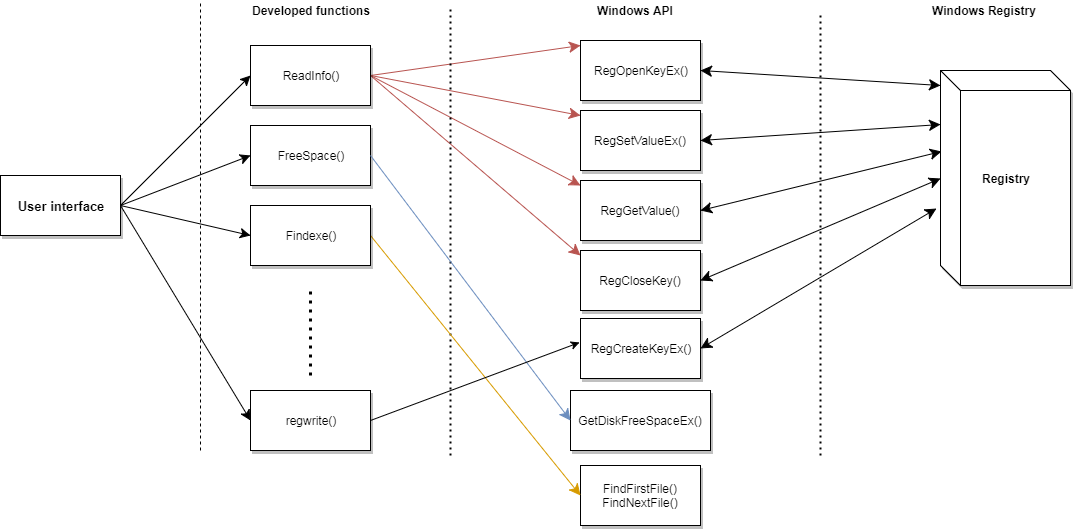


Рис. 1 Функциональная структура приложения

# Описание функций используемых в приложении

## Функции работы с реестром

**Функция RegOpenKeyEx** открывает указанный ключ.

**Синтаксис**:

LONG RegOpenKeyEx(

HKEY hKey, // дескриптор указанного ключа

LPCTSTR lpSubKey, // адрес имени открываемого подключа

DWORD ulOptions, // зарезервировано

REGSAM samDesired, // маска доступа безопасности

PHKEY phkResult // адрес дескриптора открытого ключа

);

Параметры:

|  |  |
| --- | --- |
| hKey | Идентифицирует открытый в текущий момент ключ или один из следующих предопределенных значений дескрипторов:  HKEY\_CLASSES\_ROOT HKEY\_CURRENT\_USER HKEY\_LOCAL\_MACHINE HKEY\_USERS HKEY\_CURRENT\_CONFIG |
| lpSubKey | Адрес нульзаканчивающейся строки, содержащей имя ключа на открытие. Если этот параметр равен NULL или адрес указывает на пустую строку, функция возвращает тот-же дескриптор, который ей был дан в параметре hKey. В этом случае, функция не закрывает прежде закрыет дескриптор. |
| ulOptions | Зарезервировано; должно быть нулевым. |
| samDesired | Определяет маску доступа, которая определяет желаемый доступ безопасности для нового ключа. Этот параметр может быть комбинацией из следующих значений:   |  |  | | --- | --- | | Значение: | Описание: | | KEY\_ALL\_ACCESS | Комбинация из KEY\_QUERY\_VALUE, KEY\_ENUMERATE\_SUB\_KEYS, KEY\_NOTIFY, KEY\_CREATE\_SUB\_KEY, KEY\_CREATE\_LINK, и KEY\_SET\_VALUE. | | KEY\_CREATE\_LINK | Права создавать символические ссылки. | | KEY\_CREATE\_SUB\_KEY | Права создавать подключи. | | KEY\_ENUMERATE\_SUB\_KEYS | Права перечислять подключи. | | KEY\_EXECUTE | Права на доступ ко чтению. | | KEY\_NOTIFY | Права на изменение нотификации. | | KEY\_QUERY\_VALUE | Права запрашивать данные подключей. | | KEY\_READ | Комбинация из KEY\_QUERY\_VALUE, KEY\_ENUMERATE\_SUB\_KEYS, и KEY\_NOTIFY. | | KEY\_SET\_VALUE | Права устанавливать данные подключей. | | KEY\_WRITE | Комбинация из KEY\_SET\_VALUE и KEY\_CREATE\_SUB\_KEY. | |

**Возвращаемые значения**:

Если функция успешна, возвращается значение ERROR\_SUCCESS.

При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.

**Функция RegSetValueEx** сохраняет данные в поле значения открытого ключа реестра. Она, также, может устанавливать дополнительные значения и типы информации для указанного ключа.

