TrustWeave Open Source Release Strategy

> **Comprehensive plan for launching TrustWeave as an open source project**

Created: 2025
Status: Planning Phase
Target Launch: TBD (Recommended: 4-6 weeks from strategy approval)

Executive Summary

TrustWeave is well-positioned for open source release. The codebase is well-structured, comprehensive documentation exists, and a dual-license model is already defined. This strategy document outlines the complete roadmap for launching and growing TrustWeave as a successful open source project.

Key Strengths:

- ☑ Comprehensive documentation (25+ use case scenarios)
- ☑ Well-architected plugin system
- ☑ Dual-license model already defined
- ☑ Production-ready codebase
- ☑ Strong developer experience (type-safe APIs, DSLs)

Strategic Approach:

1. **Open Source First** - Release SDK as open source to maximize adoption
2. **Commercial Layer** - Monetize through commercial licensing and services
3. **Community Growth** - Build vibrant developer community
4. **Future SaaS** - Add managed services layer after community traction

---

Phase 1: Pre-Release Preparation (Weeks 1-2)

1.1 Code & Documentation Cleanup

Immediate Actions:

- ☐ Create `CONTRIBUTING.md` at root (consolidate from `docs/contributing/`)
- ☐ Create `CHANGELOG.md` with initial v1.0.0 entry
- ☐ Add `CODE_OF_CONDUCT.md` (use Contributor Covenant)
- ☐ Add `SECURITY.md` for vulnerability reporting
- ☐ Review and clean up any internal-only documentation
- ☐ Ensure all license headers are present in source files
- ☐ Remove or document any deprecated code paths

Code Quality Checks:

```
# Run comprehensive checks
./gradlew clean build test ktlintCheck
./gradlew :distribution:all:build  # Ensure distribution builds
```

1.2 GitHub Repository Setup

Repository Configuration:

- □ Set repository to Public
- □ Add repository topics: `kotlin`, `did`, `verifiable-credentials`, `blockchain`, `identity`, `ssi`, `w3c`, `decentralized-identity`
- □ Configure repository description: "The Foundation for Decentralized Trust and Identity - A neutral, reusable Kotlin library for W3C-compliant DIDs and Verifiable Credentials"
- □ Set up branch protection rules for `main`:
    - Require PR reviews (1-2 reviewers)
    - Require status checks to pass
    - Require branches to be up to date
    - Do not allow force pushes
- □ Enable GitHub Discussions
- □ Enable GitHub Sponsors (optional, for future)

GitHub Actions/CI Setup:

- □ Create `.github/workflows/ci.yml` for automated testing
- □ Create `.github/workflows/release.yml` for automated releases
- □ Set up automated dependency updates (Dependabot)
- □ Configure code scanning (CodeQL)

1.3 Versioning Strategy

Recommendation: Semantic Versioning (SemVer)

Current version: `1.0.0-SNAPSHOT`

Release Plan:

- **v1.0.0** (Initial Release) - Current stable codebase
- **v1.0.x** (Patch releases) - Bug fixes, security patches
- **v1.x.0** (Minor releases) - New features, backward compatible
- **v2.0.0** (Major releases) - Breaking changes

Version Management:

```
// In build.gradle.kts
version = "1.0.0"  // Remove -SNAPSHOT for release
```

Release Branches:

- `main` - Development (SNAPSHOT versions)
- `release/1.0.x` - Release branches for patches
- `release/1.x.0` - Release branches for minor versions

1.4 Legal & Licensing Finalization

Verify:

- □ All source files have license headers
- □ Third-party dependencies are compatible with AGPL v3.0
- □ Commercial license terms are clear and accessible
- □ Contributor License Agreement (CLA) or DCO (Developer Certificate of Origin) decision
  - **Recommendation:** Use DCO (simpler, more developer-friendly)
- □ Update `LICENSE-COMMERCIAL.md` with final terms
- □ Create `NOTICE` file listing third-party licenses

