MagicKeys – это программа с открытым исходным кодом для обеспечения доступности пользовательского интерфейса с обширными возможностями автоматизации. Данное ПО позволяет взаимодействовать с элементами GUI, недоступными для программ чтения с экрана при помощи псевдо-интерфейса, функций автоматизации действий пользователя, оптического распознавания текста с экрана.

Базовый функционал MagicKeys возможно расширить при помощи дополнений, создание которых доступно любому пользователю.

Не проходите мимо, перед тем как приступить к написанию своего расширения для MagicKeys, категорически рекомендуется ознакомится с несколькими правилами. Соблюдая их, вы сэкономите себе не мало времени, побережёте нервы разработчика, и даже расстройство желудка вам будет не страшно.

* /\*
* Перед тем как что-то написать, убедитесь, что это нельзя реализовать проще, я серьёзно, если в вашем плагине много кода который палит из пушки по воробьям, то дело плохо и пора бы поспать.
* \*/
* Формат плагинов для MagicKeys является декларативным, то есть писать только так, и никак иначе. Да, в нём существуют некоторые нестандартные подходы, однако они оправдали своё существование на практике.
* Когда вы напишите свой первый плагин, у вас возникнет закономерный вопрос: "Зачем столько одинаковых, абсолютно повторяющихся данных?". Когда вы напишите свой десятый плагин, вы будете счастливы, что эти данные повторяются, и позволяют вам добиться максимальной расширяемости. //Внезапно окажется, что данные не повторяются, а имеют глубоко динамическую природу.
* Если вы будете писать расширения не так как того требует документация, вы получите ошибки, очень много некрасивых ошибок. В целях оптимизации в ядре MagicKeys довольно мало проверок на синтаксис, поэтому ваш код исключительно на вашей совести.
* Плагины в MagicKeys чувствительны к регистру, то есть VUI и vui не одно и тоже.
* Имена директорий тоже чувствительны к регистру.
* Для изображений используйте только формат BMP.

Ну, вот и вводная закончилась, все советы даны и теперь можно приступать к написанию своих плагинов.

В MagicKeys реализованы два уровня API для создания плагинов.

API высокого уровня. Предназначен для создания простых плагинов, имеет не сложный синтаксис ini файлов, не требует навыков программирования и дополнительных средств разработки.

API низкого уровня. Используется для сложных плагинов, основной функционал компилируется в библиотеку классов dll при помощи Visual studio. Требуются базовые знания программирования на C#.

Рассмотрим структуру файлов для плагинов. Папка Plugins является корневой для них, в ней располагаются папки самих плагинов. Имена этих папок должны соответствовать названиям программ, с которыми они позволяют взаимодействовать. Если плагин содержит несколько слов в названии, то они пишутся слитно, каждое слово с заглавной буквы.

Так же здесь присутствуют папки, название которых содержит имя плагина, например “Kontakt”, + Lib. В подобных папках располагаются дочерние расширения основного плагина. Подстрока Lib в конце имени папки в данном случае является обязательной. То есть если вы создаёте дочернее расширение для плагина, его необходимо размещать в папке с названием, “Имя плагина+Lib”.

Например OmnisphereLib.

В папке Plugins так же находится файл Hosts.ini. Он содержит обобщённые подстроки для заголовков и классов, предназначенных для обнаружения нужных окон. Подстроки разделяются вертикальной чертой “|”. Этот файл является служебным, изменяйте его с большой осторожностью.

Рассмотрим структуру стандартного плагина высокого уровня. На данный момент таковым является плагин PlugSound.

Папка Images – для изображений в формате bmp. Другие форматы не поддерживаются.

Папка VUI – предназначена для файлов виртуальных псевдо-интерфейсов в формате vui – virtual user interface и файлов интерпретатора в формате vuf – virtual user functions.

Файл Manifest.ini – Содержит в себе основную информацию о плагине и точки входа для его загрузки.

Плагины низкого уровня отличаются только наличием библиотеки классов dll, в остальном они идентичны плагинам высокого уровня.

