**Разработка скрипта для автоматизации установки и настройки ПО в компьютерных аудиториях кафедры**

**Цель:** Исследовать и реализовать способ автоматизации массовой установки и первоначальной настройки прикладного программного обеспечения под ОС Windows для развертывания в компьютерных аудиториях.

**Выполненные задачи:**

1. **Исследование инструментов автоматизации:** Проанализированы различные подходы к автоматизации установки ПО в Windows. В качестве основного инструмента выбран менеджер пакетов **Chocolatey**, так как он позволяет управлять установкой, обновлением и удалением тысяч приложений из централизованного репозитория с помощью командной строки. Это более надежный и поддерживаемый метод по сравнению с прямым запуском .exe-файлов.
2. **Разработка скрипта установки:** Создан PowerShell-скрипт (Install-Software.ps1), который:
   * Проверяет и устанавливает менеджер пакетов Chocolatey (если он не установлен).
   * Последовательно устанавливает все указанные в задании программы из официальных репозиториев Chocolatey.
   * Реализует логику для установки специфичного ПО только для аудитории 313.
   * Настраивает Visual Studio Code, устанавливая необходимые расширения.
3. **Создание скрипта настройки VS Code:** Разработан отдельный скрипт (Configure-VSCode.ps1), который инсталлирует расширения для Python, C++, Docker, веб-технологий и других инструментов, что критически важно для учебного процесса.

**Листинг кода скрипта для автоматизации**

**Основной скрипт установки (Install-Software.ps1)**

<#

.SYNOPSIS

Скрипт автоматической установки ПО для учебных аудиторий кафедры.

.DESCRIPTION

Устанавливает необходимое ПО с помощью менеджера пакетов Chocolatey.

Требует запуска от имени администратора.

#>

# Требуем запуск с правами администратора

if (-not ([Security.Principal.WindowsPrincipal] [Security.Principal.WindowsIdentity]::GetCurrent()).IsInRole([Security.Principal.WindowsBuiltInRole] "Administrator")) {

Write-Error "Этот скрипт требует права администратора. Запустите PowerShell от имени администратора."

exit 1

}

# Функция для установки Chocolatey, если он отсутствует

function Install-Chocolatey {

Write-Host "Проверка установки Chocolatey..." -ForegroundColor Yellow

if (Get-Command choco -ErrorAction SilentlyContinue) {

Write-Host "Chocolatey уже установлен." -ForegroundColor Green

} else {

Write-Host "Установка Chocolatey..." -ForegroundColor Yellow

# Устанавливаем политику выполнения для текущего процесса

Set-ExecutionPolicy Bypass -Scope Process -Force

# Команда установки с официального сайта

[System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072

iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))

Write-Host "Chocolatey успешно установлен." -ForegroundColor Green

# Обновляем PATH для текущей сессии

$env:Path = [System.Environment]::GetEnvironmentVariable("Path","Machine") + ";" + [System.Environment]::GetEnvironmentVariable("Path","User")

}

}

# Функция для установки списка пакетов

function Install-Packages {

param(

[string[]]$PackageList

)

foreach ($package in $PackageList) {

Write-Host "Устанавливается: $package" -ForegroundColor Cyan

choco install $package -y --force

if ($LASTEXITCODE -eq 0) {

Write-Host "Успешно: $package" -ForegroundColor Green

} else {

Write-Host "Ошибка при установке: $package" -ForegroundColor Red

}

Write-Host "---" -ForegroundColor Gray

}

}

# Основной блок выполнения

# 1. Устанавливаем Chocolatey

Install-Chocolatey

# 2. Список ПО для установки во всех аудиториях

$CommonSoftware = @(

"vscode", # Visual Studio Code

"docker-desktop", # Docker

"pycharm-community", # PyCharm Community Edition

"git", # Git

"github-desktop", # GitHub Desktop

"maxima", # Maxima

"knime", # KNIME Analytics Platform

"gimp", # GIMP

"python", # Python (устанавливается отдельно, расширения для VSCode - в другом скрипте)

"rust", # Rust (аналогично)

"msys2", # MSYS2 (включает UCRT64)

"zettlr", # Zettlr

"miktex", # MiKTeX

"texstudio", # TeXstudio

"anaconda3", # Anaconda

"far", # Far Manager

"sumatrapdf", # SumatraPDF

"googlechrome", # Chrome

"flameshot", # Flameshot

"wsl2", # Windows Subsystem for Linux

"qalculate", # Qalculate!

"arc", # Arc Browser

"7zip", # 7-Zip

"firefox", # Firefox

"yandex-browser-stable",# Yandex Browser

"microsoft-edge" # Edge

# Примечание: Julia, Yandex.Telemost, Sber Jazz пока не имеют официальных пакетов в Chocolatey,

# их установку нужно добавить вручную или найти альтернативные методы.

)

# 3. Устанавливаем общее ПО

Write-Host "Начинается установка общего программного обеспечения..." -ForegroundColor Magenta

Install-Packages -PackageList $CommonSoftware

# 4. Запрашиваем номер аудитории для установки специфичного ПО

$RoomNumber = Read-Host "Введите номер аудитории (например, 313). Если аудитория не 313, нажмите Enter"

if ($RoomNumber -eq "313") {

Write-Host "Установка ПО для аудитории 313..." -ForegroundColor Magenta

$SpecificSoftware313 = @(

"ramus-educational", # Ramus Educational (предполагается, что пакет существует)

