@ ? d: -> =>

Partylog

A Standard Notation for Group RPG Session Logging

Version 1.1.0

CC BY SA

Table of contents

1. Introduction **1**
1.1 Why “Partylog”? 1
1.2 What Partylog Does 2
1.3 How to Use This Notation 2
1.4 Quick Start: Your First Session 3
1.5 The Scribe 4
2. Digital vs Analog Formats **6**
2.1 Digital Format (Markdown) 6
2.2 Analog Format (Notebooks) 6
2.3 Format Examples 7
3. Core Notation **9**
3.1 Player Actions 9
3.2 World Events 11
3.3 Mechanics and Resolution 12
3.4 Consequences 16
3.5 Complete Action Sequences 17
4. Optional Layers **20**
4.1 Persistent Elements 20
4.2 Progress Tracking 27
4.3 Random Tables & Generators 28
4.4 Narrative Excerpts 31
4.5 Meta Notes 32
5. Optional Structure **35**
5.1 Campaign Header 35
5.2 Session Header 36
5.3 Scene Structure 37
5.4 Session End 41
5.5 Interlude 42
6. Complete Examples **44**
6.1 Minimal Shorthand Log 44
6.2 Hybrid Digital Format 44
6.3 Analog Notebook Format 45
6.4 Complete Session Log (Digital) 46
6.5 Complete Session Log (Analog) 50

6.6 Combat Example (Detailed) 53
7. Best Practices 56
7.1 Good Practices ✓ 56
7.2 Bad Practices ✗ 57
8. Templates 59
8.1 Campaign Template (Digital) 59
8.2 Campaign Template (Analog) 60
8.3 Session Template 60
8.4 Quick Scene Template 62
8.5 Combat Template 62
9. Adapting to Your System 64
9.1 System-Specific Roll Notation 64
9.2 Handling Edge Cases 66
Appendices 68
A. Symbol Reference 68
B. Oracle Notation for GM-less Play 70
C. Symbol Design Philosophy 72
Credits & License 74

1. Introduction

If you've ever been the one taking notes at a TTRPG session, you know the problem: the bard is arguing with the shopkeeper, the rogue just triggered a trap, the GM is describing a collapsing ceiling, and you're scribbling furiously trying to capture any of it.

Maybe you've tried free-form journaling (becomes a wall of text), pure prose (loses the mechanics), or bullet points (unreadable two weeks later). Partylog offers a different approach: a **lightweight shorthand** that captures the essential game elements — who did what, what the dice said, what happened — while leaving room for as much (or as little) narrative as you want.

1.1 Why "Partylog"?

Partylog is a **fork of Lonelog**, a notation system designed for solo RPG session logging. Lonelog's core philosophy — separate mechanics from fiction, stay compact at the table, scale from one-shots to long campaigns — works beautifully for solo play. But group play introduces challenges that solo notation doesn't face: multiple players acting simultaneously, a GM pushing events into the fiction, real-time chaos that no single scribe can fully capture.

Partylog adapts the Lonelog foundation for group play. The name mirrors the structure: *Party* (group play) + *log* (session record). Where Lonelog centers the lone player and their oracle, Partylog centers the party and the GM.

If you know Lonelog, you'll feel at home. The tag system, scene structure, progress tracking, and formatting philosophy are shared DNA. The core symbols shift to reflect group dynamics — attributed actions, GM-initiated events — but the underlying logic is the same.

If you don't know Lonelog, don't worry. This document is fully standalone. You don't need to read anything else to use it.

1.2 What Partylog Does

Whether you're playing *D&D*, *Blades in the Dark*, *Call of Cthulhu*, or any other tabletop RPG with a group and a GM, this notation helps you:

- **Record what happened** without drowning in prose
- **Track ongoing elements** like NPCs, locations, and plot threads
- **Share your sessions** with players who missed the game or want a recap
- **Review past sessions** and quickly find the detail everyone forgot

The notation is designed to be:

- **Flexible** — usable across different systems and formats
- **Layered** — works as both quick shorthand or expanded narrative
- **Searchable** — tags and codes make it easy to track NPCs, events, and locations
- **Format-agnostic** — works in digital markdown files or analog notebooks
- **Fast** — a scribe should be able to use it in real time without falling behind

1.3 How to Use This Notation

Think of this as a **toolbox, not a rulebook**. The system is fully modular: grab what works for you and leave the rest behind.

At its core are just **four elements** (see *Section 3: Core Notation*). They're carefully chosen to avoid conflicts with markdown formatting and to mirror the natural flow of group play:

- `@` for player actions
- `!` for GM / world events
- `d: ->` for mechanics and their results
- `=>` for consequences

That's it. **Everything else is optional.**

Scenes, campaign headers, session headers, threads, clocks, narrative excerpts — these are all enhancements you can add when they serve your logging. Want to track a long campaign? Add campaign headers. Need to follow complex plots? Use thread tags. Logging a casual one-shot? Stick to the core symbols.

Think of it as concentric circles:

- **Core Notation** (required): Actions, World Events, Resolutions, Consequences
- **Optional Layers** (add as needed): Persistent Elements, Progress tracking, Notes, etc.
- **Optional Structure** (for organization): Campaign Header, Session Header, Scenes

Start small. Try the core notation for one scene. If it clicks, great — keep going. If you need more, layer in what helps. Your notes should serve your table, not the other way around.

1.4 Quick Start: Your First Session

Never used notation before? Here's everything you need:

```
S1 *Your starting scene*
@(Name) Action a player takes
d: the roll result -> Success or Fail
=> What happens as a result

! Something the GM introduces
=> What that means for the party
```

That's it! Everything else is optional. Try this for one scene and see how it feels.