**Синтаксис**:

LONG RegSetValueEx(

HKEY hKey, // дескриптор ключа

LPCTSTR lpValueName,// адрес имени установливаемого значения

DWORD Reserved, // зарезервировано

DWORD dwType, // тип данных

CONST BYTE \*lpData, // адрес данных для установки

DWORD cbData // размер данных

);

Параметры:

|  |  |
| --- | --- |
| hKey | Идентифицирует открытый в текущий момент ключ или один из следующих предопределенных значений дескрипторов:  HKEY\_CLASSES\_ROOT HKEY\_CURRENT\_USER HKEY\_LOCAL\_MACHINE HKEY\_USERS HKEY\_CURRENT\_CONFIG  Возвращаемые ключи относительны ключу, указанному в hKey. |
| lpValueName | Адрес нульзаканчивающейся строки, содержащей имя устанавливаемого значения. Если значение с таким именем не существует в ключе реестра, функция его создает.  Если этот параметр равен NULL или ссылается на пустую строку и параметр dwType установлен в тип REG\_SZ, функция устанавливает такое же значение, как и функция RegSetValue. |
| Reserved | Зарезервировано; должно быть NULL |
| dwType | Определяет тип сохраняемых данных значения. Этот параметр может быть одним из следующих значений:   |  |  | | --- | --- | | Значение: | Назначение: | | KEY\_ALL\_ACCESS | Комбинация из KEY\_QUERY\_VALUE, KEY\_ENUMERATE\_SUB\_KEYS, KEY\_NOTIFY, KEY\_CREATE\_SUB\_KEY, KEY\_CREATE\_LINK, KEY\_SET\_VALUE. | | KEY\_CREATE\_LINK | Права создавать символические ссылки. | | KEY\_CREATE\_SUB\_KEY | Права создавать подключи. | | KEY\_ENUMERATE\_SUB\_KEYS | Права перечислять подключи. | | KEY\_EXECUTE | Права на доступ к чтению. | | KEY\_NOTIFY | Права на изменение нотификации. | | KEY\_QUERY\_VALUE | Права запрашивать данные подключей. | | KEY\_READ | Комбинация из KEY\_QUERY\_VALUE, KEY\_ENUMERATE\_SUB\_KEYS, KEY\_NOTIFY. | | KEY\_SET\_VALUE | Права устанавливать данные подключей. | | KEY\_WRITE | Комбинация из KEY\_SET\_VALUE KEY\_CREATE\_SUB\_KEY. | |

**Возвращаемые значения**:

Если функция успешно, возвращается значение ERROR\_SUCCESS.

При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.

Размер данных ограничен возможностями системы. Длинные значения (большие за 2048 байт) лучше хранить в файле и указывать путь к файлу в реестре. Это позволяет реестру работать более эффективно. Такие элементы приложения, как иконки, рисунки и исполняющиеся файлы должны храниться как файлы, а не помещатся в реестр.

Ключ, указанный в hKey, должен быть открытым с правом доступа KEY\_SET\_VALUE. Для открытия ключа, используйте функцию RegCreateKeyEx или RegOpenKeyEx.

Если dwType установлен в тип REG\_SZ, REG\_MULTI\_SZ или REG\_EXPAND\_SZ и используется ANSI версия этой функции (если произведен точный вызов RegSetValueEx или не был объявлен Unicode перед добавлением заголовочного файла WINDOWS.H), парметр lpData должен быть строкой ANSI. Строка преобразовывается в Unicode перед сохранением в реестре.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Функция RegSetValue** устанавливает значение указанному ключу.  Это значение должно быт текстовой строкой и не должно иметь имени. Эта функция оставлена для совместимости с Windows версии 3.1. Win32-приложения должны использовать функцию RegSetValueEx, которая позволяет приложению устанавливать любое количество значений с именами данных любых типов.  **Синтаксис**:  LONG RegSetValue(  HKEY hKey, // дескриптор ключа  LPCTSTR lpSubKey, // адрес имени подключа  DWORD dwType, // тип значения  LPCTSTR lpData, // адрес данных значения  DWORD cbData // размер данных  );  **Параметры**:   |  |  | | --- | --- | | hKey | Идентифицирует открытый в текущий момент ключ или один из следующих предопределенных значений дескрипторов: HKEY\_CLASSES\_ROOT HKEY\_CURRENT\_USER HKEY\_LOCAL\_MACHINE HKEY\_USERS HKEY\_CURRENT\_CONFIG | | lpSubKey | Адрес нульзаканчивающейся строки, содержащей имя подключа, в котором будет установлено значение. Этот параметр может быть нулевым или указывать на пустую строку. В этом случае, значение будет добавлено в ключ, указанный в параметре hKey. | | dwType | Указывает тип сохраняемой информации. Этот параметр должен быть REG\_SZ. Для сохранения других типов данных, используйте функцию RegSetValueEx. | | lpData | Адрес нульзаканчивающейся строки, содержащей значение к установке его в указанный ключ. | | cbData | Указывает размер, в байтах, строки lpData, без учета завершающего строку нулевого символа. |   **Возвращаемые значения**:  Если функция успешно, возвращается значение ERROR\_SUCCESS.  При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.  Если ключ, указанный в параметре lpSubKey, не существует, функция RegSetValue создает его.  Размер данных ограничен возможностями системы. Длинные значения (большие за 2048 байт) лучше хранить в файле и указывать путь к файлу в реестре. Это позволяет реестру работать более эффективно.  Ключ, указанный в hKey, должен быть открытым с правом доступа KEY\_SET\_VALUE. Для открытия ключа, используйте функцию RegCreateKeyEx илиRegOpenKeyEx.  Если используется ANSI версия этой функции (если произведен точный вызов RegSetValue или не был объявлен Unicode перед добавлением заголовочного файла WINDOWS.H), парметр lpData должен быть строкой ANSI. Строка преобразовывается в Unicode перед сохранением в реестре. |

**Функция RegCloseKey** освобождает дескриптор указанного ключа.

**Синтаксис**:

LONG RegCloseKey(

HKEY hKey // дескриптор ключа на закрытие

);

**Параметры**:

hKey Идентифицирует открытый ключ для закрытия.

