# Docker Setup for GLSP Generator

This document describes how to run the GLSP Generator using Docker, providing both CLI and API access.

## Quick Start - Using Docker Hub

The GLSP Generator is available on Docker Hub. No build required!

### Pull from Docker Hub

# Pull the latest version  
docker pull johnholliday/glsp-generator:latest  
  
# Or pull a specific version  
docker pull johnholliday/glsp-generator:2.1.171

### CLI Usage

# Generate VSIX from grammar file  
docker run --rm -v $(pwd):/workspace \  
 johnholliday/glsp-generator \  
 generate /workspace/my-grammar.langium  
  
# Show help  
docker run --rm johnholliday/glsp-generator --help  
  
# Validate grammar  
docker run --rm -v $(pwd):/workspace \  
 johnholliday/glsp-generator \  
 validate /workspace/my-grammar.langium

### API Server Usage

# Start API server  
docker run -d -p 3000:3000 --name glsp-api \  
 johnholliday/glsp-generator \  
 node dist/api-server.js  
  
# Check health  
curl http://localhost:3000/health  
  
# Stop server  
docker stop glsp-api && docker rm glsp-api

## Quick Start - Building Locally

### Running the API Server

# Build and start the API server  
docker-compose up -d  
  
# Check logs  
docker-compose logs -f glsp-generator-api  
  
# Stop the server  
docker-compose down

The API will be available at http://localhost:3000.

### Using the CLI

# Run CLI commands using docker-compose  
docker-compose run --rm glsp-generator-cli generate /workspace/grammar.langium -o /workspace/output  
  
# Or use the profile  
docker-compose --profile cli run glsp-generator-cli validate /workspace/grammar.langium

## API Endpoints

### Health Check

curl http://localhost:3000/health

### Generate VSIX

curl -X POST http://localhost:3000/generate/vsix \  
 -F "grammar=@./my-grammar.langium" \  
 -F "name=my-extension" \  
 -F "version=1.0.0" \  
 -o my-extension-1.0.0.vsix

### Generate Theia Application

curl -X POST http://localhost:3000/generate/theia \  
 -F "grammar=@./my-grammar.langium" \  
 -F "name=my-app" \  
 -o my-app-theia.zip

### Validate Grammar

curl -X POST http://localhost:3000/validate \  
 -F "grammar=@./my-grammar.langium"

## Development Setup

### Running in Development Mode

# Start development server with hot-reload  
docker-compose --profile development up  
  
# The API will be available at http://localhost:3001

### Building the Image

# Build the Docker image  
docker build -t glsp-generator .  
  
# Or using docker-compose  
docker-compose build

### Running with Custom Configuration

# Run with custom environment variables  
docker run -p 3000:3000 \  
 -e LOG\_LEVEL=debug \  
 -e PORT=3000 \  
 -v $(pwd)/workspace:/workspace \  
 glsp-generator node dist/api-server.js

## Production Deployment

### With Nginx Reverse Proxy

# Start with production profile (includes Nginx)  
docker-compose --profile production up -d  
  
# The service will be available at http://localhost

### SSL/TLS Configuration

1. Place your SSL certificates in the ssl/ directory:
   * ssl/cert.pem - Certificate file
   * ssl/key.pem - Private key file
2. Uncomment the HTTPS server block in nginx.conf
3. Restart the services:

* docker-compose --profile production restart

## Volume Mounts

The following volumes are used:

* /workspace - Input/output directory for grammar files and generated code
* /app/templates - Template directory (read-only in production)

## Environment Variables

* NODE\_ENV - Environment mode (development/production)
* PORT - API server port (default: 3000)
* LOG\_LEVEL - Logging level (trace/debug/info/warn/error)

## Troubleshooting

### Container won't start

# Check logs  
docker-compose logs glsp-generator-api  
  
# Rebuild the image  
docker-compose build --no-cache

### Permission issues

# Ensure workspace directory has proper permissions  
chmod -R 755 ./workspace

### API not responding

# Check health endpoint  
curl http://localhost:3000/health  
  
# Check if container is running  
docker-compose ps

## Docker Commands Reference

# Build image  
docker-compose build  
  
# Start services  
docker-compose up -d  
  
# Stop services  
docker-compose down  
  
# View logs  
docker-compose logs -f  
  
# Execute command in running container  
docker-compose exec glsp-generator-api sh  
  
# Run one-off command  
docker-compose run --rm glsp-generator-cli generate --help  
  
# Remove all containers and volumes  
docker-compose down -v