Quick Start Example

```
S1 *Dark alley, midnight*
@(Kael) Sneak past the guard
d: Stealth d20+5=8 vs DC 14 -> Fail
=> Kicks a bottle. Guard turns!

! Guard draws his blade and advances
@(Sable) Cast Sleep before he can shout
d: Spell DC 13, guard WIS save d20=9 ->
Fail
=> The guard crumples. Silence holds —
for now.
```

1.5 The Scribe

In solo play, the player is the writer. In group play, someone has to take notes while everyone else plays. This person is the **scribe**.

The scribe doesn't need to capture everything. As a player-character or diegetic observer at the table, the scribe only logs what is spoken aloud or revealed to the party. Secret GM-to-player whispers or private notes are omitted from the group log until they become common knowledge in the fiction.

Priorities, in order:

1. **Actions and rolls** — who did what, what the dice said
2. **Consequences** — what happened as a result
3. **World events** — what the GM introduced
4. **Key dialogue** — only the memorable lines
5. **Narrative color** — atmosphere, description, mood

The first three are the skeleton. The last two are flesh — add them when you can, skip them when the game moves too fast.

Practical tips for scribing:

- Use shorthand aggressively during play; expand later if you want
- If you fall behind, skip narrative and capture mechanics — you can reconstruct fiction from rolls, not the other way around
- It's fine to write up notes post-session from memory and whatever you jotted down
- Rotating the scribe between sessions keeps the burden fair
- If multiple people log digitally, one person can own the "canonical" log while others contribute fragments

Scribe attribution (optional):

```
=== Session 7 ===
[Date]     2025-11-15
[Scribe]   Jordan
```

2. Digital vs Analog Formats

This notation works in **both digital markdown files and analog notebooks**. Choose the format that suits your table.

2.1 Digital Format (Markdown)

In digital markdown files:

- **Campaign metadata** → YAML front matter (top of file)
- **Campaign Title** → Level 1 heading
- **Sessions** → Level 2 headings (`## Session 1`)
- **Scenes** → Level 3 headings (`### S1`)
- **Core notation and tracking** → Code blocks for easy copying/parsing
- **Narrative** → Regular prose between code blocks

> ***Note:*** *Always wrap notation in code blocks (```) when using digital markdown. This prevents conflicts with Markdown syntax and ensures symbols like* `=>` *render correctly.*

2.2 Analog Format (Notebooks)

In paper notebooks:

- Write headers and metadata directly as shown
- Core notation works identically but without code fences
- Use the same symbols and structure
- Brackets and tags help scanning paper pages

2.3 Format Examples

Digital markdown

```
## Session 1
*Date: 2025-09-03 | Duration: 3h |
Scribe: Jordan*

### S1 *Tavern common room, evening*

```
@(Kael) Ask the barkeep about the missing
merchant
d: Persuasion d20+3=16 vs DC 12 ->
Success
=> He leans in and whispers about strange
lights at the old mill.

! A hooded figure in the corner stands
and leaves abruptly.
=> [N:Hooded Figure|mysterious|left
quickly]
```
```

Analog notebook

```
=== Session 1 ===
Date: 2025-09-03 | Duration: 3h | Scribe:
Jordan
```

```
S1 *Tavern common room, evening*

@(Kael) Ask the barkeep about the missing
merchant
d: Persuasion d20+3=16 vs DC 12 ->
Success
=> He leans in and whispers about strange
lights at the old mill.

! A hooded figure in the corner stands
and leaves abruptly.
=> [N:Hooded Figure|mysterious|left
quickly]
```

Both formats use identical notation — only the wrapping differs.

3. Core Notation

This is the heart of the system. Everything else in this document is optional, but these core elements are what make the notation work.

There are four elements to remember, and they mirror the natural flow of group play: a player acts or the GM introduces something, it gets resolved with mechanics, and you record what happens as a result.

3.1 Player Actions

The `@` symbol represents a player character acting in the game world. In group play, actions need **attribution** — who is doing this?

Format:

```
@(Name) Action description
```

Examples:

```
@(Kael) Pick the lock on the cellar door
@(Sable) Cast Detect Magic on the altar
@(Mira) Charge the orc chieftain
@(Finn) Persuade the merchant to lower
his price
```

3.1.2 Collaborative Actions (Lead/Assist)

In group play, characters often work together. Use the `>` connector to distinguish between the **Leader** (the one making the primary roll) and the **Assistant** (the one providing a bonus or help).

```
@(Mira > Kael) Mira assists Kael in
climbing the wall
d(Mira > Kael): Athletics d20+2 (Mira)
helps Kael -> Success
=> Mira gives Kael a boost. He reaches
the ledge.
```

This syntax clearly identifies who is helping whom, which is essential for tracking "Help" actions or system-specific assistance mechanics.

3.1.3 Implicit Attribution

When one character is clearly the focus of a scene, you can drop the name:

```
(note: Kael is acting alone this scene)
@ Pick the lock
@ Slip inside
@ Search the desk
```

But when multiple PCs are present and active, always attribute. When in doubt, attribute.

Multiple Characters Acting Together

```
@(Kael+Mira) Force open the heavy door
d: Athletics (Kael d20+2=14, Mira
d20+4=18, take highest) -> 18 vs DC 15 ->
Success
=> The door groans open.
```

3.2 World Events

The ! symbol represents the game world acting — things the GM introduces, declares, or narrates that aren't in response to a specific player action. Think of it as the world *pushing* events into the fiction.

