FT_2025 Complete Windows Development Setup

Your complete guide to professional Field Trainer development on Windows with VS Code and GitHub integration.

□ What You Get

I've created a **complete Windows development environment** for your FT_2025 Field Trainer project:

Windows-Optimized Development

✅ **VS Code configured** for Windows Python development
✅ **PowerShell automation** for Device 0 management
✅ **Windows SSH setup** with automated key generation
✅ **Virtual environment** using Windows Python
✅ **One-click deployment** from Windows to Device 0

Professional Development Tools

✅ **IntelliSense** - Smart code completion for Field Trainer
✅ **Debugging** - Local and remote debugging on Device 0
✅ **Automated testing** - Pytest integration
✅ **Code formatting** - Black formatter on save
✅ **Git integration** - Complete GitHub workflow

Device 0 Integration

✅ **SSH automation** - Passwordless connection to Device 0
✅ **Remote deployment** - Deploy from Windows to Linux Device 0
✅ **Live monitoring** - Stream logs from Device 0 to Windows
✅ **Service management** - Start/stop/restart from Windows

🚀 5-Minute Setup Process

Step 1: Prerequisites (2 minutes)

Install on Windows:

- **Python 3.7+** - python.org
- **Git for Windows** - git-scm.com
- **VS Code** - code.visualstudio.com

Check OpenSSH (usually pre-installed):

```powershell
ssh -V
# If not found: Settings → Apps → Optional Features → Add OpenSSH Client
```

Step 2: Create GitHub Repository (1 minute)

1. Go to GitHub.com → **New Repository**
2. Name: `FT_2025`
3. Description: `Field Trainer 2025 - Circuit training management system`
4. ✅ Initialize with README, Python .gitignore, MIT license

Step 3: Clone and Setup (2 minutes)

```powershell
# Clone your repository
git clone https://github.com/YOUR_USERNAME/FT_2025.git
cd FT_2025

# Copy all files from artifacts (see WINDOWS_FILE_LOCATIONS.md)
# Then run automated setup:
.\setup_vscode.bat
```

The setup script will:

- ✅ Install VS Code extensions
- ✅ Create Python virtual environment
- ✅ Install dependencies
- ✅ Configure VS Code for Windows
- ✅ Open VS Code with your project

📂 File Organization

All files are in the artifacts above in our conversation. Here's what goes where:

Core Files (Root Directory)

From artifacts, copy to root of FT_2025:

- `field_trainer_core.py` - Core device management

- `field_trainer_web.py` - Web interface
- `field_trainer_main.py` - Main application
- `setup_vscode.bat` - Windows setup script
- `requirements.txt` - Python dependencies
- `.gitignore` - Git exclusions

VS Code Configuration (`.vscode\` folder)

From artifacts, copy to `.vscode\`:

- `settings.json` - Windows-optimized VS Code settings
- `extensions.json` - Recommended extensions
- `launch.json` - Windows debug configurations
- `tasks.json` - PowerShell-based tasks

Scripts (`scripts\` folder)

From artifacts, copy to `scripts\`:

- `setup_ssh.ps1` - PowerShell SSH setup for Windows
- `github_deploy.sh` - Linux deployment script (for Device 0)
- **Other .sh files** - Linux utility scripts

Configuration (`config\` folder)

- `field_trainer.conf` - System configuration
- `courses.json` - Training course definitions

Documentation

- `WINDOWS_DEVELOPMENT.md` - Complete Windows development guide
- `WINDOWS_FILE_LOCATIONS.md` - File collection guide
- **Other .md files** - Additional documentation

🔧 Windows Development Features

VS Code Tasks (Ctrl+Shift+P → "Tasks: Run Task")

Local Development:

- **Run Local Development Server** - Test locally on Windows

- **Run Tests** - Execute pytest
- **Format Python Code** - Auto-format with Black
- **Lint Python Code** - Check code quality

Device 0 Management (via SSH):

- **Deploy to Device 0** - Push latest code to Device 0
- **Check Device 0 Status** - Service status check
- **View Device 0 Logs** - Live log streaming
- **Restart Field Trainer Service** - Remote restart
- **Backup Device 0** - Create backup
- **Setup SSH Key for Device 0** - Run SSH setup

Debug Configurations (F5)

- **Field Trainer - Full System** - Debug complete system locally
- **Field Trainer - Web Only** - Debug just Flask web interface
- **Field Trainer - Core Only** - Debug just TCP server
- **Python: Remote Device 0 Debug** - Debug code running on Device 0

PowerShell Automation

SSH Setup:

```powershell
.\scripts\setup_ssh.ps1
# Generates SSH keys, configures connection to Device 0
```

Quick Commands:

```powershell
```

```
# Deploy to Device 0
ssh device0 "/opt/field-trainer/scripts/github_deploy.sh"

# Check service status
ssh device0 "sudo systemctl status field-trainer"

# View live logs
ssh device0 "sudo journalctl -u field-trainer -f"
```

🔗 Windows to Device 0 Workflow

1. Develop on Windows

- Edit Python files in VS Code with IntelliSense
- Debug locally with breakpoints (F5)
- Test web interface at `http://localhost:5000`
- Format and lint code automatically

