Building Heaven GUI

Instructions for building the Heaven Interface GUI companion app.

Prerequisites

System Requirements

- **Operating System:** macOS, Linux, or Windows
- **Node.js:** 18+
- **Rust:** 1.70+
- **Tauri CLI:** Latest version

Install Rust

```
# Install Rust (if not already installed)
curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh

# Add to PATH
source $HOME/.cargo/env

# Verify installation
rustc --version
cargo --version
```

Install Tauri Prerequisites

macOS

```
# Install Xcode Command Line Tools
xcode-select --install

# No additional dependencies needed
```

Linux (Ubuntu/Debian)

```
sudo apt update
sudo apt install libwebkit2gtk-4.0-dev \
    build-essential \
    curl \
    wget \
    file \
    libssl-dev \
    libgtk-3-dev \
    libayatana-appindicator3-dev \
    librsvg2-dev
```

Linux (Fedora)

```
sudo dnf install webkit2gtk4.0-devel \
    openssl-devel \
    curl \
    wget \
    file \
    libappindicator-gtk3-devel \
    librsvg2-devel

sudo dnf group install "C Development Tools and Libraries"
```

Windows

```
# Install Visual Studio C++ Build Tools
# Download from: https://visualstudio.microsoft.com/visual-cpp-build-tools/

# Install WebView2
# Download from: https://developer.microsoft.com/en-us/microsoft-edge/webview2/
```

Development Setup

1. Clone and Install Dependencies

```
# Navigate to GUI directory
cd heaven-gui

# Install Node dependencies
npm install

# This will install:
# - React & React DOM
# - Tauri API bindings
# - Vite (build tool)
# - TypeScript
# - D3.js (for graphs)
# - Monaco Editor
# - Recharts (for metrics)
```

2. Run in Development Mode

```
# Start development server with hot reload
npm run tauri:dev

# This will:
# 1. Start Vite dev server on http://localhost:5173
# 2. Compile Rust backend
# 3. Launch Tauri window
# 4. Enable hot module replacement (HMR)
```

Development Features:
- Hot reload for React components
- Fast refresh (< 1 second)
- Debug console in DevTools
- Rust compilation on save

3. Build for Production

```
# Build optimized production bundle
npm run tauri:build

# Output locations:
# - macOS: src-tauri/target/release/bundle/dmg/
# - Linux: src-tauri/target/release/bundle/appimage/
# - Windows: src-tauri/target/release/bundle/msi/
```

Build Targets:

Platform	Output Format	Location
macOS	`.app`, `.dmg`	`target/release/bundle/macos/`
Linux	`.AppImage`, `.deb`	`target/release/bundle/`
Windows	`.exe`, `.msi`	`target/release/bundle/`

4. Build for Specific Platform

```
# Build only the app (no installer)
npm run tauri:build -- --no-bundle

# Build specific format
npm run tauri:build -- --bundles dmg      # macOS DMG
npm run tauri:build -- --bundles appimage # Linux AppImage
npm run tauri:build -- --bundles msi      # Windows MSI
```

Project Structure

```
heaven-gui/
├── src/                    # React frontend
│   ├── components/         # React components
│   ├── styles/             # CSS files
│   ├── App.tsx             # Main app component
│   └── main.tsx            # Entry point
├── src-tauri/              # Rust backend
│   ├── src/
│   │   └── main.rs         # Tauri backend
│   ├── Cargo.toml          # Rust dependencies
│   └── tauri.conf.json     # Tauri configuration
├── index.html              # HTML template
├── package.json            # Node dependencies
├── vite.config.ts          # Vite configuration
└── tsconfig.json           # TypeScript config
```

Configuration

Tauri Configuration (`src-tauri/tauri.conf.json`)

Key settings:

```
{
  "tauri": {
    "allowlist": {
      "fs": {
        "scope": ["$HOME/.sologit/**"]  // File system access
      }
    },
    "windows": [{
      "width": 1400,
      "height": 900,
      "minWidth": 1000,
      "minHeight": 600
    }]
  }
}
```

Environment Variables

```
# Development mode (enables DevTools)
export TAURI_DEBUG=true

# Custom state directory (for testing)
export SOLOGIT_STATE_DIR=/path/to/test/state
```

Testing

Frontend Tests

```
# Run React component tests (when implemented)
npm test

# Run with coverage
npm run test:coverage
```