Format:

```
! Event description
```

Examples:

```
! The ceiling begins to crack
! A messenger bursts through the tavern
door
! Orc reinforcements pour from the side
tunnel
! The ship lurches — a storm is coming
! The NPC turns hostile without warning
```

When to use !:

- GM describes something happening to or around the party

- Environmental events (weather, terrain, time passing)
- NPC-initiated actions against the party
- Plot developments the GM introduces
- Anything the party didn't cause or ask for

When NOT to use `!`:

- Direct consequences of player actions (use `=>` instead)
- NPC reactions during combat turns (use `@(NPC)` — see §3.5)
- Atmospheric description that doesn't change the game state (use prose)

The line between `!` and `=>` can be blurry. A good rule of thumb: if it flows directly from a player's roll, it's `=>`. If the GM is introducing something new, it's `!`.

```
@(Kael) Sneak past the guards
d: Stealth d20+5=8 vs DC 14 -> Fail
=> Kicks a barrel, noise echoes
(consequence of the action)

! The captain of the guard emerges from
the gatehouse

(new element the GM introduces)
```

3.3 Mechanics and Resolution

Once an action is declared (`@`) or a world event triggers a response, the game's mechanics determine what happens. The `d:` prefix marks a roll or rule resolution. The `->` arrow declares the outcome.

Format:

```
d: [roll or rule] -> outcome
```

Examples:

```
d: d20+Lockpicking=17 vs DC 15 -> Success
d: 2d6=8 -> Strong Hit (Ironsworn)
d: d100=42 -> Partial Success
d: 4d6=6,5,4,2 (take highest=6) ->
Critical (FitD)
```

Attributed Rolls

When it matters who rolled:

```
@(Kael) Attack the orc
d(Kael): d20+5=18 vs AC 15 -> Hit

@(Sable) Counterspell
d(Sable): d20+4=16 vs DC 14 -> Success
```

Attribution on the `d:` line is optional if the preceding `@(Name)` makes it clear. Use `d(Name):` when rolls happen in quick sequence and clarity matters — especially in combat.

Comparison Shorthand

When comparing rolls to target numbers:

Notation	Meaning	Example
`vs`	Neutral "against"; the outcome says who won	`d: 18 vs DC 15 -> Success`
`≥` or `>=`	Meets or beats the target	`d: 18≥15 -> S`
`<`	Falls short of the target	`d: 8<14 -> F`

Write the comparison so it is true, and put `S` (Success) or `F` (Fail) after `->`, as in core Lonelog.

Contested Rolls

Common in group play — PC vs. NPC, PC vs. PC:

```
@(Kael) Grapple the assassin
d: Athletics (Kael d20+2=15 vs Assassin
d20+4=12) -> Kael wins
=> Pins the assassin to the ground.

@(Kael) Arm-wrestle @(Mira) for the last
ale
d: STR (Kael d20+1=14 vs Mira d20+3=17) -
> Mira wins
=> Mira slams his hand down. The table
cheers.
```

Saving Throws and Reactions

When the GM calls for a save or check:

```
! Poison gas fills the chamber
d(Kael): CON save d20+2=14 vs DC 13 ->
Success
d(Sable): CON save d20-1=8 vs DC 13 ->
Fail
=> Kael holds his breath. Sable chokes —
[PC:Sable|poisoned]
```

3.3.2 Roll Context

When a roll draws on specific tags, traits, or situational modifiers, list them before the dice: `+[…]` for what helps, `-[…]` for what hinders. This records which factors influenced the roll without changing persistent states.

```
@(Mira) Attack the Ogre
d: +[flanking] -[wounded] d20+6=21 vs AC
16 -> Hit

@(Sable) Persuade the Guard
d: +[trait: silver tongue] -[suspicious]
2d6=9 -> Success
```

Roll context means "these factors are **active for this roll**." This is especially useful in group play for tracking why a player has advantage or what situational modifiers the GM applied.

3.4 Consequences

Record the narrative result after rolls and events using `=>`. The double arrow shows consequences flowing forward from actions and resolutions.

```
=> The door creaks open, but the noise
echoes through the hall.
=> The guard spots Kael and raises the
alarm.
=> Sable finds a hidden diary with a
crucial clue.
```

Multiple Consequences

Chain multiple consequence lines for cascading effects:

```
@(Mira) Charge through the barricade
d: Athletics d20+4=22 vs DC 15 -> Success
=> The barricade splinters
=> But the noise alerts the entire camp
=> [Clock:Alert 3/6]
```

GM-Narrated Consequences

Sometimes the GM describes effects that aren't tied to a specific roll:

```
! As you rest, the wound festers
=> [PC:Kael|HP-2|infected]
```

3.5 Complete Action Sequences

Here's how the core elements combine in group play:

Player-Driven Sequence

```
@(Kael) Pick the lock
d: Thieves' Tools d20+7=19 vs DC 15 ->
Success
=> The door creaks open, revealing a
dusty archive.
```

World-Driven Sequence

```
! The bridge starts to give way underfoot
d(Mira): DEX save d20+1=14 vs DC 12 ->
Success
d(Sable): DEX save d20+0=9 vs DC 12 ->
Fail
=> Mira leaps to safety. Sable falls —
catches a beam! Dangling.
```

Combined Sequence

```
@(Kael) Sneak into the guardroom
d: Stealth d20+5=8 vs DC 14 -> Fail
=> Bumps a shelf, bottles rattle.
[Clock:Alert 2/6]
```

```
! The sergeant looks up from his desk
@(Sable) Quick — cast Minor Illusion of a
cat knocking things over
d: Spell DC 13, sergeant WIS save d20+1=7
-> Fail
=> He mutters "damn cat" and goes back to
his paperwork.
```

