Snake Evolution - Documentation Standards & Guidelines v1.0

Document Header

Project: Snake Evolution Game
Document Type: Documentation Standards & Guidelines
Version: 1.0
Date: November 2025
Status: Reference Documentation
Prepared by: Senior Software Architect
Audience: All Team Members, Future Contributors

Executive Summary

This document establishes **standards and guidelines** for all documentation in the Snake Evolution project. It ensures consistency, quality, and maintainability across all documentation artifacts. All team members should follow these standards for new documentation and updates to existing documents.

Purpose

- **Who should read:** Everyone creating or updating documentation
- **What's covered:** Document structure, formatting, naming conventions, quality standards, review process
- **What's NOT covered:** Code standards (see code style guide)

1. Document Classification

1.1 Document Types

All Snake Evolution documentation falls into one of these categories:

Type	Purpose	Audience	Format	Owner
Product Documentation	What the system does	End users	PDF + HTML	Product
Technical Documentation	How to build it	Developers	PDF + Markdown	Architects
Process Documentation	How to work on it	Team members	Markdown	DevOps/Team Lead
Reference Documentation	System details	Support/DevOps	PDF + Wiki	Technical Lead

Type	Purpose	Audience	Format	Owner
Decision Documentation	Why decisions made	Future maintainers	Markdown + PDF	Architects

1.2 Mapping

```
PRODUCT DOCUMENTATION:
  ├─ PRD (Product Requirements)
  ├─ User Guide
  └─ Marketing materials

TECHNICAL DOCUMENTATION:
  ├─ Technical Analysis
  ├─ Functional Analysis
  ├─ API Reference
  └─ Architecture Diagrams

PROCESS DOCUMENTATION:
  ├─ Implementation Guide
  ├─ DevOps &amp;amp; CI/CD Guide
  ├─ Deployment &amp;amp; Operations
  └─ Testing Procedures

REFERENCE DOCUMENTATION:
  ├─ Maintenance &amp;amp; Support Guide
  ├─ Logging &amp;amp; Configuration
  ├─ Troubleshooting Guide
  └─ Performance Baselines

DECISION DOCUMENTATION:
  ├─ Design Decisions &amp;amp; Rationale
  ├─ ADRs (Architecture Decision Records)
  └─ Historical decisions log
```

2. Naming Conventions

2.1 Document File Naming

Standard Format:

```
&lt;PROJECT&gt;-&lt;DOCTYPE&gt;-v&lt;VERSION&gt;-&lt;LANGUAGE&gt;.pdf

Components:
  &lt;PROJECT&gt;     = SNAKEEVO (project identifier)
  &lt;DOCTYPE&gt;     = Document type abbreviation (see table)
  &lt;VERSION&gt;     = Semantic version (v1.0, v2.1)
  &lt;LANGUAGE&gt;    = EN (English), IT (Italian), etc.

Examples:
  SNAKEEVO-PRD-v2.1-EN.pdf
```

```
SNAKEEVO-USER-GUIDE-v1.0-EN.pdf
SNAKEEVO-API-REFERENCE-v1.0-EN.pdf
SNAKEEVO-IMPL-v2.0-EN.pdf
```

2.2 Document Type Abbreviations

Type	Abbreviation	Filename
Product Requirements	PRD	SNAKEEVO-PRD-v2.1-EN
User Guide	USER-GUIDE	SNAKEEVO-USER-GUIDE-v1.0-EN
API Reference	API-REFERENCE	SNAKEEVO-API-REFERENCE-v1.0-EN
Technical Analysis	TECH	SNAKEEVO-TECH-v2.1-EN
Functional Analysis	FUNC	SNAKEEVO-FUNC-v2.1-EN
Implementation Guide	IMPL	SNAKEEVO-IMPL-v2.0-EN
DevOps & CI/CD	DEVOPS	SNAKEEVO-DEVOPS-v2.0-EN
Deployment & Operations	DEPLOY	SNAKEEVO-DEPLOY-v2.0-EN
Logging & Configuration	LOGGING	SNAKEEVO-LOGGING-v2.0-EN
Maintenance & Support	MAINTENANCE	SNAKEEVO-MAINTENANCE-v1.0-EN
Design Decisions	DESIGN-DECISIONS	SNAKEEVO-DESIGN-DECISIONS-v1.0-EN
Documentation Standards	DOC-STANDARDS	SNAKEEVO-DOC-STANDARDS-v1.0-EN