Описание Manifest.ini.

Секция Info содержит основную информацию о плагине.

[Info]

Name=Основное имя плагина. Например Kontakt

Author=Имя автора плагина

Mail=Адрес электронной почты автора

После секции Info следуют секции точек входа в плагин. Их может быть несколько. Названия точек входа могут быть произвольными, однако Их формирование по следующим правилам позволит избежать ошибок и ложных срабатываний.

Имя разработчика или компании + имя приложения + имя дочернего окна, для которого действует точка входа.

Например, в Kontakt 6 существует основное окно и несколько дочерних. Имя точки входа для основного окна может выглядеть так - “NativeInstrumentsKontaktGeneral”. Имя точки входа для дочернего окна – “NativeInstrumentsKontaktProgress”.

Заполним точку входа для основного окна Kontakt.

[NativeInstrumentsKontaktGeneral]

WTitle – Содержит подстроку из заголовка окна, например “Progress.

WClass – Класс окна, например #32770.

Если ключи WTitle и WClass не указаны, то используются подстроки из файла Hosts.ini.

В данном случае основное окно Kontakt содержит подстроки указанные в Hosts.ini, поэтому эти поля не указываются.

PClass=Kontakt

При создании плагина высокого уровня, поле PClass должно иметь значение “MagicKeys”.

При создании плагина низкого уровня, PClass имеет название основного класса в библиотеке dll. В данном случае плагин является низкоуровневым, поэтому в поле указано Kontakt.

BClass=None

Поле BClass представляет имя родительского плагина. Если вы создаёте дочерний плагин для библиотеки Kontakt, то это поле должно иметь значение Kontakt.

Если плагин не имеет родителя, тогда значение BClass должно быть None. Для удобного понимания BClass можно сравнить с отчеством.

VUI=Kontakt

Имя виртуального интерфейса без расширения для данной точки входа, который находится в папке VUI. Может быть произвольным.

PluginName=Kontakt

Имя папки с плагином. Если плагин низкого уровня, то его библиотека классов dll должна иметь такое же имя. Например, папка для плагина Native Instruments Kontakt называется Kontakt, а библиотека dll называется Kontakt.dll.

Module=Kontakt

Подстрока для обнаружения модуля, создающего окно. В данном случае модуль называется Kontaktx64.dll, поэтому для обнаружения достаточно подстроки Kontakt.

В итоге Manifest.ini со всеми точками входа для Kontakt выглядит следующим образом:

[Info]

Name=Native instruments Kontakt

Author=Viruzober

Mail=Den-viruzober@yandex.ru

[NativeInstrumentsKontaktGeneral]

PClass=Kontakt

BClass=None

VUI=Kontakt

PluginName=Kontakt

Module=Kontakt

[NativeInstrumentsKontaktContentMissing]

WTitle=Content Missing

WClass=#32770

PClass=Kontakt

BClass=None

VUI=ConntentMissing

PluginName=Kontakt

Module=Kontakt

[NativeInstrumentsKontaktProgress]

WTitle=Progress

WClass=#32770

PClass=Kontakt

BClass=None

VUI=Progress

PluginName=Kontakt

Module=Kontakt

Виртуальный интерфейс и функции

Папка VUI содержит файлы виртуальных интерфейсов в формате vui, и файлы интерпретатора MagicKeys в формате vuf.

Файлы vui представляют собой описание элементов графического интерфейса. В основном они соответствуют его логике, то есть виртуальные объекты располагаются в том же порядке, в котором они отображаются на экране. Однако не стоит следовать только этому. Представим ситуацию, вам необходимо щёлкнуть мышкой в пункт недоступного контекстного меню, вы можете создать объект меню открывающий его, а затем загружающий интерфейс с самими пунктами. Это оправдано делать только в случае, если вам действительно необходимы все пункты меню. Например, если из 10 пунктов вам нужно только 3, то лучше создать три виртуальных объекта, выполняющих по два клика. Ищите золотую середину, излишнее упрощение или усложнение виртуального интерфейса сделает неудобным его в первую очередь для вас.