**Возвращаемые значения**:

Если функция успешна, возвращается значение ERROR\_SUCCESS.

При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.

Дескриптор указанного ключа не должен использоваться после того как он был закрыт, потому что он больше не действителен.

Функция RegCloseKey не обязательно записывает информацию в реестр перед возвращением; может пройти более нескольких секунд, перед тем как кеш будет скинут на жесткий диск. Если приложению нужно срочно записать информацию реестра на жесткий диск, оно может использовать функцию RegFlushKey. Тем не менее, RegFlushKey использует много системных ресурсов и должна вызываться только тогда, когда необходимо.

**Функция RegDeleteKey** удаляет указанный ключ. Эта функция не может удалить ключ, которяй является подключем.

**Синтаксис**:

LONG RegDeleteKey(

HKEY hKey, // дескриптор открытого окна

LPCTSTR lpSubKey // адрес имени подключа на удаление

);

**Параметры**:

hKey Идентифицирует уже открытый ключ или один из предопределенных дескрипторов:

HKEY\_CLASSES\_ROOT

HKEY\_CURRENT\_USER

HKEY\_LOCAL\_MACHINE

HKEY\_USERS

HKEY\_CURRENT\_CONFIG

Ключ, указанный в параметре lpSubKey должен быть подключем ключа, указанного в hKey.

lpSubKey Адрес нульзаканчивающейся строки, указывающей имя ключа на удаление. Этот параметр не может быть NULL, и указанный ключ не должен иметь подключей.

**Возвращаемые значения**:

Если функция успешна, возвращается значение ERROR\_SUCCESS.

При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.

Если функция успешна, то RegDeleteKey удаляет указанный ключ из реестра. Вместе с ключем удаляются все его значения.

Для открытия ключа используйте функцию RegCreateKeyEx и RegOpenKeyEx. Не используйте функцию RegCreateKey или RegOpenKey.

**Функция RegCreateKeyEx** создает указанный ключ. Если ключ уже существует в реестре, то функция открывает его. Эта функция оставлена для совместимости с Windows версии 3.1. Приложения для Win32 должны использовать функцию RegCreateKeyEx.

**Синтаксис**:

LONG RegCreateKeyEx(

HKEY hKey, // дескриптор открытого ключа

LPCTSTR lpSubKey, // адрес имени подключа

DWORD Reserved, // зарезервировано

LPTSTR lpClass, // адрес строки класса

DWORD dwOptions, // флаг особых опций

REGSAM samDesired, // желаемый доступ безопасности

LPSECURITY\_ATTRIBUTES lpSecurityAttributes, // адрес структуры

// ключа безопасности

PHKEY phkResult, // адрес буфера для открытого ключа

LPDWORD lpdwDisposition // адрес буфера характерного значения

);

Параметры:

