RENT A HAL: Windows 10 Setup Guide

This guide will walk you through setting up the RENT A HAL reference implementation of the Multi-Tronic Operating Realm (MTOR) on a Windows 10 system. By following these steps, you'll be able to run your own instance of this revolutionary intent-based compute platform.

Prerequisites

Before we begin, ensure your system meets these basic requirements:

- Windows 10 (64-bit)
- At least 16GB RAM
- At least 50GB free disk space
- Administrator privileges
- NVIDIA GPU (for full functionality)

Step 1: Install Git

First, we need to install Git to clone the repository:

1. Download Git from https://git-scm.com/download/win
2. Run the installer, accepting the default options
3. Verify installation by opening Command Prompt and typing:

   ```
   git --version
   ```

Step 2: Install Python

RENT A HAL requires Python 3.10 or later:

1. Download Python 3.10.x from https://www.python.org/downloads/windows/
2. Run the installer
3. **IMPORTANT**: Check "Add Python to PATH" during installation
4. Click "Install Now"
5. Verify installation by opening a new Command Prompt and typing:

   ```
   python --version
   ```

Step 3: Install CUDA Toolkit (for NVIDIA GPUs)

For optimal performance with GPU acceleration:

1. Visit https://developer.nvidia.com/cuda-downloads
2. Select:
   - Operating System: Windows
   - Architecture: x86_64
   - Version: 10/11
   - Installer Type: exe (local)
3. Download and run the installer
4. Choose "Express" installation
5. After installation, verify CUDA by opening Command Prompt and typing:

   ```
   nvcc --version
   ```

Step 4: Clone the RENT A HAL Repository

Now let's get the code:

1. Open Command Prompt
2. Navigate to where you want to install RENT A HAL:

   ```
   cd C:\Path\To\Your\Preferred\Directory
   ```

3. Clone the repository:

   ```
   git clone https://github.com/jimpames/rentahal.git
   ```

4. Navigate into the directory:

   ```
   cd rentahal
   ```

Step 5: Create a Virtual Environment

It's best practice to use a virtual environment:

1. In the rentahal directory, create a virtual environment:

   ```
   python -m venv venv
   ```

2. Activate the virtual environment:

```
venv\Scripts\activate
```

Your command prompt should now show `(venv)` at the beginning of the line

Step 6: Install Dependencies

Install all the required packages:

1. While in the virtual environment, install the requirements:

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

2. For PyTorch with CUDA support (which is commented out in the requirements.txt):

```
pip install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu121
```

Step 7: Configure RENT A HAL

1. Create a basic configuration file:

```
copy config.ini.example config.ini
```

2. Edit `config.ini` with Notepad or another text editor:

```
notepad config.ini
```

3. Update the following settings:
   - Set `host` to `0.0.0.0` (to allow access from other devices on your network) or `127.0.0.1` (local access only)
   - Set `port` to `5000` (or another port if 5000 is in use)
   - Configure worker addresses as needed (if you have other AI services running)

Step 8: Install Additional Dependencies

Some AI capabilities require additional software:

For Speech Synthesis/Recognition:

1. Install FFmpeg:
   - Download from https://www.gyan.dev/ffmpeg/builds/ (get the "essentials" build)
   - Extract the ZIP file
   - Copy the contents of the `bin` folder to `C:\Windows\System32\`

   - Verify installation by opening a new Command Prompt and typing:

     ```
     ffmpeg -version
     ```

2. Install Redis (for caching and message bus):
   - Download Redis for Windows from https://github.com/tporadowski/redis/releases
   - Run the MSI installer
   - Accept the default options

Step 9: Run RENT A HAL

Now you're ready to run the system:

1. Start Redis (if not started automatically):

   ```
   redis-server
   ```

2. In a new Command Prompt window with your virtual environment activated, start the RENT A HAL server:

   ```
   python webgui.py
   ```

3. Once running, open your web browser and navigate to:

   ```
   http://localhost:5000
   ```

4. Set your nickname in the interface to begin using the system

Step 10: Connect Worker Nodes (Optional)

If you want to use external AI worker nodes:

1. Configure worker addresses in `config.ini`
2. Ensure the worker nodes are running and accessible
3. Restart RENT A HAL if it's currently running

Common Issues and Solutions

Issue: "No module named X" error

Solution: Ensure you're in the virtual environment and try reinstalling dependencies:

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

Issue: CUDA not recognized

Solution: Make sure you have compatible NVIDIA drivers installed, then reinstall PyTorch with CUDA support:

```
pip uninstall torch torchvision torchaudio
pip install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu121
```

Issue: Redis connection error

Solution: Make sure Redis is running. Open a new Command Prompt and type:

```
redis-cli ping
```

It should reply with "PONG". If not, restart Redis.

Issue: Port already in use

Solution: Change the port in `config.ini` to an unused port (e.g., 5001, 8000, etc.)

Using RENT A HAL

Once set up, you'll have access to various AI capabilities:

1. **Chat**: Text-based conversation with AI
2. **Vision**: Upload images for AI analysis
3. **Imagine**: Generate images from text descriptions
4. **Speech**: Voice-based interaction (with Wake Word mode)

To use the Wake Word feature, click "Enable Wake Word Mode" and say "Computer" to activate the system.

Performance Optimization

For best performance:

1. Adjust the number of workers in `config.ini` based on your CPU cores
2. If using NVIDIA GPUs, ensure you have the latest drivers
3. Close other GPU-intensive applications when running RENT A HAL

Extending RENT A HAL

The system is designed to be extended:

1. New worker nodes can be added in the admin panel
2. New capabilities can be integrated by creating new worker types
3. Custom front-end modifications can be made in the static files

Conclusion

You now have a working instance of RENT A HAL, the reference implementation of the Multi-Tronic Operating Realm. This revolutionary architecture drastically reduces computing resource requirements while providing a unified interface for diverse AI capabilities.

Explore the system's capabilities, monitor its efficiency, and experience firsthand how intent-based, stateless event-driven computing can transform our approach to AI.

For more information and updates, visit:

- RENT A HAL on X
- RENT A HAL on GitHub
- About RENT A HAL