Phase 2: Release Artifacts (Week 2-3)

2.1 Maven Central Publication

Setup Maven Central Publishing:

1. **Sonatype OSSRH Account:**
   - □ Create account at https://issues.sonatype.org
   - □ Request namespace: `com.trustweave`
   - □ Set up GPG signing keys
2. **Gradle Configuration:**
   - □ Add `maven-publish` plugin configuration
   - □ Configure signing plugin
   - □ Set up publication tasks for all modules
   - □ Create BOM (Bill of Materials) publication
3. **Publication Script:**

```
// Example structure needed in build.gradle.kts
publishing {
    publications {
        create<MavenPublication>("maven") {
            from(components["java"])
            // Configure POM
        }
    }
    repositories {
        maven {
            name = "OSSRH"
            url =
uri("https://oss.sonatype.org/service/local/staging/deploy/maven2/")
            credentials {
                username = project.findProperty("ossrhUsername") as String?
                password = project.findProperty("ossrhPassword") as String?
            }
        }
    }
}
```

2.2 Release Documentation

Create Release Notes Template:

- □ `RELEASE_NOTES.md` template
- □ Include: new features, breaking changes, deprecations, bug fixes, migration guides

Initial v1.0.0 Release Notes Should Include:

- Complete feature list (DIDs, VCs, Wallets, Anchoring)
- Supported plugins (DID methods, KMS, blockchains)
- Quick start guide
- Architecture overview
- License information

2.3 Distribution Packages

Create:

- □ GitHub Releases with:
  - Source code archives (zip, tar.gz)
  - Release notes
  - Checksums (SHA256)
- □ Maven Central artifacts (automatic via CI/CD)
- □ Documentation site (GitHub Pages or separate hosting)

Phase 3: Community Infrastructure (Week 3-4)

3.1 Communication Channels

Set Up:

- □ **GitHub Discussions** - Q&A, feature requests, general discussion
  - Categories: General, Q&A, Ideas, Show and Tell
- □ **Discord/Slack** (optional) - Real-time community chat
- □ **Mailing List** (optional) - For announcements
- □ **Twitter/X Account** - `@TrustWeaveSDK` (or similar)
- □ **LinkedIn Company Page** - For enterprise visibility

3.2 Documentation Site

Options:

1. **GitHub Pages** (easiest, free)
   - □ Set up `gh-pages` branch
   - □ Use Jekyll or MkDocs
   - □ Custom domain: `docs.trustweave.io` (optional)
2. **Separate Site** (more control)
   - □ Use Docusaurus, VuePress, or similar
   - □ Host on Vercel/Netlify
   - □ Custom domain: `trustweave.io`

Documentation Checklist:

- □ API reference (auto-generated from KDoc)
- □ Getting started guide
- □ Plugin documentation
- □ Architecture diagrams
- □ Use case scenarios (already have 25+)
- □ Migration guides
- □ FAQ

3.3 Issue Templates

Create `.github/ISSUE_TEMPLATE/`:

- □ `bug_report.md`
- □ `feature_request.md`
- □ `plugin_request.md`
- □ `documentation_improvement.md`
- □ `question.md`

3.4 Pull Request Templates

Create `.github/pull_request_template.md`:

- □ Description of changes
- □ Type of change (bug fix, feature, docs, etc.)
- □ Testing checklist
- □ Documentation updates
- □ Breaking changes notice

Phase 4: Launch Strategy (Week 4-5)

4.1 Soft Launch (Week 4)

Target Audience: Early adopters, SSI community

Activities:

- □ Announce on personal/professional networks
- □ Post on relevant Reddit communities:
  - `r/selfhosted`
  - `r/kotlin`
  - `r/blockchain`
  - `r/selfsovereignidentity`
- □ Share on LinkedIn (Geoknoesis page)
- □ Reach out to 5-10 key influencers in SSI space
- □ Submit to awesome lists:
  - `awesome-kotlin`
  - `awesome-selfhosted`
  - `awesome-decentralized-id`