|  |  |
| --- | --- |
| hKey | Идентифицирует уже открытый ключ или один из предопределенных дескрипторов:  HKEY\_CLASSES\_ROOT HKEY\_CURRENT\_USER HKEY\_LOCAL\_MACHINE HKEY\_USERS HKEY\_CURRENT\_CONFIG  Ключ, открытый или созданный функцией RegCreateKeyEx, являются подключами ключа, указанного в параметре hKey. |
| lpSubKey | Адрес нульзаканчивающейся строки, определяющей имя подключа, который эта функция создает или открывает. Указанный подключ должен быть подключем ключа, указанного в параметре hKey. Этот подключ не должен начинатся с знака обратной черты ('\'). Этот параметр не должен быть равен NULL. |
| Reserved | Зарезервирован; должен быть нулевым. |
| lpClass | Адрес нульзаканчивающейся строки, определяющей класс (тип объекта) этого ключа. Этот параметр игнорируется если ключ уже существует. |
| dwOptions | Определяет особые опции для ключа. Этот параметр может быть одним из следующих значений:   |  |  | | --- | --- | | Значение: | Назначение: | | REG\_OPTION  \_NON\_VOLATILE | Это значение по умолчанию. При перезагрузке системы значение этого ключа сохраняется, т.е. информация сохраняется в файле, а не в памяти. | | REG\_OPTION  \_VOLATILE | Windows NT/2000/XP: Информация сохраняется в памяти и не сохраняется при перезагрузке системы. Функция RegSaveKey не сохраняет такие ключи. Этот флаг игнорируется, если ключ уже существует. Это значение игнорируется в Windows 95. | | REG\_OPTION  \_BACKUP\_RESTORE | Windows NT/2000/XP: Если установлен этот флаг, то функция игнорирует параметр samDesired и пытается открыть ключ с требуемым доступ для резервного сохранения или востановления ключа. Если вызывающий поток имеет включенную привилегию SE\_BACKUP\_NAME, ключ открывается с доступом ACCESS\_SYSTEM\_SECURITY и KEY\_READ. Если вызывающий поток имее включенную привилегию SE\_RESTORE\_NAME, ключ открывается с доступом ACCESS\_SYSTEM\_SECURITY и KEY\_WRITE. Если включены обе привилегии, ключ будет иметь комбинированный доступ к обоим привилегиям. В Windows 95 этот параметр игнорируется. | |
| samDesired | Определяет маску доступа, которая определяет желаемый доступ безопасности для нового ключа. Этот параметр может быть комбинацией из следующих значений:   |  |  | | --- | --- | | Значение: | Назначение: | | KEY\_ALL  \_ACCESS | Комбинация из KEY\_QUERY\_VALUE, KEY\_ENUMERATE\_SUB\_KEYS, KEY\_NOTIFY, KEY\_CREATE\_SUB\_KEY, KEY\_CREATE\_LINK, и KEY\_SET\_VALUE. | | KEY\_CREATE  \_LINK | Права создавать символические ссылки. | | KEY\_CREATE  \_SUB\_KEY | Права создавать подключи. | | KEY\_ENUMERATE  \_SUB\_KEYS | Права перечислять подключи. | | KEY\_EXECUTE | Права на доступ ко чтению. | | KEY\_NOTIFY | Права на изменение нотификации. | | KEY\_QUERY  \_VALUE | Права запрашивать данные подключей. | | KEY\_READ | Комбинация из KEY\_QUERY\_VALUE, KEY\_ENUMERATE\_SUB\_KEYS, и KEY\_NOTIFY. | | KEY\_SET\_  VALUE | Права устанавливать данные подключей. | | KEY\_WRITE | Комбинация из KEY\_SET\_VALUE и KEY\_CREATE\_SUB\_KEY. | |
| lpSecurityAttributes | Адрес структуры SECURITY\_ATTRIBUTES которая определяет: может ли возвращенный дескриптор наследоваться дочерними процессами. Если lpSecurityAttributes равен NULL, дескриптор не может наследоваться.  Windows NT/2000/XP: член lpSecurityDescriptor структуры определяет дескриптор безопасности для нового ключа. Если lpSecurityAttributes равен NULL, ключ получает дескриптор безопасности по умолчанию.  Windows 95/98/Me: член lpSecurityDescriptor структуры игнорируется. |
| phkResult | Адрес переменной, куда будет возвращен дескриптор,открываемого или создаваемого ключа. |
| lpdwDisposition | Адрес переменной, куда будет возвращен один из следующих значений:   |  |  | | --- | --- | | Значение: | Назначение: | | REG\_CREATED\_NEW\_KEY | Ключ не существует и был создан. | | REG\_OPENED\_EXISTING\_KEY | Ключ существует и был запросто открыт без изменений. | |

**Возвращаемые значения**:

Если функция успешна, возвращается значение ERROR\_SUCCESS.

При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.

Ключ, который создает функция RegCreateKeyEx, не имеет значений. Приложение может использовать функцию RegSetValue или RegSetValueEx для установки значений ключей.

Ключ, определенный в параметре hKey, должен быть открытым с доступом KEY\_CREATE\_SUB\_KEY. Для открытия ключа используйте функцию RegCreateKeyEx или RegOpenKeyEx.

Приложение не может создавать ключ под HKEY\_USERS или HKEY\_LOCAL\_MACHINE.

Приложение может использовать RegCreateKeyEx для временного закрытия части реестра. Когда закрывающий процесс создает новый ключ, он получает характерное значение REG\_CREATED\_NEW\_KEY, указывающее на то, он владелец закрытой части реестра. Другой процесс пытается создать тот-же ключ и получает характерное значение REG\_OPENED\_EXISTING\_KEY, указывающее на то, что другой процесс владеет закрытой частью.

**Функция RegOpenKey** открывает указанный ключ. Эта функция оставленя для совместимости с Windows версии 3.1. Приложения для Win32 должны использовать функцию RegOpenKeyEx.

**Синтаксис**:

LONG RegOpenKey(

HKEY hKey, // дескриптор открытого ключа

LPCTSTR lpSubKey, // адрес имени открываемого подключа

PHKEY phkResult // адрес дескриптора открытого ключа

);

Параметры:

hKey Идентифицирует открытый в текущий момент ключ или один из следующих предопределенных значений дескрипторов:

HKEY\_CLASSES\_ROOT

HKEY\_CURRENT\_USER

HKEY\_LOCAL\_MACHINE

HKEY\_USERS

HKEY\_CURRENT\_CONFIG

Ключ, открытый функцией RegOpenKey, является подключем ключа, указанного в hKey.

lpSubKey Адрес нульзаканчивающейся строки, содержащей имя ключа на открытие. Этот ключ должен быть подключем ключа, указанного в параметре hKey. Если этот параметр равен NULL или адрес указывает на пустую строку, функция возвращает тот-же дескриптор, который ей был дан в параметре hKey.

phkResult Адрес переменной, в которую возвращается дескриптор открытого ключа.

**Возвращаемые значения**:

Если функция успешна, возвращается значение ERROR\_SUCCESS.

При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.

Функция RegOpenKey использует маску из доступа безопасности по умолчанию для открытия ключа. Если открываемый ключ требует другую маску, функция завершается ошибкой и возвращает ERROR\_ACCESS\_DENIED. Приложение может использовать функцию RegOpenKeyEx для указания маски доступа в такой ситуации.

В отличии от RegCreateKey, функция RegOpenKey не создает указанный ключ если такой ключ не существует в базе данных.

**Функция RegCreateKey** создает указанный ключ. Если ключ уже существует в реестре, то функция открывает его. Эта функция оставлена для совместимости с Windows версии 3.1. Приложения для Win32 должны использовать функцию RegCreateKeyEx.