3. Document Structure & Format

3.1 Standard Document Structure

All documents should follow this structure:

```
# [Document Title]

## Document Header
  - Project name
  - Document type
  - Version
  - Date
  - Status
  - Author(s)
  - Audience
  - Standard reference (if applicable)

## Executive Summary
  - 2-3 paragraph overview
  - Key findings or purpose
  - Who should read
  - Document scope
```

```
## 1. [Main Topic]
### 1.1 [Subtopic]
  - Detailed content
  - Multiple paragraphs
  - Tables, examples, code blocks

## 2. [Next Main Topic]
...

## Document Information
  - Document classification
  - Version number
  - Publication date
  - Language
  - Related documents

### Document Version History
  - Table of version changes

## Appendices (if needed)
```

3.2 Header Requirements

Every document must begin with a structured header:

```
# [Document Title]

## Document Header

**Project:** Snake Evolution Game
**Document Type:** [User Guide | API Reference | etc.]
**Version:** [1.0 | 2.1 | etc.]
**Date:** November 2025
**Status:** [Production Ready | Draft | Review | etc.]
**Prepared by:** [Author Name]
**Audience:** [Who should read this?]
**Standard:** [Standard followed, if applicable]
```

3.3 Executive Summary Requirements

Every document must start with a clear executive summary:

```
## Executive Summary

[1-2 sentences: What is this document?]

### Key Points
- [Main finding or concept #1]
- [Main finding or concept #2]
- [Main finding or concept #3]

### Document Purpose
```

```
- **Who should read:** [Target audience]
- **What's covered:** [Key topics]
- **What's NOT covered:** [Out of scope]
```

4. Formatting Standards

4.1 Heading Hierarchy

```
# Level 1 - Document Title (ONE per document)

## Level 2 - Main Sections (5-10 per document)

### Level 3 - Subsections (multiple per section)

#### Level 4 - Minor sections (use sparingly)

NEVER skip levels: ✖ Don't go directly from # to ###
```

4.2 Text Formatting

```
**Bold** — Key terms, decisions, important phrases
*Italic* — Emphasis, new concepts introduced first
`Code` — Function names, variables, commands
```

Usage:

- ✅ The **state machine** uses **atomic locking** to prevent *race conditions*
- ✖ ***Everything bold italic*** (too much emphasis)
- ✖ Bold text in entire sentences (use sparingly)

4.3 Lists

Unordered lists — When order doesn't matter:

```
- Item 1
- Item 2
- Item 3

(Use single-level only; no nesting)
```

Ordered lists — When order matters:

```
1. First step
2. Second step
3. Third step
```

```
(Use for procedures, sequences)
```

Definition lists — For term-definition pairs:

```
**Term 1** — Definition here
**Term 2** — Definition here
```

4.4 Tables

Use Markdown tables for comparisons, specifications, reference data:

```
| Column 1 | Column 2 | Column 3 |
|----------|----------|----------|
| Item 1   | Value 1  | Status 1 |
| Item 2   | Value 2  | Status 2 |
```

Rules:

- ✅ Use tables for: Features, requirements, specifications, comparisons
- ❌ Don't use tables for: Narrative text, complex hierarchies (use headings)
- ✅ Keep tables to 4-6 columns maximum for readability

4.5 Code Examples

Use fenced code blocks with language identifiers:

```
// JavaScript example
function example() {
  console.log('Hello');
}
```

```
# Bash/Shell example
npm install
npm start
```

```
Plain text or pseudocode
when language not specified
```

5. Content Guidelines

5.1 Writing Style

Voice & Tone:

- ✅ Professional, clear, direct
- ✅ Active voice preferred ("The engine manages state" not "State is managed")
- ✅ Second person for instructions ("Click button" not "The user clicks button")
- ✅ Avoid jargon or explain technical terms on first use

Example (Good):
> The state machine uses atomic locking to prevent race conditions. When a state transition is requested, the engine acquires a lock, validates the transition, executes hooks, and finally releases the lock.

Example (Poor):
> The thing with the state uses locking stuff. It does things with the state when stuff happens and stuff.

5.2 Paragraph Structure

Each paragraph should:

1. Start with **topic sentence** (main idea)
2. Provide **supporting detail** (explanation)
3. Give **examples or evidence** (concrete)
4. Connect to **larger context** (relevance)

Example:
> The **spatial hash algorithm provides O(1) collision detection** (topic). It divides the game grid into cells and tracks occupancy (explanation). When checking the snake head for collision, we only check adjacent cells instead of all body segments (example). This ensures consistent 60 FPS performance even with long snakes (relevance).

