Building MD4IoT SSH Ping Check

This guide explains how to build the executable and installer for the MD4IoT SSH Ping Check application.

Prerequisites

Required Software

1. **Python 3.8 or higher**
   - Download from python.org
   - Make sure to check "Add Python to PATH" during installation
2. **Inno Setup 6** (for creating Windows installer)
   - Download from jrsoftware.org
   - Install to default location for automatic detection

Required Files

Ensure your project directory contains:

```
project_root/
├── MSH.py              # Main application file
├── core_classes.py     # Business logic classes
├── favicon.ico         # Application icon
├── translations/       # Translation files directory
│   ├── en.json
│   └── ja.json
├── build.py            # Build automation script
├── requirements.txt    # Python dependencies
└── exampleCred.json    # Example credentials file (optional)
```

Build Process

Option 1: Automated Build (Recommended)

1. **Install Python dependencies**

```bash
pip install -r requirements.txt
```

2. **Run the build script**

```bash
python build.py
```

This will:

- Clean previous build artifacts
- Verify all required files are present
- Check Python dependencies
- Build the executable with PyInstaller
- Generate an Inno Setup script
- Create the Windows installer (if Inno Setup is installed)

3. **Find your outputs**
    - Executable: `dist/MD4IoT SSH Ping Check/`
    - Installer: `installer_output/MD4IoT_SSH_Ping_Check_Setup_v1.0.0.exe`

Option 2: Manual Build

If you need to build manually or customize the process:

Step 1: Install Dependencies

```bash
pip install cryptography netmiko pandas pyinstaller
```

Step 2: Build with PyInstaller

```bash
```

```
pyinstaller --add-data "translations;translations" ^
        --add-data "favicon.ico;." ^
        --onedir ^
        --noconsole ^
        --icon=favicon.ico ^
        --name="MD4IoT SSH Ping Check" ^
        --collect-all cryptography ^
        --collect-all netmiko ^
        --collect-all pandas ^
        --hidden-import=tkinter ^
        MSH.py
```

Note: On Linux/Mac, use colons (`:`) instead of semicolons (`;`) in `--add-data`:

```bash
--add-data "translations:translations"
```

Step 3: Test the Executable

```bash
cd "dist/MD4IoT SSH Ping Check"
"MD4IoT SSH Ping Check.exe"
```

Step 4: Create Installer (Optional)

1. Run the build script once to generate `installer.iss`:

```bash
  python build.py
```

Or manually create the Inno Setup script using the template below.

2. Open `installer.iss` with Inno Setup Compiler
3. Click Build → Compile
4. Find the installer in `installer_output/`

Troubleshooting

PyInstaller Issues

Problem: "Module not found" errors

```bash
# Solution: Add missing modules explicitly
pyinstaller ... --hidden-import=module_name
```

Problem: Translations folder not included

```bash
# Solution: Verify the --add-data path matches your directory structure
# Windows: "translations;translations"
# Linux/Mac: "translations:translations"
```

Problem: Icon not displaying

```bash
# Solution: Ensure favicon.ico exists and use absolute path if needed
pyinstaller ... --icon=C:\full\path\to\favicon.ico
```

Inno Setup Issues

Problem: ISCC.exe not found by build script

```
# Solution: Either:
# 1. Install Inno Setup to default location
# 2. Manually compile installer.iss using Inno Setup IDE
# 3. Update iscc_paths in build.py with your installation path
```

Problem: Missing files in installer

```
# Solution: Check the [Files] section in installer.iss
# Ensure Source paths match your dist directory structure
```

Runtime Issues

Problem: Application crashes on startup

```
# Solution: Test the executable in dist folder first:
cd "dist/MD4IoT SSH Ping Check"
"MD4IoT SSH Ping Check.exe"

# Check for missing DLLs or dependencies
# Run in console mode to see error messages:
pyinstaller ... --console ...  # Remove --noconsole flag
```

Problem: Translations not working

```
# Solution: Verify translations folder is in same directory as .exe
# The structure should be:
# MD4IoT SSH Ping Check.exe
# translations/
#    ├── en.json
#    └── ja.json
```

Build Configuration

Customizing the Build

Edit `build.py` to customize:

- `APP_NAME`: Application name
- `APP_VERSION`: Version number
- `MAIN_SCRIPT`: Entry point Python file
- `ICON_FILE`: Icon file name
- `PYINSTALLER_ARGS`: PyInstaller arguments

Modifying the Installer

Edit `installer.iss` (generated by build.py) to customize:

- Installation directory
- Start menu entries
- Desktop/Quick Launch icons
- Uninstaller behavior
- Language options

- Welcome/finish messages

Distribution

Portable Executable

To distribute without an installer:

1. Zip the entire `dist/MD4IoT SSH Ping Check/` folder
2. Users extract and run the .exe directly
3. No installation required

Windows Installer

Use the generated `.exe` installer from `installer_output/`

- Handles installation to Program Files
- Creates Start Menu shortcuts
- Registers uninstaller
- Manages file cleanup

Advanced Topics

Code Signing (Optional)

To sign your executable for Windows SmartScreen:

```bash
signtool sign /f certificate.pfx /p password /tr http://timestamp.digicert.com /td sha256 /fd sha256 "MD4IoT SSH Ping Check.
```

Creating a Single File Executable

To create a single .exe instead of a directory:

```bash
```

```
# Change in build.py:
PYINSTALLER_ARGS = [
    "--onefile",  # Instead of --onedir
    ...
]
```

Note: Single file builds are slower to start but easier to distribute.

Optimizing Size

To reduce executable size:

```bash
# Use UPX compression (install UPX first)
pyinstaller ... --upx-dir=C:\path\to\upx

# Exclude unnecessary modules
pyinstaller ... --exclude-module=module_name
```

CI/CD Integration

To automate builds in CI/CD:

```yaml
# Example GitHub Actions workflow
name: Build
on: [push]
jobs:
  build:
    runs-on: windows-latest
    steps:
      - uses: actions/checkout@v2
      - uses: actions/setup-python@v2
        with:
          python-version: '3.10'
      - run: pip install -r requirements.txt
      - run: python build.py
      - uses: actions/upload-artifact@v2
        with:
          name: installer
          path: installer_output/*.exe
```

License and Credits

See main README.md for application details and license information.