**Синтаксис**:

LONG RegCreateKey(

HKEY hKey, // дескриптор открытого окна

LPCTSTR lpSubKey, // адрес имени открываемого подключа

PHKEY phkResult // адрес буфера для открытого дескриптора

);

Параметры:

hKey Идентифицирует уже открытый ключ или один из предопределенных дескрипторов:

HKEY\_CLASSES\_ROOT

HKEY\_CURRENT\_USER

HKEY\_LOCAL\_MACHINE

HKEY\_USERS

HKEY\_CURRENT\_CONFIG

Ключ, открытый или созданный функцией RegCreateKey, являются подключами ключа, указанного в параметре hKey.

lpSubKey Адрес нульзаканчивающейся строки, определяющей имя подключа, который эта функция создает или открывает. Указанный подключ должен быть подключем ключа, указанного в параметре hKey.

Если hKey – это один из предопределенных ключей, то lpSubKey должен быт NULL. В этом случае, возвращаемый дескриптор в phkResult, это тот-же дескриптор, который передается функции в hKey.

phkResult дрес переменной, куда будет возвращен дескриптор,открываемого или создаваемого ключа.

**Возвращаемые значения**:

Если функция успешна, возвращается значение ERROR\_SUCCESS.

При ошибке, возвращаемое значение – это код ошибки, объявленный в WINERROR.H. Вы можете использовать функцию FormatMessage с флагом FORMAT\_MESSAGE\_FROM\_SYSTEM для получения общего описания ошибки.

Приложение может использовать функция RegCreateKey для создания нескольких ключей одновременно. Например, приложение может создать подключ четырех уровней вглубь, если они указаны в строке lpSubKey в таком виде: subkey1\subkey2\subkey3\subkey4

Ключ, который указывается в параметре hKey, должен быть открыть с доступом KEY\_CREATE\_SUB\_KEY (доступ KEY\_WRITE вкючает в себя и доступ KEY\_CREATE\_SUB\_KEY).

Если lpSubKey ссылается на пустую строку, функция открывает и потом возвращает обратно ключ, указанный в параметре hKey.

**Функция FindFirstFile** ищет каталог файла или подкаталог, название которого соответствует указанному имени файла.

**Синтаксис**

HANDLE FindFirstFile(

LPCTSTR lpFileName, // имя файла

LPWIN32\_FIND\_DATA lpFindFileData // буфер данных

);

**Параметры**

lpFileName

[in] Указатель на символьную строку с нулем в конце, которая определяет правильный каталог или путь и имя файла, которое может содержать символы подстановки (\* и ?). Если символьная строка заканчивается символом подстановки, точкой или именем каталога, пользователь должен иметь доступ к корневому каталогу и всем вложенным в него каталогам на пути.

lpFindFileData

[out] Указатель на структуру WIN32\_FIND\_DATA, которая получает информацию о найденном файле или вложенном каталоге.

**Возвращаемые значения**

Если функция завершается успешно, возвращаемое значение - дескриптор поиска, используемый в последующем вызове функции FindNextFile или FindClose.

Если функция завершается ошибкой, возвращаемое значение - INVALID\_HANDLE\_VALUE.

**Функция GetDiskFreeSpaceEx** возвращает информацию относительно количества места на диске. Эта информация включает в себя количество свободного места для текущего пользователя, общий объем диска и количество свободного места на диске (все байтах)

**Синтаксис**

Declare Function GetDiskFreeSpaceEx Lib "kernel32" Alias "GetDiskFreeSpaceExA" (ByVal lpRootPathName As String, \_

lpFreeBytesAvailableToCaller As Currency, \_

lpTotalNumberOfBytes As Currency, \_

lpTotalNumberOfFreeBytes As Currency) As Long

**Параметры**

lpDirectoryName Имя директории на диске, информацию относительно которой нужно узнать. Это может быть имя корневой директории или любая другая директория. Если аргумент установлен в Null, то возвращается информация о текущем диске

lpFreeBytesAvailableToCaller Безнаковое 64-разрядное целое число, содержащее количество свободного места в байтах для текущего пользователя

lpTotalNumberOfBytes Безнаковое 64-разрядное целое число, содержащее объем диска в байтах

lpTotalNumberOfFreeBytes Безнаковое 64-разрядное целое число, содержащее общее количество свободного места в байтах

**Возвращаемые значения**

В случае ошибки функция возвращает 0 (используйте GetLastError для получения кода ошибки). В успешном случае возвращается ненулевое значение.

Функция archivate – Собирает все файлы приложения, которое необходимо установить и помещает их в архив с расширением .bin

Функция FreeSpace – Осуществляет анализ свободного места на диске

Функция ReadInfo – Осуществляет получение системных данных путем сбора информации из реестра

Функция Uninst()– Осуществляет размещение в списке деинсталлируемых программ

Функция link()– Осуществляет создание ярлыка и ссылок в меню «Пуск»

Функция reg() - Осуществляет настройку системного реестра

Функция unzip - Осуществляет распаковку архива и размещение в каталоге установленных программ

# Тестирование

Тестирование приложения проводилось на двух операционных системах Microsoft Windows 10 Pro (информация о системе приведена в начале отчета) и Windows 7:

C:\Users\Елена>systeminfo

Имя узла: HELEN\_COMP2

Название ОС: Microsoft Windows 7 Профессиональная

Версия ОС: 6.1.7601 Service Pack 1 сборка 7601

Изготовитель ОС: Microsoft Corporation

Параметры ОС: Изолированная рабочая станция

Сборка ОС: Multiprocessor Free

Зарегистрированный владелец: -

Зарегистрированная организация:

Код продукта: 00371-OEM-8992671-00524

Дата установки: 12.01.2016, 17:10:09

Время загрузки системы: 17.01.2017, 3:52:33

Изготовитель системы: ASUSTeK Computer Inc.