5.3 Paragraph Length

- Target: **4-6 sentences per paragraph**
- Minimum: **3 sentences** (else combine with previous)
- Maximum: **8 sentences** (else split)
- Rationale: Readable, not overwhelming

5.4 Citations & References

Internal references (to other sections in same document):

```
See Section 2.1 for details on state machine architecture.
```

External references (to other documents):

```
For implementation details, see Technical Analysis v2.1, Section 3.2.
```

Hyperlinks (in HTML/Markdown versions):

```
For more information, [see the User Guide](./SNAKEEVO-USER-GUIDE-v1.0-EN.pdf).
```

6. Review & Quality Standards

6.1 Review Checklist

Before finalizing any document, reviewers should verify:

Content Quality:

- ☐ All claims are accurate and verifiable
- ☐ Examples are concrete and relevant
- ☐ Information is current (not outdated)
- ☐ No contradictions with other documents
- ☐ Scope is clearly defined

Structure & Organization:

- ☐ Logical flow from introduction to conclusion
- ☐ Hierarchy is clear and consistent
- ☐ Cross-references are accurate
- ☐ All sections have introductory paragraphs
- ☐ Document feels complete (no obvious gaps)

Format & Style:

- ☐ Follows naming conventions
- ☐ Follows heading hierarchy rules
- ☐ Consistent formatting throughout
- ☐ Tables and lists are clear
- ☐ Code examples are syntactically correct

Language & Clarity:

- ☐ No typos or grammatical errors
- ☐ Sentences are clear and concise
- ☐ Jargon is defined or avoided
- ☐ Tone is professional and consistent
- ☐ Technical terms are used correctly

6.2 Review Process

```
AUTHOR → DRAFT
  ↓
  Author self-reviews (checklist)
  ↓
AUTHOR → PEER REVIEW
  ↓
  Technical peer reviews (at least 1)
  ↓
  Feedback incorporated
  ↓
AUTHOR → FINAL REVIEW
  ↓
  Team lead final approval
  ↓
APPROVED → PUBLISHED
  ↓
  Added to documentation repository
  Version tagged in Git
  Linked from central index
```

6.3 Approval Sign-off

Every document should have this section before publication:

```
## Document Approval

| Role | Name | Date | Approved |
|------|------|------|----------|
| Author | [Name] | [Date] | ✅ / ❌ |
| Reviewer 1 | [Name] | [Date] | ✅ / ❌ |
| Reviewer 2 | [Name] | [Date] | ✅ / ❌ |
| Lead | [Name] | [Date] | ✅ / ❌ |
```

7. Version Management

7.1 Versioning Scheme

Use **Semantic Versioning: MAJOR.MINOR.PATCH**

```
v1.0.0
 | | |
 | | └─ PATCH (typo fixes, clarifications)
 | └─── MINOR (new sections, expanded topics)
 └───── MAJOR (restructuring, significant updates)

Examples:
  v1.0 → Initial release
  v1.1 → Added troubleshooting section
  v2.0 → Restructured entire document
  v2.1 → Added API examples
  v2.1.1 → Fixed typos
```

7.2 Version History Table

Every document must include a version history:

```
### Document Version History

| Version | Date | Author | Changes |
|---------|------|--------|---------|
| 1.0 | 2025-11-06 | John Doe | Initial release with complete documentation |
| 1.1 | 2025-11-10 | Jane Smith | Added troubleshooting section |
| 2.0 | 2025-12-01 | John Doe | Restructured with new sections on monitoring |
```

7.3 Update Procedures

For minor updates (typos, clarifications):

- Increment PATCH version
- No need for formal review
- Update version history
- Republish

For content updates (new sections, expanded topics):

- Increment MINOR version
- Quick peer review (1 person)
- Update version history
- Republish and notify team

For major restructuring:

- Increment MAJOR version
- Full review process (2+ reviewers)
- Update version history
- Republish, announce, update cross-references

8. Documentation Maintenance

8.1 Review Schedule

```
Upon creation:
  - Full review process

After each release:
  - Verify accuracy
  - Update examples/screenshots
  - Add any new information discovered

Quarterly (every 3 months):
  - Full audit of all documents
  - Check for outdated information
  - Update version references
  - Verify all cross-references

Annually (every 12 months):
  - Major review of structure
  - Consider reorganization if needed
  - Prune duplicated content
  - Update contact information
```

8.2 Deprecation Process

When documentation becomes outdated:

```
⚠ **DEPRECATED:** This document is deprecated. Use [link to new document] instead.
This documentation will be removed on [date]. Please migrate to the new format.
```

When removing documentation:

```
1. Mark as deprecated (minimum 30 days)
2. Update all cross-references
3. Add redirect (if digital)
4. Archive old version
5. Remove from active index
6. Keep archived for historical reference
```