Combat Sequence

In combat, player and NPC actions alternate. Use `@(NPC)` for enemy turns — the GM is narrating them, but they're character actions, not world events:

```
[COMBAT]
(init: Mira 19, Orc Chief 16, Kael 14,
Sable 11)

Rd1
@(Mira) Reckless Attack with greatsword
d: d20+6 (adv)=21 vs AC 16 -> Hit,
2d6+4=14 dmg
=> Heavy slash across the orc's chest.
[F:Orc Chief|HP-14]

@(Orc Chief) Retaliates at Mira (she's
reckless — advantage)
d: d20+6 (adv)=22 vs AC 18 -> Hit,
1d12+4=13 dmg
```

```
=> Brutal counter. [PC:Mira|HP-13]

@(Kael) Sneak attack from the shadows
d: d20+7=23 vs AC 16 -> Hit, 1d6+3d6+3=18
dmg
=> Blade finds the gap in his armor.
[F:Orc Chief|HP-18]

@(Sable) Firebolt
d: d20+5=12 vs AC 16 -> Miss
=> Flames scorch the wall behind him.

Rd2
...
[/COMBAT]
```

The `@(NPC)` convention works because NPCs are *acting* — making attacks, casting spells, taking turns. It keeps combat legible: every turn is an `@`, every roll is a `d:`, every result is `=>`. The `!` symbol is reserved for events *outside* the turn structure — reinforcements arriving, the floor collapsing mid-fight, environmental changes.

```
@(Orc Chief) Bellows for reinforcements
! Three more orcs burst through the side
door
=> [F:Orcs|×3|HP 11 each|pos:Close]
```

4. Optional Layers

You've got the basics — actions, world events, rolls, and consequences. That's enough for simple play. But longer campaigns often need more: NPCs who reappear, plot threads that weave through sessions, progress that accumulates over time.

This section covers the **tracking elements** that help you manage complexity. They're all optional. Pick and choose based on what your campaign needs.

4.1 Persistent Elements

As your campaign grows, certain things stick around: NPCs who reappear, locations you return to, ongoing threats, story questions that span sessions. Tags let you track them consistently.

Format: Brackets, a type prefix, a name, and optional details.

Why use tags?

- **Searchability**: Find every scene where an NPC appears
- **Consistency**: Reference elements the same way every time
- **Status tracking**: See how elements change over time
- **Memory aid**: Remind the table of details weeks later

You don't need to tag everything — only what matters to your campaign.

Logical Tag Flow (Optional Recommendation): For long campaigns, a consistent internal order helps both the human eye and digital search tools (like `grep` or search filters). A recommended flow is: `[Identity | Narrative Status | Mechanical Stats]`

Example: `[N:Baron Holt | captured, angry | HP 12/45]`

4.1.1 NPCs

```
[N:Baron Holt|hostile|powerful]
[N:Innkeeper|friendly|knows rumors]
[N:Assassin|unknown|hired by someone]
```

Updating NPC tags:

When an NPC's status changes:

- Restate with new tags: `[N:Baron Holt|captured|wounded]`
- Show just the change: `[N:Baron Holt|captured]`
- Use explicit updates: `[N:Baron Holt|hostile→cooperative]`
- Add `+` or `-`: `[N:Baron Holt|+captured]` or `[N:Baron Holt|-hostile]`

4.1.2 Player Characters

Multiple PCs are standard in group play. Establish them early:

```
[PC:Kael|HP 28/28|Class:Rogue|
Player:Alex]
[PC:Sable|HP 18/18|Class:Wizard|
Player:Jordan]
[PC:Mira|HP 34/34|Class:Fighter|
Player:Sam]
```

The `Player:` field is optional but useful when sharing logs or when players have multiple characters.

Updating PC stats:

```
[PC:Kael|HP-5]
[PC:Kael|HP 23]
[PC:Sable|poisoned]
[PC:Mira|HP+10|-wounded]
```

Tags accumulate, as in core Lonelog: `HP-5` lowers HP by 5, `HP 23` sets it, `poisoned` adds a condition, and `-wounded` removes one. Use `old>new` to replace a value.

Advancement events:

Use `[Advance:]` to record when a character levels up or gains a significant progression milestone. This makes advancement searchable across a long campaign log.

```
[Advance:Kael|Rogue 6|+Expertise: Thieves
Tools, Stealth]
[Advance:Sable|Wizard 6|+3rd level spell
slots]
[Advance:Mira|Fighter 6|+Extra Attack 2]
```

Follow up with a `[PC:]` update to reflect the new state:

```
[PC:Kael|Level 6|HP 34]
```

`[Advance:]` works system-agnostically: use whatever your game tracks — levels, XP thresholds, advancement moves (PbtA), or experience dots (FitD).

4.1.3 Party-Level Tracking

Some resources and states belong to the group, not any individual. The tag's name is the party's — whatever the table calls itself, or simply `Crew`:

```
[Party:Crew|Gold 150|Rations 10|
Wagon:intact]
[Party:Crew|Reputation:feared in
Northport]
[Party:Crew|Quest:Retrieve the Sunstone]
```

Updating:

```
[Party:Crew|Gold-30]
[Party:Crew|Rations-3]
[Party:Crew|Wagon:intact>damaged]
```

The party spends 30 gold, uses three days' rations, and the wagon is damaged in an ambush.