Модель системы: K53SJ

Тип системы: x64-based PC

Процессор(ы): Число процессоров - 1.

[01]: Intel64 Family 6 Model 42 Stepping 7 GenuineIntel ~2301 МГц

Версия BIOS: American Megatrends Inc. K53SJ.208, 14.02.2011

Папка Windows: C:\Windows

Системная папка: C:\Windows\system32

Устройство загрузки: \Device\HarddiskVolume2

Язык системы: ru;Русский

Язык ввода: ru;Русский

Часовой пояс: (UTC+03:00) Волгоград, Москва, Санкт-Петербург

Полный объем физической памяти: 4 008 МБ

Доступная физическая память: 2 017 МБ

Виртуальная память: Макс. размер: 8 014 МБ

Виртуальная память: Доступна: 5 968 МБ

Виртуальная память: Используется: 2 046 МБ

Расположение файла подкачки: C:\pagefile.sys

Для запуска приложения в режиме архивации файлов, необходимо запустить его из окна командной стоки указав в качестве параметров ключ (-а) и путь до директории с программой, которую необходимо инсталлировать.

C:\Users\Helen\Documents\Visual Studio 2013\Projects\InstallerArch\Debug>Installer.exe -a I:\1

---Welcome to the Archivator !---

Start Archivating ...

MyTestApplication1.exe

Pyton

Archivating ...

code.py

Archivating ended sucsess !

В результате выполнения программы в режиме архиватора, был получен архив с файлами.

После чего был запущен исполняемый файл инсталлятора (с правами администратора):

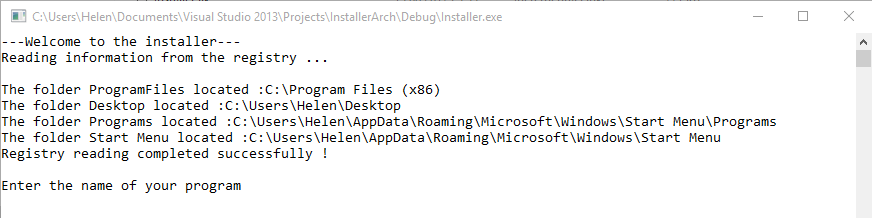


Рис.2 Запуск инсталлятора

Программа выводит сведения о расположении директорий Program Files и Program Files (x86), Рабочего стола, меню «Пуск». После чего, пользователю предлагается ввести название программы и происходит процесс установки.

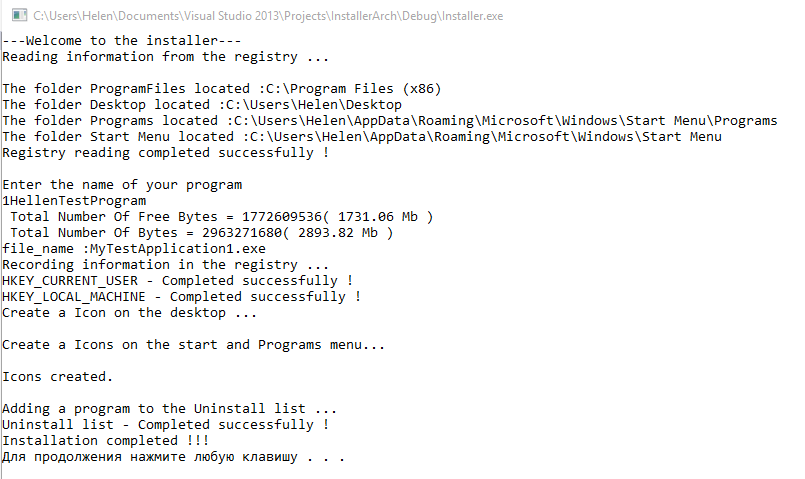


Рис.3 Выполнение программы

Установка успешна завершена.

На Рабочем столе появился новый ярлык установленного приложения.

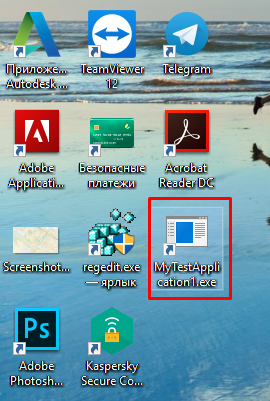


Рис.4 Ярлык

Также ссылки появились в каталоге «Программы», «Главное меню» и «Недавно добавленные приложения». Что показано на рисунках 5-7.