Рассмотрим стандартный vui файл.

Как и Manifest.ini , любой vui файл начинается с обязательной секции Info. Данные секции во многом похожи. Рассмотрим vui файл плагина высокого уровня PlugSound. Для плагинов низкого уровня содержимое идентично.

[Info]

VUI=PlugSound ; Название интерфейса без расширения.

PClass=MagicKeys ; Основной класс плагина, так как это плагин высокого уровня, его основной класс MagicKeys.

BClass=None ; Данный плагин не имеет родителя, поэтому значение этого поля None.

PluginName=PlugSound ; Имя плагина. Такое же, как и имя его папки.

Loader=None ; Поле Loader представляет способ предварительной загрузки плагина, это действия, которые выполняются перед передачей управления пользователю. Например, в плагине Omnisphere перед началом управления, необходимо закрыть splash screen. В данном случае для PlugSound входные действия не требуются. Поэтому значение установлено в None. Как создать и использовать предварительный загрузчик будет описано далее в этом руководстве.

[Keys]

Секция Keys является необязательной. В ней описана связь горячих клавиш с функциями. Здесь действуют следующие правила для их назначения.

Все клавиши начинаются с заглавных букв, !shift и Shift не одно и тоже. Модификаторы с клавишами связываются через знак плюса “+”. Например, Ctrl+Home.

Поддерживаются 4 модификатора со следующим приоритетом обработки: Alt, Ctrl, Shift, Win.

M=Menu

В данном случае клавиша M назначается на функцию, которая открывает меню приседов в PlugSound. Создание и использование функций будет рассмотрено далее в этом руководстве.

После необязательной секции Keys, следуют секции виртуальных объектов. Имена секций указываются строго от 1 и по возрастанию. [1], [2], [3], [4], [5] …

Так же возможно не указывать секции объектов, а использовать для взаимодействия только горячие клавиши в секции Keys.

[1] – Секция объекта с порядковым номером от 1 по возрастанию.

Text=Bank and patch menu ; Текст, который будет произнесён при перемещении виртуального навигатора на объект.

ObjectType=Button ; Тип объекта, который произносится после текста. Может представлять следующие значения: Button, Slider, CheckBox, RadioBoxGroup, Progress, Label.

Типизация объектов является не строгой, и может быть произвольной. Данный ключ используется для речевой подсказки. Это позволяет более точно представить визуальную составляющую графического интерфейса. Поэтому необходимо использовать короткие и наиболее информативные типы. Если объект выполняет несколько действий, следует указывать тип первого объекта, с которым он взаимодействует в визуальном интерфейсе.

Help=Открыть меню банков и патчей ; Текст справки по объекту, который произносится по нажатию клавиши F1.

AutoFunc=Menu ; Автоматически выполняемая функция при перемещении виртуального навигатора на объеккт. В данном случае не требуется, представлена как пример.

Func=Menu ; Функция, выполняемая при нажатии Enter на объекте.

Param=Next

Аргумент для передачи в функцию для плагинов низкого уровня. Используются преимущественно в сценариях навигации. Например, для переключения приседов в плагине nexus, для уменьшения повторяющихся частей кода, в функцию навигации по песетам передаётся аргумент Back или Next, на основе которого выполняется дальнейший сценарий. В данном случае этот ключ не используется, и должен быть опущен для плагинов высокого уровня.

Key=M ; Если для функции объекта существует аналог в секции Keys в виде горячей клавиши, необходимо указывать данный ключ для проговаривания. Если не назначена, тогда ключ опускается.

Пример готового виртуального интерфейса для плагина высокого уровня PlugSound.

[Info]

VUI=PlugSound

PClass=MagicKeys

BClass=None

PluginName=PlugSound

Loader=None

[Keys]

M=Menu

[1]

Text=Bank and patch menu

ObjectType=Button

Help=Открыть меню банков и патчей

Func=Menu

Функции плагинов высокого уровня

В папке VUI так же находятся файлы интерпретатора MagicKeys в формате vuf. Их синтаксис во многом похож на vui, однако имеет ряд отличий.