4.1.4 Factions

Factions are persistent organizations — guilds, noble houses, city watch, cults, criminal networks — that operate independently of individual NPCs. Track their power and the party's standing with them.

```
[Faction:City Watch|tier:2|
standing:neutral]
[Faction:Thieves Guild|tier:3|
```

```
standing:allied]
[Faction:Baron Holt's House|tier:4|
standing:hostile]
```

`tier` reflects the faction's relative power in the fiction (optional). `standing` reflects the party's current relationship with them.

Updating faction standing:

```
[Faction:City Watch|
standing:neutral>suspicious]
[Faction:Thieves Guild|+owes us a debt]
[Faction:Baron Holt's House|-hostile|
+hunting us]
```

4.1.5 Locations

```
[L:Thornfield Manor|abandoned|haunted]
[L:Docks District|dangerous|night]
[L:The Rusty Anchor|tavern|friendly]
```

4.1.6 Events

An event tag marks a notable thing that happened, so the campaign's turning points are easy to find:

```
[E:Bridge Collapse]
[E:Baron Holt Exposed]
```

Anything that builds up over time — a ritual, the city's alert level, a dragon's approach — is a clock or track instead:

```
[Clock:Cultist Ritual 3/8]
[Clock:City Alert 2/6]
[Clock:Dragon Approach 5/10]
```

4.1.7 Story Threads, Goals & Quests

While **Threads** track general plot lines, **Goals** and **Quests** represent the party's shared objectives and milestones.

```
[Thread:Find the Missing Merchant|Open]
[Goal:Escort the Prince to Northport|
Active]
[Quest:The Sunstone Conspiracy|Main]
```

4.1.8 Loot (Unassigned Items)

Group play often results in "party loot" that hasn't been claimed by a specific PC yet. Use the `[Loot: ...]` tag to track these items in a shared stash.

```
! The party finds a hidden coffer
=> [Loot: Ancient Silver Ring |
unassigned]
=> [Loot: Potion of Healing x2]

... later ...
```

```
@(Mira) Take the silver ring from the
stash
=> [Loot: -Ancient Silver Ring]
=> [PC:Mira | +Silver Ring]
```

4.1.9 Reference Tags

To reference a previously established element without restating tags, use the `#` prefix:

```
[N:Baron Holt|hostile|powerful]
(first mention — full details)

... later in the log ...

[#N:Baron Holt]
(reference — look back for context)
```

When to use reference tags:

- First mention: Full tag with details `[N:Name|tags]`
- Later mentions in same scene: Optional
- Later mentions in different scenes/sessions: Use `[#N:Name]`
- Status changes: Drop the `#` and show new tags `[N:Name|new_tags]`

4.1.10 Tag Categories

When a tag holds values across distinct types, group them with category prefixes. The category name is followed by `:` and then a comma-separated list of values.

```
[PC:Kael|trait:agile,curious|
status:wounded|stat:HP 23]
[N:Baron Holt|status:captured,angry|
info:knows about the cult]
```

Categories are freeform — use labels that fit your system (e.g., `role:`, `condition:`, `stat:`).

4.1.11 Multi-Line Tags

For characters or locations with many tags, break them across lines using the same `|` separator.

```
[PC:Mira
  | trait: brave, reckless, loyal
  | status: bandaged
  | stat: HP 27, AC 18
  | gear: Greatsword, Shield
]
```

Multi-line and single-line forms are equivalent. Multi-line is best for "character sheet" summaries at the start of a session or when an NPC becomes very complex.

4.2 Progress Tracking

Three formats handle different types of progression:

Clocks (fill up toward something happening — usually bad):

```
[Clock:Ritual 5/12]
[Clock:Suspicion 3/6]
```

Tracks (progress toward a goal — usually good):

```
[Track:Heist Plan 3/8]
[Track:Investigation 6/10]
```

Timers (count down toward zero — urgency):

```
[Timer:Dawn 3]
[Timer:Reinforcements 5]
```

The difference? Clocks and tracks both go up, but clocks are threats (bad when full) and tracks are progress (good when full). Timers go down and create urgency.

4.3 Random Tables & Generators

When the GM rolls on a table or uses a generator, record it to maintain transparency and reproducibility. As in core Lonelog, `tbl:` covers any random result that gives content, from a single roll to a multi-step generator:

```
tbl: Loot d100=42 -> A broken sword
tbl: Weather d20=15 -> Thunderstorm at
midnight
tbl: Random NPC d8=3,d10=7 -> Gruff /
```

```
Pilot
tbl: Encounter d100=78 -> Dragon patrol,
distant
```

4.3.1 Inline Table Definitions

When the GM creates a table on the fly or uses a custom one, embed it directly in the log to make it self-contained.

```
tbl: Room Contents (d4)
  1: Empty — eerie silence
  2: Hidden loot (DC 15 to find)
  3: Sleeping guard
  4: Environmental hazard

! The party enters the dark chamber
tbl: Room Contents d4=3 -> Sleeping guard
=> A guard is slumped against the far
wall, snoring.
```

4.3.2 Filtered Option Sets

For curated lists where you pick or draw from specific options relevant to the scene.

```
tbl: Mood [Tense, Melancholic, Hopeful,
Uncanny]
tbl: Mood -> Uncanny
=> A thick, unnatural fog rolls in.
```

```
tbl: Available Leads [The dockworker's
tip, The torn letter, The library
records]
tbl: Available Leads -> The torn letter
```

4.3.3 Multi-Line Result Blocks

For generators that produce compound results (e.g., an NPC with several traits).

```
tbl: Random NPC
  Role: d6=2 -> Merchant
  Trait: d6=6 -> Obsessive
  Want: d6=4 -> Knowledge
=> [N:The Collector|merchant|obsessive|
seeks forbidden texts]
```

Integrating with world events:

```
! Random encounter on the road
tbl: Road Encounter d100=67 -> Merchant
caravan under attack
=> Screams ahead — a merchant's wagon is
being raided by goblins.
```

4.4 Narrative Excerpts

The shorthand captures everything mechanically. But sometimes the fiction demands more.