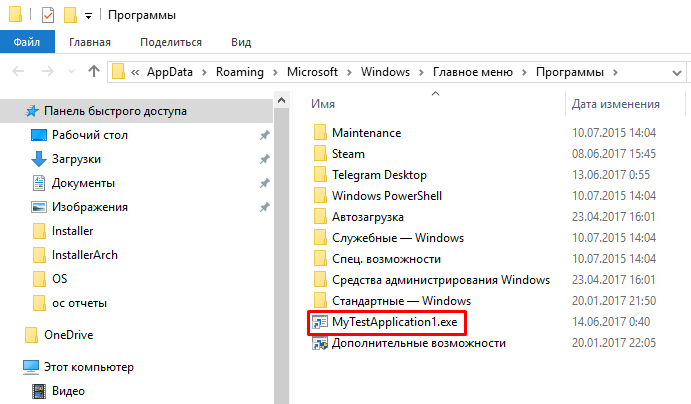


Рис.5 каталог «Программы»

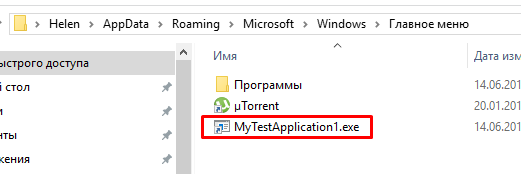


Рис.6 каталог «Главное меню»

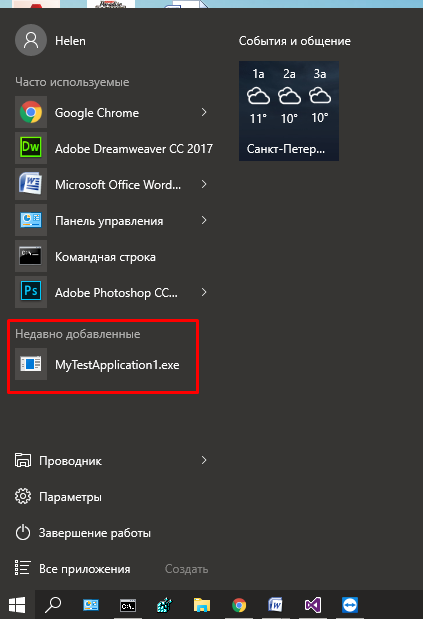


Рис.7 каталог «Недавно добавленные приложения»

В директории Program Files (x86), появился каталог нового приложения.

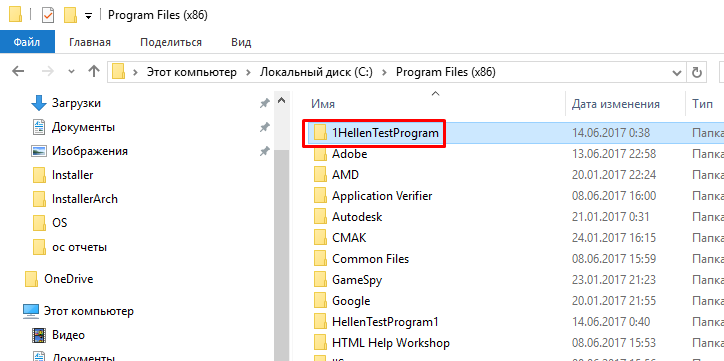


Рис.8 Program Files (x86)

Установленная программа, появилась в окне «Программы и компоненты» (Рис.9) в списке удаления или изменения программ.

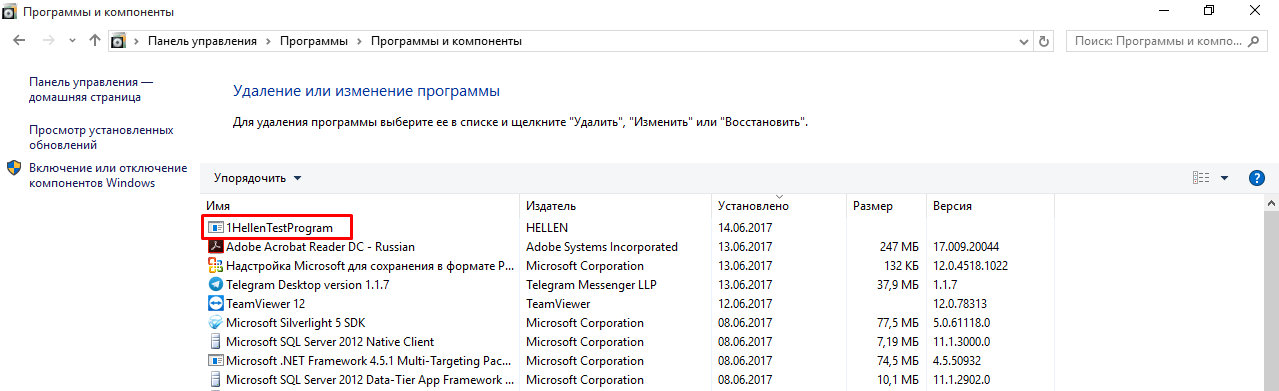


Рис.9 «Программы и компоненты»

Настройка системного реестра прошла успешно, о чем свидетельствуют появившиеся записи, настройки реестра показаны на рисунках 10-12.