4.2 Public Launch (Week 5)

Launch Day Activities:

Morning (9 AM EST):

- □ Publish v1.0.0 release on GitHub
- □ Publish to Maven Central
- □ Update website/landing page
- □ Post on Twitter/X with announcement thread
- □ Post on LinkedIn with detailed announcement
- □ Post on Hacker News (Show HN)

Afternoon:

- □ Reach out to tech blogs for coverage:
  - Kotlin Weekly
  - SSI/DID newsletters
  - Blockchain development blogs

- □ Post on Dev.to with detailed tutorial
- □ Create YouTube video (optional, but high impact)
- □ Post on relevant Discord/Slack communities

Evening:

- □ Monitor GitHub for issues/questions
- □ Engage with early adopters
- □ Respond to all comments/questions

4.3 Launch Content

Blog Post Template:

```
Title: "Announcing TrustWeave: The Foundation for Decentralized Trust and Identity"

Sections:
1. The Problem (Why TrustWeave?)
2. What is TrustWeave?
3. Key Features
4. Quick Start Example
5. Use Cases
6. What's Next
7. Get Involved
```

Social Media Posts:

- Twitter: Thread with key features, code examples, use cases
- LinkedIn: Professional announcement with business value
- Reddit: Technical deep-dive with code examples

Phase 5: Post-Launch Growth (Months 2-6)

5.1 Community Building

Monthly Activities:

- □ Release monthly updates (even if minor)
- □ Host monthly community calls (optional)
- □ Feature community projects in README
- □ Create "Show and Tell" section in GitHub Discussions
- □ Respond to all issues within 48 hours
- □ Review all PRs within 1 week

Metrics to Track:

- GitHub stars
- Maven Central downloads
- GitHub Discussions activity
- PRs opened/merged
- Issues opened/closed
- Documentation page views

5.2 Content Marketing

Regular Content:

- □ Weekly blog posts (tutorials, use cases, deep dives)
- □ Video tutorials (YouTube)
- □ Conference talks (submit to SSI, Kotlin, blockchain conferences)
- □ Podcast appearances
- □ Guest posts on tech blogs

Content Ideas:

- "Building a Verifiable Credential System in 30 Minutes"
- "Comparing DID Methods: Which Should You Choose?"
- "TrustWeave vs. Other SSI Libraries"
- "Real-World Use Cases: Healthcare Credentials"
- "Architecture Deep Dive: Plugin System"

5.3 Partnership & Integration

Target Partnerships:

- □ Other SSI/DID projects (cross-promotion)
- □ Kotlin ecosystem projects
- □ Blockchain platforms (Algorand, Polygon, etc.)
- □ KMS providers (AWS, Azure, HashiCorp)
- □ Identity platforms (Auth0, Okta - for integration guides)

Integration Guides:

- □ Create integration guides for popular frameworks
- □ Build example applications
- □ Create starter templates

5.4 Developer Experience Improvements

Based on Community Feedback:

- □ Improve error messages
- □ Add more examples

- □ Create video tutorials
- □ Build interactive playground
- □ Improve IDE support (IntelliJ plugin?)

Phase 6: Monetization Strategy (Ongoing)

6.1 Commercial License Sales

Target Customers:

- Enterprises building production SSI systems
- SaaS companies embedding TrustWeave
- Government agencies
- Healthcare organizations
- Financial institutions

Sales Process:

- □ Clear pricing on website
- □ Contact form for enterprise inquiries
- □ Sales team training on TrustWeave
- □ Case studies from commercial customers

6.2 Professional Services

Offerings:

- □ Implementation consulting
- □ Custom plugin development
- □ Training workshops
- □ Architecture reviews
- □ Support contracts

6.3 Future SaaS Platform

Timeline: 6-12 months post-launch

Strategy:

- Build on open source success
- Offer managed hosting
- Enterprise features (SSO, advanced analytics)
- Premium support