Inline prose (short descriptions):

```
=> The chamber reeks of mildew and old
blood. The walls are covered in
scratches.
```

Dialogue (conversations worth recording):

```
PC(Kael): "I don't trust him."
PC(Sable): "We don't have a choice."
N(Baron): "You'll do as I say, or face
the consequences."
PC(Mira): [stands, hand on sword] "Try
us."
```

Use `PC(Name)` for player characters and `N(Name)` for NPCs.

Long narrative blocks (found documents, GM read-aloud text):

```
\---
The GM reads from the scroll:
"When the three moons align, the seal
will weaken. Those bound to the
covenant must gather at the stone circle,
or the pact is broken and
```

```
the darkness returns."
---\
```

The `\---` and `---\` markers separate in-fiction content from your log. The asymmetric delimiters prevent conflicts with Markdown horizontal rules.

How much narrative should you write? Only as much as the scribe can capture and the table finds useful. In group play, speed matters more than completeness.

4.5 Meta Notes

Step outside the fiction with parenthetical notes for out-of-character (OOC) commentary, rulings, and table-level status.

General Notes:

```
(note: Sam had to leave early — Mira
stays at camp for the rest of the
session)
(reflection: the heist scene was
incredible, everyone was on point)
```

Rulings:

Use the `rule:` prefix inside parentheses to record GM rulings or house rules. This creates a searchable index of "Table Law" for future reference.

```

```

```
(rule: Flanking provides a +2 bonus
instead of advantage)
(rule: Potions can be used as a Bonus
Action)
(rule: Healing Spells do max healing out
of combat)
```

Retrospective notes:

Use `(post: ...)` when a section was written after the session rather than live — reconstructed from memory or rough notes. Helps readers (and future you) calibrate how reliable the details are.

```
(post: S18-S19 reconstructed from memory
— sequence may be approximate)
(post: this scene was written up the next
day — dialogue is paraphrased)
```

OOC Status:

Use the `[OOC: ...]` tag for table-level status flags such as safety tools, breaks, or player presence. This avoids confusion with the `tbl:` prefix.

```
[OOC: Break | 15 mins]      (The table is
taking a break)
[OOC: X-Card | spiders]     (Safety tool
invoked)
[OOC: Mira AFK]             (Player is
away, character is inactive)
```

Use sparingly. The game is the thing — meta notes are margin comments.

Safety Tool Protocol:

When a safety tool (X-Card, Lines/Veils, etc.) is used, the log should reflect the narrative adjustment to ensure continuity for future reading. Use the `(safety: ...)` note.

```
(safety: X-Card | Scene Ended | Spiders)
(safety: Rewind | to S14 | Alternate
outcome follows)
(safety: Fade to Black | skipped to S16)
```

5. Optional Structure

Structure helps navigation and signals boundaries. Use what helps you stay oriented without slowing you down.

5.1 Campaign Header

The "cover page" of your campaign log. Useful when running multiple campaigns, sharing logs, or returning after a break.

Digital format (YAML front matter):

```
title: The Sunstone Conspiracy
ruleset: D&D 5e
genre: Dark fantasy / intrigue
gm: Roberto
players: Alex (Kael), Jordan (Sable), Sam
(Mira)
pcs:
  - Kael [PC:Kael|Rogue 5|HP 28]
  - Sable [PC:Sable|Wizard 5|HP 18]
  - Mira [PC:Mira|Fighter 5|HP 34]
start_date: 2025-09-03
last_update: 2025-11-15
themes: Betrayal, power, loyalty
tone: Gritty, morally grey
notes: Homebrew world, modified rest
rules
```

Analog format:

```
=== Campaign Log: The Sunstone Conspiracy
===
[Title]        The Sunstone Conspiracy
[Ruleset]      D&D 5e
[Genre]        Dark fantasy / intrigue
[GM]           Roberto
[Players]      Alex (Kael), Jordan
(Sable), Sam (Mira)
[PCs]          Kael [Rogue 5], Sable
[Wizard 5], Mira [Fighter 5]
[Start Date]   2025-09-03
[Last Update]  2025-11-15
[Themes]       Betrayal, power, loyalty
[Tone]         Gritty, morally grey
[Notes]        Homebrew world, modified
rest rules
```

Optional fields: `[Setting]`, `[Inspiration]`, `[Safety Tools]`

5.2 Session Header

Marks boundaries between play sessions and provides context.

Digital:

```
## Session 7
*Date: 2025-11-15 | Duration: 3h30 |
Scenes: S18-S22*
*Players: Alex (Kael), Jordan (Sable),
Sam (Mira)*
```

```
*Scribe: Jordan*

**Recap:** The party infiltrated Baron
Holt's estate but were discovered.
Escaped through the sewers. Mira was
wounded.

**Goals:** Regroup, find a healer, plan
next move against the Baron.
```

Analog:

```
=== Session 7 ===
[Date]         2025-11-15
[Duration]     3h30
[Players]      Alex (Kael), Jordan
(Sable), Sam (Mira)
[Scribe]       Jordan
[Recap]        Infiltrated Baron's
estate, discovered, escaped via sewers.
               Mira wounded.
[Goals]        Regroup, find healer, plan
next move.
```

Optional fields: `[Absent]` (players not present), `[Mood]`, `[Threads]` (active this session), `[Notes]`

5.3 Scene Structure

Scenes are the basic unit of play within a session.