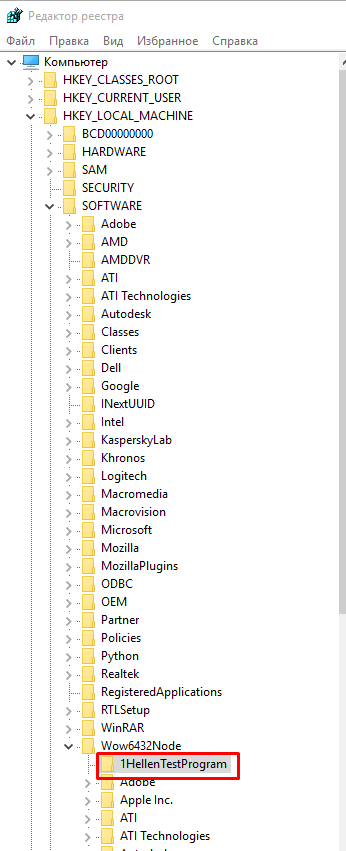


Рис.10 Раздел HKEY\_LOCAL\_MACHINE

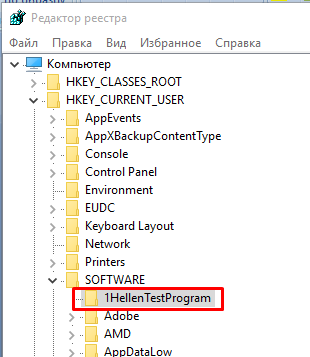


Рис.11 Раздел HKEY\_CURRENT\_USER

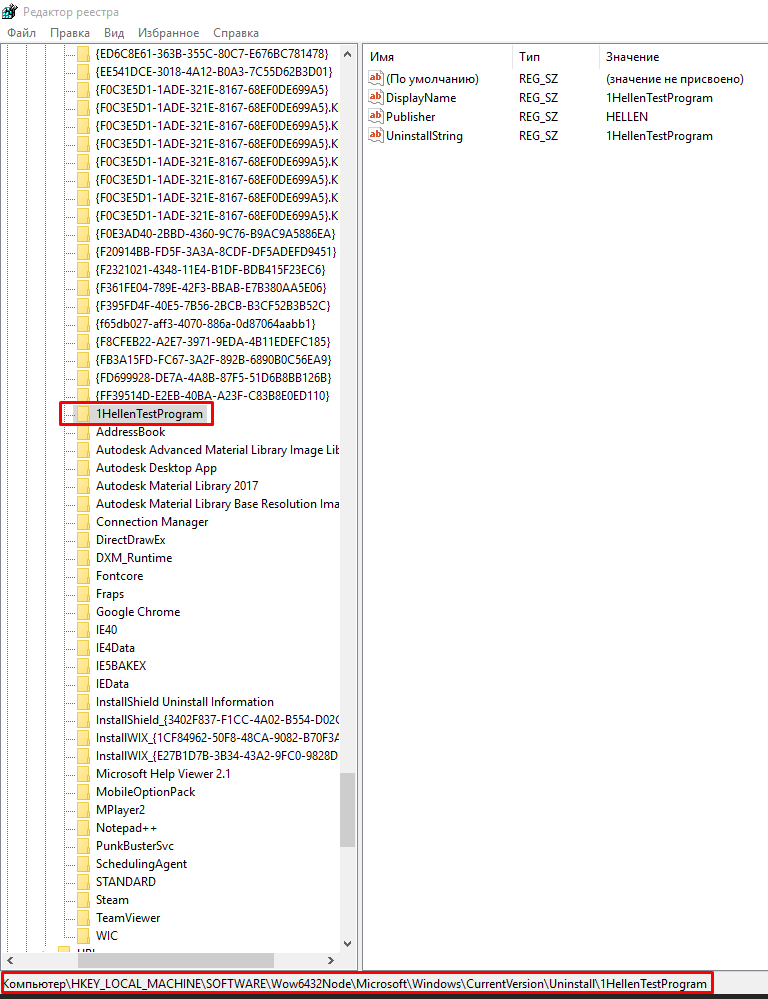


Рис.12 Раздел Uninstall

При тестировании приложения в Microsoft Windows 7 произошли аналогичные изменения, программа была успешно инсталлирована.

# Заключение

В данном курсовом проекте было разработано приложение, осуществляющее установку программ. Были рассмотрены существующие инструментальные средства для создания инсталляторов. Получены навыки работы с реестром Windows.

Реализованное приложение осуществляет получение системных данных путем сбора информации из реестра, получение данных пользователя, создание архива, проверку свободного места на диске, распаковку, настройку системного реестра, создание ярлыка и ссылок в меню «Пуск», размещение в каталоге установленных программ, размещение в списке деинсталлируемых программ

# Список использованных источников

1. Сведения о реестре Windows для опытных пользователей [Электронный ресурс] URL: <https://support.microsoft.com/ru-ru/help/256986/windows-registry-information-for-advanced-users>
2. w32api - Справочник по Windows API [Электронный ресурс] URL: <http://w32api.ru/functions/RegOpenKeyEx.html>
3. XServer.ru - бесплатная, виртуальная, электронная, Интернет библиотека онлайн [Электронный ресурс] URL: <http://www.xserver.ru/computer/sredaprogr/bd/10/7.shtml>
4. Национальная библиотека им. Н. Э. Баумана Bauman National Library [Электронный ресурс] URL: <http://ru.bmstu.wiki/>
5. Microsoft Developer Network Класс Registry [Электронный ресурс] URL: <https://msdn.microsoft.com/ru-ru/library/microsoft.win32.registry(v=vs.110).aspx>
6. MSDN — структура реестра, программирование реестра [Электронный ресурс] URL: <https://msdn.microsoft.com/en-us/library/ms724871.aspx>