ReadMe

MyTools - GPS File Processing Suite

A comprehensive Docker-based toolkit for GPS and mapping file processing, featuring two powerful applications for converting and extracting GPS data.

□□ Applications

□□ Google GPX Converter (Port 6010)
Convert Google My Maps (.kmz) files to standard GPX format with a client-side web interface.

Features:
- □□ Convert Google My Maps to GPX format
- □□ Client-side processing (no data uploaded to servers)
- □□ Responsive web interface
- ⚡ Fast, browser-based conversion
- □□ Complete privacy - all processing local

□□ Extract GPX Parts (Port 6020)
Extract and separate components (waypoints, tracks, routes) from GPX container files.

Features:
- □□ Extract waypoints, tracks, and routes from GPX files
- □□ Web interface with drag-drop upload
- □□ Batch download as ZIP archives
- □□ File preview and content inspection
- □□ Processing statistics and summaries
- □□ Persistent storage with automatic cleanup

□□ Quick Start

Prerequisites
- Docker with Compose plugin
- 8GB+ available disk space
- Ports 6010 and 6020 available

Installation

1. **Clone the repository:**
```bash
git clone https://github.com/your-username/mytools-gps-suite.git
cd mytools-gps-suite
```

2. **Deploy with Docker Compose:**
```bash
cd /opt/containerd/myTools
./deploy.sh
```

3. **Access Applications:**
- **Google GPX Converter:** http://localhost:6010
- **Extract GPX Parts:** http://localhost:6020

□□ Directory Structure

```
myTools/
├── docker-compose.yml # Main deployment configuration
├── deploy.sh # Deployment script
├── google-gpx-converter/ # Google Maps to GPX converter
│ ├── Dockerfile
│ ├── .copilot-instructions.md
│ ├── index.html
│ ├── script.js
│ ├── styles.css
│ └── url-expander.js
├── extract-gpx-parts/ # GPX component extractor
│ ├── Dockerfile
│ ├── .copilot-instructions.md
│ ├── gpx_extractor.py # CLI extraction tool
│ ├── data/ # Persistent storage
│ │ ├── uploads/ # User uploaded files
│ │ └── processed/ # Extracted components
│ └── web/ # Flask web application
│ ├── app.py
│ ├── requirements.txt
│ ├── templates/
│ └── static/
└── README.md # This file
```

□□ Usage

Google GPX Converter

1. **Open** http://localhost:6010 in your browser
2. **Upload** a Google My Maps (.kmz) file via drag-drop or file selection
3. **Convert** - processing happens instantly in your browser
4. **Download** the resulting GPX file

Supported Input Formats:
- Google My Maps (.kmz files)
- Google Maps shared URLs

Output Format:
- Standard GPX 1.1 format compatible with GPS devices and mapping software

Extract GPX Parts

1. **Open** http://localhost:6020 in your browser
2. **Upload** a GPX file containing multiple components
3. **Review** extracted components in the results table
4. **Select** individual files or use batch selection
5. **Download** selected files individually or as a ZIP archive

Supported Input:
- GPX 1.0 and 1.1 files
- Large GPS tracks (tested with 900+ waypoints, 28+ tracks)
- Complex route files with multiple segments

Output:
- Separate GPX files for waypoints, tracks, and routes
- Preserved metadata and naming conventions
- ZIP archives for batch downloads

□□ Management Commands

Container Management
```bash
# Start services
docker compose up -d

# Stop services
docker compose down

# View logs
docker compose logs -f

# Restart services
docker compose restart

# Rebuild containers
docker compose up -d --build
```

File Management
```bash
```

```
# View uploaded files
ls -la /opt/containerd/myTools/extract-gpx-parts/data/uploads/

# View processed files
ls -la /opt/containerd/myTools/extract-gpx-parts/data/processed/

# Clean up old files (optional - auto-cleanup after 1 hour)
find /opt/containerd/myTools/extract-gpx-parts/data/ -type f -mtime +1 -delete
```

□□ Security & Privacy

Google GPX Converter
- **100% Client-Side:** No data transmitted to external servers
- **Browser Processing:** All conversion happens in your browser
- **No Storage:** Files are not stored after conversion
- **Open Source:** Transparent processing logic

Extract GPX Parts
- **Local Processing:** Files processed on your own server
- **Automatic Cleanup:** Temporary files removed after 1 hour
- **Secure Upload:** File validation and sanitization
- **Isolated Environment:** Docker container isolation

□□ Troubleshooting

Common Issues

Port Conflicts:
```bash
# Check if ports are in use
sudo netstat -tlnp | grep -E ':601[0-9]'

# Stop conflicting services
sudo fuser -k 6010/tcp 6020/tcp
```

Container Build Failures:
```bash
# Clean Docker cache
docker system prune -a

# Rebuild from scratch
docker compose down
docker compose up -d --build
```

Permission Issues:
```bash
# Fix directory permissions
sudo chown -R gerald:docker /opt/containerd/myTools
sudo chmod -R 775 /opt/containerd/myTools
```

Storage Issues:
```bash
# Check disk space
df -h /opt/containerd/

# Clean old files manually
sudo find /opt/containerd/myTools/extract-gpx-parts/data/ -type f -mtime +1 -delete
```

□□ Development

Local Development Setup

1. **Google GPX Converter** (Static files):
```bash
cd google-gpx-converter
python -m http.server 8080
# Access at http://localhost:8080
```

2. **Extract GPX Parts** (Flask development):
```bash
cd extract-gpx-parts/web
python -m venv venv
source venv/bin/activate
pip install -r requirements.txt
python app.py
```

Testing

Test Google GPX Converter:
- Download sample .kmz files from Google My Maps
- Test with various map types (points, routes, areas)
- Verify GPX output in GPS software

Test Extract GPX Parts:
- Use provided S.gpx test file (909 waypoints, 28 tracks)
- Test with various GPX file structures
- Verify component separation accuracy

□□ API Documentation

Extract GPX Parts API Endpoints

- `GET /` - Main upload interface
- `POST /upload` - Upload and process GPX file
- `GET /download/<directory>/<filename>` - Download individual file
- `POST /download-selected` - Download selected files as ZIP
- `GET /preview/<directory>/<filename>` - Preview file content

□□ Contributing

1. **Fork** the repository
2. **Create** a feature branch (`git checkout -b feature/amazing-feature`)
3. **Commit** your changes (`git commit -m 'Add amazing feature'`)
4. **Push** to the branch (`git push origin feature/amazing-feature`)
5. **Open** a Pull Request

Development Guidelines
- Follow existing code style and patterns
- Add tests for new functionality
- Update documentation as needed
- Ensure Docker compatibility

□□ License

This project is licensed under the MIT License - see the LICENSE file for details.

□□ Acknowledgments

- **ElementTree** - XML processing for GPX files
- **Flask** - Web framework for Python applications
- **JSZip** - JavaScript ZIP file handling
- **Docker** - Containerization platform
- **Nginx** - Web server for static file serving

□□ Support

For issues, questions, or contributions:

- **Issues:** Create an issue on GitHub
- **Discussions:** Use GitHub Discussions for questions
- **Documentation:** Check `.copilot-instructions.md` in each application directory

Made with ❤ for the GPS and mapping community

Revision #1
Created 2026-02-22 17:54:47 UTC by Gerald Amrhein
Updated 2026-02-22 17:55:00 UTC by Gerald Amrhein