Digital format:

```
### S18 *Sewer tunnels beneath the
estate*
```

Analog format:

```
S18 *Sewer tunnels beneath the estate*
```

5.3.1 Sequential Scenes (Standard)

```
S18 *Sewer tunnels, night*
S19 *Safe house in the Docks District*
S20 *The Rusty Anchor tavern, dawn*
```

Default for linear play. Start at S1 each session, or number continuously across the whole campaign.

5.3.2 Flashbacks

Past events that inform the current story. Use letter suffixes:

```
S20 *The Rusty Anchor — Sable recognizes
the bartender*
=> "I know you. You were at the Baron's
dinner."

S20a *Flashback: Baron Holt's dinner
```

```
party, three months ago*
@(Sable) Overhear a conversation in the
hallway
d: Perception d20+1=17 vs DC 15 ->
Success
=> Holt was talking about "the shipment"
— and the bartender was there.

S21 *The Rusty Anchor — present*
=> Sable locks eyes with the bartender.
He knows she knows.
```

5.3.3 Split Party

When the party divides, prefix each scene with its storyline number. A storyline is a line of play, not a story thread (`[Thread:]`):

```
T1-S22 *Kael and Mira at the healer's*
T2-S22 *Sable tailing the bartender*
```

When storylines rejoin, continue with plain scene numbers and say so in a note:

```
T1-S24 *Kael and Mira return to the safe
house*
T2-S24 *Sable arrives moments later*

S25 *Safe house — party reunited* (note:
storylines T1 and T2 merge)
```

5.3.4 Montages and Time Cuts

For activities spanning time or quick vignettes, use decimal notation:

```
S15 *Downtime in Northport — one week*

S15.1 *Kael: Casing the Baron's estate*
@(Kael) Gather information on guard
rotations
d: Investigation d20+4=18 vs DC 14 ->
Success
=> Maps the patrol schedule.

S15.2 *Sable: Researching the Sunstone*
@(Sable) Library research
d: Arcana d20+6=21 vs DC 16 -> Success
=> Finds a reference to the Sunstone in a
pre-war text.

S15.3 *Mira: Training at the fighting
pit*
@(Mira) Spar with a local champion
d: Athletics d20+4=15 vs DC 15 -> Success
=> Earns respect — and a contact.
[N:Voss|pit fighter|friendly]

S16 *Safe house — planning the heist*
(Montage complete)
```

5.3.5 Choosing Your Approach

- **Sequential** (S1, S2, S3): Default. Linear play, simplicity.
- **Flashbacks** (S5a, S5b): Backstory, revelations, character moments.

- **Split party** (T1-S1, T2-S1): Simultaneous events, multiple locations.
- **Montages** (S7.1, S7.2): Downtime, travel, training, shopping.

5.3.6 Scene Context

Enrich scenes with context:

```
S18 *Sewer tunnels, midnight*
S20 *The Rusty Anchor (tense)*
S15.2 *Sable at the library, Day 3*
S20a *Flashback: Baron's dinner, three
months ago*
T2-S22 *Meanwhile: Sable tailing the
bartender, same night*
```

5.4 Session End

Many systems have formal end-of-session rituals — XP awards, advancement triggers, debrief questions. An optional closing block captures these alongside the next-session hook.

Digital:

````
### End of Session 7

```
[Advance:Kael|Rogue 6|+Expertise]
[Party:Crew|XP+1 each]
(hook: the shipment arrives in 3 days —
scout the temple before it arrives)
````
````

```
(note: strong session, the escape was
tense)
```
```

**Analog:**

```
--- End of Session 7 ---
[Advance] Kael: Rogue 6 (+Expertise)
[Party:Crew|XP+1 each]
(hook: shipment in 3 days — scout the
temple)
(note: strong session)
```

Common fields:

- `[Advance: ...]` — Characters who leveled up or advanced (see §4.1.2)
- XP and resource changes via existing `[PC:]` and `[Party:]` tags
- `(hook: ...)` — The situation or question driving the next session
- `(note: ...)` — Debrief, table reflection, what worked

5.5 Interlude

Events that happen *between* sessions — off-camera travel, recovery, world changes, NPC developments — sit outside session scope. Use an Interlude block between session headers.

Digital:

```
## Interlude: One week — coast road
```
```

```
[PC:Mira|HP 34/34|-bandaged]
(fully healed)
[Party:Crew|Rations-7]
(seven days' travel)
[Clock:Holt's Search +2]
(time passes)
[Faction:City Watch|
standing:neutral>suspicious]
(note: off-camera — narrated summary, no
active play)
```

Analog:

```
=== Interlude: One week, coast road ===
[PC:Mira|-bandaged|HP full]
[Party:Crew|Rations-7]
[Clock:Holt's Search +2]
[Faction:City Watch|neutral>suspicious]
```

Interludes differ from **Montages** (§5.3.4): montages happen *during* a session as active play; interludes record what happened *off-camera* between sessions.

6. Complete Examples

6.1 Minimal Shorthand Log

Pure shorthand — for fast-paced sessions when the scribe is barely keeping up:

```
S18 @(K)Sneak d:8<14 ->F => noise
[Clock:Alert 2/6]
!Guards coming @(S)Sleep d:save 9<13 ->F
=> guard drops
S19 @(M)Barricade d:16≥12 ->S => safe for
now [L:Safe House]
```

Single-letter abbreviations for character names work fine when everyone knows who's who.