Backend Tests

```
# Run Rust tests
cd src-tauri
cargo test

# Run with output
cargo test -- --nocapture
```

Integration Tests

```
# Test state reading
cd src-tauri
cargo run -- read-state

# Test GUI launch
npm run tauri:dev
# Verify:
# 1. Window opens
# 2. State loads from ~/.sologit/state/
# 3. Commit graph displays
# 4. Status bar shows correct info
```

Troubleshooting

Build Fails: “Rust compiler not found”

```
# Reinstall Rust
rustup update
rustup default stable

# Verify
rustc --version
```

Build Fails: “webkit2gtk not found” (Linux)

```
# Install missing dependency
sudo apt install libwebkit2gtk-4.0-dev

# Or on Fedora
sudo dnf install webkit2gtk4.0-devel
```

GUI Doesn’t Load State

```
# Verify state directory exists
ls -la ~/.sologit/state/

# Check permissions
chmod 755 ~/.sologit/state/

# Test state reading
cd src-tauri
cargo run
```

Hot Reload Not Working

```
# Kill all Node/Tauri processes
killall node
killall heaven-gui

# Clear cache
rm -rf node_modules/.vite
rm -rf src-tauri/target/debug

# Restart
npm run tauri:dev
```

Windows-Specific Issues

```
# Run as administrator
# Right-click CMD/PowerShell → "Run as Administrator"

# Check WebView2
# Download and install: https://developer.microsoft.com/en-us/microsoft-edge/webview2/
```

Performance Optimization

Production Build Optimizations

The build is optimized by default:

- **Frontend:**
- Vite minification (esbuild)
- Tree shaking
- Code splitting
- Asset optimization
- **Backend:**
- Cargo release mode
- Link-time optimization (LTO)
- Strip symbols

Binary Size

Target binary sizes:

Platform	Uncompressed	Compressed
macOS	~15MB	~8MB
Linux	~18MB	~10MB
Windows	~12MB	~7MB

To reduce size further:

```
# Add to src-tauri/Cargo.toml
[profile.release]
opt-level = "z"  # Optimize for size
lto = true
codegen-units = 1
panic = "abort"
strip = true
```

Continuous Integration

GitHub Actions Example

```yaml
name: Build Heaven GUI

on: [push, pull_request]

jobs:
  build:
    strategy:
      matrix:
        platform: [ubuntu-latest, macos-latest, windows-latest]

    runs-on: ${{ matrix.platform }}

    steps:
      - uses: actions/checkout@v3

      - name: Setup Node
        uses: actions/setup-node@v3
        with:
          node-version: 18

      - name: Setup Rust
        uses: actions-rs/toolchain@v1
        with:
          toolchain: stable

      - name: Install Linux dependencies
        if: matrix.platform == 'ubuntu-latest'
        run: |
          sudo apt update
          sudo apt install -y libwebkit2gtk-4.0-dev

      - name: Install dependencies
        working-directory: heaven-gui
        run: npm install

      - name: Build
        working-directory: heaven-gui
        run: npm run tauri:build

      - name: Upload artifacts
        uses: actions/upload-artifact@v3
        with:
          name: heaven-gui-${{ matrix.platform }}
          path: heaven-gui/src-tauri/target/release/bundle/
```

Distribution

macOS

```
# Build DMG
npm run tauri:build -- --bundles dmg

# Sign and notarize (requires Apple Developer account)
codesign --force --sign "Developer ID Application" heaven.app
xcrun notarytool submit heaven.dmg
```

Linux

```
# Build AppImage (universal)
npm run tauri:build -- --bundles appimage

# Build deb (Debian/Ubuntu)
npm run tauri:build -- --bundles deb

# Build rpm (Fedora/RedHat)
npm run tauri:build -- --bundles rpm
```

Windows

```
# Build MSI installer
npm run tauri:build -- --bundles msi

# Sign (requires code signing certificate)
signtool sign /f cert.pfx /p password heaven.msi
```

Development Tips

1. **Use DevTools:** Press `Cmd+Option+I` (macOS) or `Ctrl+Shift+I` (Linux/Windows) to open DevTools
2. **Live State Updates:** GUI polls state every 3 seconds. Run CLI commands to see updates.
3. **Mock Data:** For development without Solo Git, create mock state files in `~/.sologit/state/`
4. **Hot Reload:** Frontend changes reload instantly. Backend changes require restart.
5. **Debug Backend:** Use `cargo run` in `src-tauri/` to see Rust console output.

License