Рассмотрим стандартный файл функций для плагина высокого уровня PlugSound. Файлы функций не имеют обязательных секций, они содержат только описание функций. Имя каждой секции, это имя функции, в данном случае функция называется Menu.

[Menu] – Имя функции.

После имени функции следуют ключи, которые необходимо называть от 1 и по возрастанию, в данном случае функция выполняет один клик, который открывает меню банков и патчей.

Описание вызова функции из ядра MagicKeys.

Номер ключа от 1 и по возрастанию, равно, имя функции, вертикальная черта – “|”, аргументы через вертикальную черту.

1=MouseClick|Left|250|30|1|0|0|10

В данном случае выполнится 1 клик левой кнопкой мыши относительно указанного вызывающего модуля, по координатам X – 250, Y – 30, с максимальной скоростью движения курсора, без ожидания после наведения курсора и с ожиданием 10 миллисекунд перед отпусканием кнопки мыши. Подробное описание всех доступных функций содержится в соответствующем разделе руководства.

Конечный vuf файл для данного плагина.

[Menu]

1=MouseClick|Left|250|30|1|0|0|10

Создание и использование загрузчиков виртуальных интерфейсов

Загрузчики виртуальных интерфейсов предназначены для выполнения предварительных действий перед передачей управления пользователю. Например, в плагине Omnisphere перед использованием необходимо закрыть стартовый экран. Благодаря этому существенно сокращается объём кода для плагина. Каждый интерфейс может иметь свой собственный загрузчик, или не иметь его вообще. Для управления загрузчиком интерфейса в секции Info существует ключ Loader.

В зависимости от его значения, загрузчик выполняется в определённом контексте.

None – Загрузчик для данного интерфейса не используется.

Code – Это значение предназначено для плагинов низкого уровня, скомпилированных в библиотеку классов dll. При его указании выполняется метод из основного класса плагина с именем Имя интерфейса+Loader. Например, в плагине Kontakt для коррекции визуального интерфейса используется загрузчик “LibListLoader”. В данном случае “LibList” – это имя загружаемого интерфейса.

VUI – Используется в плагинах высокого уровня. Для создания загрузчика высокого уровня в папке VUI вашего плагина необходимо добавить следующие файлы.

Имя интерфейса+Load.vui – Данный файл содержит описание условий, при которых должен выполниться загрузчик.

Имя интерфейса+Load.vuf – Содержит описания загрузчиков для интерпретатора MagicKeys.

Рассмотрим файл условий для загрузчика высокого уровня.

[1] – Имя секции от 1 и по возрастанию.

Trigger=Img|Logo|true – Ключ Trigger представляет описание условия, при котором должен выполниться загрузчик. Поддерживается только условие поиска изображений на экране, в будущем возможно добавление других триггеров.

Img – Инициализирует триггер поиска изображения на экране, далее следует имя изображения в формате bmp, которое должно находиться в папке Images вашего плагина. В данном случае изображение называется Logo. Третий аргумент указывает на условие, при котором будет выполнен загрузчик.

true – Загрузчик выполнится если изображение найдено.

false – Загрузчик выполнится если изображение не найдено.

TimeOut=200 – Ожидание в миллисекундах между действиями загрузчика.

Action=1 – Указывает номер секции для выполнения, которая находится в файле интерпретатора имя интерфейса+Load.vuf.

Рассмотрим файл интерпретатора для загрузчика.

[1] – Имя секции от 1 и по возрастанию.

Далее следует описание сценария загрузчика. Оно соответствует файлам интерпретатора в формате vuf.

1=MouseClick|Left|135|65|2|0|0|10

Конечные файлы загрузчика выглядят следующим образом.

InterfaceNameLoad.vui

[1]

Trigger=Img|Logo|true

TimeOut=200

Action=1

InterfaceNameLoad.vuf

[1]

1=MouseClick|Left|135|65|2|0|0|10