9. Documentation Template

Use this template for new documents:

```
# [Document Title]

## Document Header

**Project:** Snake Evolution Game
**Document Type:** [Type]
**Version:** 1.0
**Date:** November 2025
**Status:** Production Ready
**Prepared by:** [Your Name]
**Audience:** [Target Audience]
**Standard:** [If applicable, e.g., IEEE 1063-2001]

---

## Executive Summary

[2-3 paragraphs describing the document purpose and content]

### Document Purpose
- **Who should read:** [Audience]
- **What's covered:** [Key topics]
- **What's NOT covered:** [Out of scope]

---

## 1. Main Topic
### 1.1 Subtopic

[Content organized in sections...]

---

## [Additional main sections as needed]

---

## Document Information

| Item | Details |
|------|---------|
| **Document Type** | [Type] |
| **Version** | 1.0 |
| **Date Published** | November 2025 |
| **Language** | English |

### Document Version History

| Version | Date | Author | Changes |
|---------|------|--------|---------|
| 1.0 | 2025-11-06 | [Your Name] | Initial release |
```

```
---

**End of [Document Title]**

*Related documents: [Cross-references to related docs]*
```

10. Common Mistakes to Avoid

✖ Mistake	✓ Correction
Inconsistent heading levels	Always follow hierarchy: # → ## → ###
Overly long paragraphs (>10 sentences)	Break into multiple paragraphs
No topic sentences	Start each paragraph with main idea
Using all bold for emphasis	Use bold sparingly for key terms only
Nested bullet lists	Keep lists flat; use separate sections if complex
No version numbers	Always include version number
Outdated examples	Refresh examples quarterly
No table of contents	Add TOC for documents > 20 pages
Inconsistent formatting	Use formatting stylesheet
Missing document header	Always include project, author, date, status

11. Distribution & Access

11.1 Documentation Repository

```
GitHub Repository Structure:
/
├── docs/
│   ├── product/          (PRD, User Guide)
│   ├── technical/        (Architecture, API, Technical Analysis)
│   ├── processes/        (Implementation, DevOps)
│   ├── reference/        (Maintenance, Support)
│   └── decisions/        (Design Decisions, ADRs)
├── README.md            (Index of all documentation)
└── CONTRIBUTING.md      (How to contribute docs)
```

11.2 Documentation Index

Maintain a **central index** (README.md):

```
# Snake Evolution - Complete Documentation Index

## Product Documentation
```

```
- [PRD v2.1](./docs/product/PRD-v2.1-EN.pdf)
- [User Guide v1.0](./docs/product/USER-GUIDE-v1.0-EN.pdf)

## Technical Documentation
- [Technical Analysis v2.1](./docs/technical/TECH-v2.1-EN.pdf)
- [API Reference v1.0](./docs/technical/API-REFERENCE-v1.0-EN.pdf)

## Process Documentation
- [Implementation Guide v2.0](./docs/processes/IMPL-v2.0-EN.pdf)
- [DevOps &amp;amp; CI/CD v2.0](./docs/processes/DEVOPS-v2.0-EN.pdf)

## Reference Documentation
- [Maintenance Guide v1.0](./docs/reference/MAINTENANCE-v1.0-EN.pdf)

## Decision Documentation
- [Design Decisions v1.0](./docs/decisions/DESIGN-DECISIONS-v1.0-EN.pdf)

## Standards
- [Documentation Standards v1.0](./docs/standards/DOC-STANDARDS-v1.0-EN.pdf)
```

12. Document Information

Item	Details
Document Type	Documentation Standards & Guidelines
Version	1.0
Date Published	November 2025
Status	Active (enforce for all new documentation)
Language	English

Document Version History

Version	Date	Author	Changes
1.0	2025-11-06	Senior SA	Initial standards for Snake Evolution project

Appendix: Quick Checklist

Before submitting any document:

- ☐ Follows standard naming convention
- ☐ Has complete document header
- ☐ Has executive summary
- ☐ Follows heading hierarchy (no skipped levels)
- ☐ 4-6 sentences per paragraph (minimum 3)
- ☐ Tables used for comparisons/data (not narrative)

- ☐ All code examples are correct syntax
- ☐ Cross-references are accurate
- ☐ Has version history table
- ☐ Professional tone throughout
- ☐ No typos or grammatical errors
- ☐ Passed review checklist (Section 6.1)
- ☐ Has approval sign-offs

End of Documentation Standards & Guidelines

Apply these standards to all new documentation.
Existing documents should be updated to comply gradually.
Questions? Ask the Technical Lead.
[1] [2]

⁂

1. PRD-v2-1-EN.pdf
2. Functional-Analysis-EN.pdf