2. Deploy to Device 0

- Commit changes: `git add . && git commit -m "message"`
- Push to GitHub: `git push origin main`
- Deploy to Device 0: VS Code task or `ssh device0 "/opt/field-trainer/scripts/github_deploy.sh"`

3. Monitor Device 0

- Check service: VS Code task or `ssh device0 "sudo systemctl status field-trainer"`
- View logs: VS Code task for live streaming
- Access web interface: `http://192.168.99.100:5000`

4. Remote Debugging (Optional)

- Add `debugpy` code to Python files
- Deploy to Device 0
- Attach VS Code debugger to running service
- Debug remotely with local breakpoints

💡 Development Workflow Example

Daily development process:

```powershell
# 1. Start development
code .  # Open VS Code
git pull origin main  # Get latest changes

# 2. Local development
F5 → "Field Trainer - Full System"  # Start debugging
# Edit code with IntelliSense
# Set breakpoints and test

# 3. Code quality
Ctrl+Shift+P → "Format Python Code"  # Auto-format
Ctrl+Shift+P → "Run Tests"  # Run tests

# 4. Commit and deploy
git add .
git commit -m "Add new feature"
git push origin main
Ctrl+Shift+P → "Deploy to Device 0"  # Deploy to Device 0

# 5. Monitor
Ctrl+Shift+P → "Check Device 0 Status"  # Verify deployment
Ctrl+Shift+P → "View Device 0 Logs"  # Monitor logs
```

🎯 Key Benefits for Windows Developers

Before: Manual Development

- Edit files in basic text editor
- Manual file copying to Device 0
- No version control
- No debugging capabilities
- Manual SSH commands

After: Professional Windows Development

✅ **VS Code IDE** - IntelliSense, debugging, integrated terminal
✅ **One-click deployment** - Windows to Device 0 in seconds
✅ **Remote debugging** - Debug Device 0 code from Windows
✅ **Automated SSH** - Passwordless connection setup
✅ **Git integration** - Professional version control workflow

✅ **Quality tools** - Automated testing, formatting, linting
✅ **Live monitoring** - Stream Device 0 logs to Windows

🔍 File Collection Quick Reference

Where to find files: All files are in the **artifacts** (code blocks) above in our conversation.

How to collect:

1. Scroll up to find each artifact
2. Click artifact to expand
3. Copy content
4. Create file in correct Windows location
5. Paste and save

See `WINDOWS_FILE_LOCATIONS.md` for complete file-by-file guide.

🛠️ Windows-Specific Configuration

Python Virtual Environment

```powershell
# Created automatically by setup_vscode.bat
.\venv\Scripts\Activate.ps1  # Activate
.\venv\Scripts\python.exe    # Python interpreter
.\venv\Scripts\pip.exe       # Package installer
```

SSH Configuration

```powershell
# Location: %USERPROFILE%\.ssh\config
# Created by: .\scripts\setup_ssh.ps1

Host device0
    HostName 192.168.99.100
    User pi
    IdentityFile ~/.ssh/id_rsa
```

VS Code Settings

- **Python interpreter:** `.\venv\Scripts\python.exe`

- **Terminal:** PowerShell
- **Formatting:** Black (88 character line length)
- **Linting:** Flake8
- **Auto-save:** Enabled

🚨 Troubleshooting Quick Fixes

VS Code Python interpreter not found:

- `Ctrl+Shift+P` → "Python: Select Interpreter" → Choose `.\venv\Scripts\python.exe`

PowerShell script execution blocked:

```powershell
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
```

SSH connection issues:

```powershell
# Test connection
ssh -v device0

# Re-run SSH setup
.\scripts\setup_ssh.ps1
```

Port conflicts (5000/6000 in use):

```powershell
netstat -ano | findstr :5000
taskkill /PID [PID] /F
```

📖 Complete Documentation

Read these guides for detailed information:

- `WINDOWS_DEVELOPMENT.md` - Complete Windows development guide
- `WINDOWS_FILE_LOCATIONS.md` - File collection instructions
- `DEPLOYMENT_GUIDE.md` - Device 0 deployment details
- `GITHUB_SETUP.md` - GitHub integration setup

🎉 Ready to Start!

Your Windows development environment provides:

🎯 **Professional Development** - VS Code, IntelliSense, debugging
🚀 **One-Click Deployment** - Windows to Device 0 instantly
🎯 **Remote Integration** - Manage Device 0 from Windows
🤝 **Team Collaboration** - Git workflow with GitHub
📊 **Quality Assurance** - Automated testing and formatting
🛡️ **Enterprise-Grade** - Professional development practices

🚀 Start Your Professional Field Trainer Development!

1. **Collect all files** from artifacts using `WINDOWS_FILE_LOCATIONS.md`
2. **Run setup:** `.\setup_vscode.bat`
3. **Configure SSH:** `.\scripts\setup_ssh.ps1`
4. **Start coding:** `F5` to debug locally
5. **Deploy:** Use VS Code tasks to deploy to Device 0

Welcome to professional Field Trainer development on Windows! 🏃⚡