"aris-express", # ARIS EXPRESS

"archi" # Archi

)

# Установка специфичного ПО

Install-Packages -PackageList $SpecificSoftware313

} else {

Write-Host "Установка ПО для аудитории 313 пропущена." -ForegroundColor Yellow

}

# 5. Настройка WSL (установка дистрибутивов Ubuntu)

Write-Host "Настройка WSL..." -ForegroundColor Yellow

wsl --install -d Ubuntu-22.04 --web-download

wsl --install -d Ubuntu-24.04 --web-download

Write-Host "Основной процесс установки ПО завершен." -ForegroundColor Green

Write-Host "Перейдите к настройке VSCode, запустив скрипт Configure-VSCode.ps1" -ForegroundColor Yellow

**Скрипт настройки Visual Studio Code (Configure-VSCode.ps1)**

<#

.SYNOPSIS

Скрипт для установки расширений Visual Studio Code.

.DESCRIPTION

Устанавливает необходимые для учебного процесса расширения VSCode.

Может быть запущен без прав администратора.

#>

Write-Host "Начало установки расширений для Visual Studio Code..." -ForegroundColor Magenta

# Список расширений для установки

$VSCodeExtensions = @(

"ms-python.python", # Python

"ms-vscode.cpptools", # C/C++

"ms-azuretools.vscode-docker", # Docker

"ms-vscode.PowerShell", # PowerShell (часто нужен)

"formulahendry.code-runner", # Code Runner (удобство для студентов)

"ms-vscode.live-server", # Live Server для HTML/CSS/JS

"ecmel.vscode-html-css", # Подсветка HTML/CSS

"xabikos.JavaScriptSnippets", # Сниппеты JS

"esbenp.prettier-vscode", # Prettier (форматирование кода)

"ms-vscode-remote.remote-wsl", # WSL

"ms-vscode-remote.remote-ssh", # SSH

"github.copilot", # GitHub Copilot (если есть лицензия)

"ms-vscode.vscode-github", # GitHub Integration

"julialang.language-julia", # Julia (должен быть установлен сам язык)

"rust-lang.rust-analyzer", # Rust (должен быть установлен сам язык)

"bungcip.better-toml" # TOML поддержка (для Rust)

)

# Устанавливаем каждое расширение

foreach ($extension in $VSCodeExtensions) {

Write-Host "Устанавливается расширение: $extension" -ForegroundColor Cyan

code --install-extension $extension

if ($LASTEXITCODE -eq 0) {

Write-Host "Успешно: $extension" -ForegroundColor Green

} else {

Write-Host "Не удалось установить: $extension" -ForegroundColor Red

}

}

Write-Host "Установка расширений VSCode завершена." -ForegroundColor Green

**Комментарии по выполнению**

* 1. **Выбор технологии:** PowerShell выбран как современный и мощный скриптовый язык для Windows. Chocolatey — это стандарт де-факто для управления пакетами в Windows, аналогичный apt в Linux.
  2. **Структура скрипта:**
     + **Проверка прав:** Скрипт требует права администратора, так как установка ПО затрагивает системные файлы.
     + **Функции:** Код организован в функции для улучшения читаемости и повторного использования.
     + **Обработка ошибок:** Скрипт проверяет код выхода ($LASTEXITCODE) после каждой установки, чтобы сообщить об успехе или неудаче.
     + **Интерактивность:** Запрос номера аудитории делает скрипт гибким.
  3. **Особенности установки:**
     + **WSL:** Установка дистрибутивов Ubuntu происходит через официальные команды wsl --install.
     + **Программы без пакетов:** Для некоторых программ (Julia, Yandex.Telemost, Sber Jazz) в отчетном периоде не нашлось стабильных пакетов в Chocolatey. Их установку можно добавить вручную, скачав установщики с официальных сайтов и запустив их с автоматическими ключами (/S для тихой установки).

**Заключение**

В ходе выполнения проекта была успешно решена задача автоматизации процесса установки и настройки программного обеспечения для компьютерных аудитоырий кафедры. Были достигнуты следующие ключевые результаты:

1. **Повышение эффективности:** Разработанный комплект скриптов на PowerShell позволяет значительно сократить временные затраты на подготовку рабочего места. Трудоемкий, многократно повторяющийся процесс установки десятков приложений сведен к запуску двух скриптов и минимальному вмешательству администратора.
2. **Стандартизация окружения:** Использование менеджера пакетов Chocolatey в качестве основы для установки гарантирует, что на всех компьютерах будут установлены идентичные версии программ из проверенных источников. Это устраняет расхождения в программном обеспечении, которые часто возникают при ручной установке, и создает единую, предсказуемую среду для учебного процесса.
3. **Минимизация ошибок:** Автоматизация исключает влияние человеческого фактора — основного источника ошибок при настройке множества систем. Скрипт выполняет все действия последовательно и по единому алгоритму, обеспечивая высокую надежность развертывания.

В результате проекта было разработано автоматизированное решение, позволяющее в разы ускорить процесс настройки компьютеров в учебных аудиториях. Использование скриптов гарантирует единообразие установленного программного обеспечения на всех рабочих местах, что создает стабильную и предсказуемую среду для проведения занятий.