Immediate Action Items (This Week)

Priority 1 (Critical for Launch):

1. Create `CONTRIBUTING.md` at root
2. Create `CHANGELOG.md` with v1.0.0 entry
3. Create `CODE_OF_CONDUCT.md`
4. Create `SECURITY.md`
5. Set up GitHub Actions CI/CD
6. Configure Maven Central publishing
7. Remove `-SNAPSHOT` and prepare v1.0.0 release

Priority 2 (Important):

1. Set up GitHub Discussions
2. Create issue/PR templates
3. Set up documentation site
4. Create launch blog post
5. Prepare social media content

Priority 3 (Nice to Have):

1. Set up Discord/Slack
2. Create video tutorial
3. Submit to awesome lists
4. Reach out to influencers

Success Metrics

3 Months:

- 500+ GitHub stars
- 50+ Maven Central downloads/week
- 10+ community contributors
- 5+ external plugins
- 20+ GitHub Discussions posts

6 Months:

- 2,000+ GitHub stars
- 200+ Maven Central downloads/week
- 25+ community contributors
- 15+ external plugins
- First commercial license sale
- Featured in 3+ tech blogs

12 Months:

- 5,000+ GitHub stars
- 1,000+ Maven Central downloads/week
- 50+ community contributors
- 30+ external plugins
- 10+ commercial customers
- Conference presentation
- Industry recognition

Risk Mitigation

Potential Risks:

1. **Low initial adoption**
   - Mitigation: Strong launch content, active community engagement
2. **Support burden**
   - Mitigation: Clear documentation, community moderation, FAQ
3. **Competing projects**
   - Mitigation: Focus on unique value (Kotlin, plugin system, domain-agnostic)
4. **License confusion**
   - Mitigation: Clear licensing documentation, FAQ section

Strategic Rationale

Why Open Source First?

1. **Developer Adoption**: Developers prefer libraries they can integrate and control, especially in decentralized identity
2. **Network Effects**: More users → more plugins → stronger ecosystem → more value
3. **Trust & Transparency**: Open source aligns with decentralized identity principles
4. **Multiple Revenue Paths**:
   - Commercial licensing for enterprise
   - Professional services and support
   - Managed SaaS later (hosted instances)
   - Premium plugins/features
5. **Lower Operational Costs**: No infrastructure to run initially
6. **Standards Alignment**: W3C-compliant libraries benefit from open ecosystems

Why Not SaaS First?

Challenges:

- Higher operational costs (infrastructure, scaling, support)
- Slower initial adoption (developers prefer self-hosted)
- Vendor lock-in concerns in a decentralized space

- Requires building a full platform, not just an SDK

Hybrid Approach (Best of Both)

1. Open source the SDK (core library)
2. Offer managed SaaS for those who want it
3. Provide premium enterprise features/commercial licensing
4. Build a marketplace for plugins and integrations

This is similar to what companies like Auth0, Stripe, and MongoDB do: open core with commercial offerings.

Next Steps

1. **Review and approve this strategy**
2. **Assign owners to each phase**
3. **Create project board with tasks**
4. **Set launch date** (recommend 4-6 weeks from now)
5. **Begin Phase 1 immediately**

Appendix: Reference Materials

Similar Projects to Study

- **MongoDB** - Open source with commercial licensing
- **Elastic** - Open source with commercial features
- **Auth0** - Open source SDK with SaaS platform
- **Stripe** - Open source libraries with commercial API
- **Supabase** - Open source with managed hosting

Key Contacts

- **Legal**: Review commercial license terms
- **Marketing**: Coordinate launch content
- **Engineering**: Set up CI/CD and publishing
- **Community**: Plan community engagement

Resources

- Open Source Guide
- Semantic Versioning
- Contributor Covenant
- Maven Central Publishing Guide

Document Version: 1.0
Last Updated: 2025
Owner: Geoknoesis LLC
Status: Active Planning