6.2 Hybrid Digital Format

Combines shorthand with narrative:

````
### S18 *Sewer tunnels beneath the
estate, midnight*

```
@(Kael) Lead the group through the
tunnels
d: Survival d20+2=14 vs DC 12 -> Success
````
````

```
=> Finds the right passage. Distant sound
of water.

! A grate ahead is rusted shut
@(Mira) Force it open
d: Athletics d20+4=19 vs DC 15 -> Success
=> Metal screams. They're through.
[L:Sewer Exit|docks|night]
```

```
Cold air hits them as they emerge by the
waterfront. Mira's wound
is bleeding again — she leans on Kael,
trying not to show it.

```
PC(Kael): "We need to find a healer.
Tonight."
PC(Sable): "I know someone in the Docks.
Owes me a favor."
PC(Mira): "I'm fine."
PC(Kael): "You're not fine."
```
```

6.3 Analog Notebook Format

Same content, formatted for handwriting:

```
S18 *Sewer tunnels, midnight*
```

```
@(Kael) Lead through tunnels
d: Survival d20+2=14 vs DC 12 -> Success
=> Finds the right passage.

! Grate ahead, rusted shut
@(Mira) Force it open
d: Athletics d20+4=19 vs DC 15 -> Success
=> Through. [L:Sewer Exit|docks|night]

Cold air. Mira's wound bleeding again.

PC(Kael): "We need a healer. Tonight."
PC(Sable): "I know someone. Owes me a
favor."
```

6.4 Complete Session Log (Digital)

```
---
title: The Sunstone Conspiracy
ruleset: D&D 5e
gm: Roberto
players: Alex (Kael), Jordan (Sable), Sam
(Mira)
---

# The Sunstone Conspiracy

## Session 7
```

```
*Date: 2025-11-15 | Duration: 3h30 |
Scribe: Jordan*
*Players: Alex (Kael), Jordan (Sable),
Sam (Mira)*

**Recap:** Infiltrated Baron Holt's
estate. Discovered in the study.
Escaped through the sewers. Mira took a
crossbow bolt.

**Goals:** Regroup, heal Mira, follow up
on the Sunstone lead.

### S18 *Sewer tunnels beneath the
estate*

```
@(Kael) Navigate the tunnels toward the
docks
d: Survival d20+2=14 vs DC 12 -> Success
=> Finds the old drainage route. Moving
quickly.

! The tunnel forks — one passage smells
of salt, the other of rot
@(Sable) Use Prestidigitation to test the
air
=> Salt passage leads to the docks. They
take it.

@(Mira) Keep moving despite the wound
```
```

````
d: CON save d20+3=11 vs DC 10 -> Success
=> Gritting teeth, she keeps pace.
[PC:Mira|HP 12/34|wounded]
```

### S19 *Docks District — Sable's contact*

```
@(Sable) Find the healer, Tomas
d: Investigation d20+1=15 vs DC 12 ->
Success
=> A cramped room above a fishmonger's.
Tomas is in.
[N:Tomas|healer|underground|owes Sable]

N(Tomas): "You look terrible. All of
you."
PC(Sable): "Mira needs patching. Crossbow
bolt, maybe three hours ago."
N(Tomas): "That'll cost."
PC(Kael): "Name your price."
```

```
@(Kael) Negotiate the fee
d: Persuasion d20+3=9 vs DC 14 -> Fail
=> Tomas won't budge. Full price.
[Party:Crew|Gold-25]

! Tomas treats the wound
````

```
=> [PC:Mira|HP+15|HP 27/34|-wounded|
bandaged]
```

S20 *Safe house in the Docks District, dawn*

```
@(Kael) Set up watches and secure the
room
=> [L:Safe House|hidden|cramped|secure
for now]
```

The party takes stock. They got what they came for — the documents
from Holt's study — but the Baron knows someone broke in.

```
@(Sable) Examine the stolen documents
d: Investigation d20+1=18 vs DC 14 ->
Success
=> The documents reference a "Sunstone
shipment" arriving by sea
   in four days. Destination: an
abandoned temple north of the city.
[Thread:Sunstone Shipment|Open]
[Timer:Shipment Arrives 4]
```

```
PC(Sable): "Four days. A shipment coming
by sea to a temple."
PC(Mira): "What's a Sunstone?"
PC(Sable): "I've been researching that.
Pre-war texts mention it as
            a source of immense power.
Holt wants it badly."
PC(Kael): "Then we get there first."
```

```
[Thread:Baron Holt's Retaliation|Open]
[Clock:Holt's Search 1/6]
```

(note: great session — the escape was
tense, and the Sunstone reveal
gives us a clear objective. next session:
prep for the temple)

6.5 Complete Session Log (Analog)

```
=== Session 7 ===
[Date]         2025-11-15
[Duration]     3h30
[Scribe]       Jordan
[Players]      Alex (Kael), Jordan
(Sable), Sam (Mira)
```

```
[Recap]        Infiltrated Baron's
estate. Discovered. Escaped via
               sewers. Mira hit by
crossbow bolt.
[Goals]        Regroup, heal Mira, follow
Sunstone lead.

S18 *Sewer tunnels beneath the estate*

@(Kael) Navigate toward the docks
d: Survival d20+2=14 vs DC 12 -> Success
=> Finds the drainage route.

! Tunnel forks — salt smell vs rot
@(Sable) Prestidigitation to test the air
=> Salt = docks. They go left.

@(Mira) Keep pace despite wound
d: CON save d20+3=11 vs DC 10 -> Success
=> Keeps moving. [PC:Mira|HP 12/34|
wounded]

S19 *Docks — Sable's contact*

@(Sable) Find Tomas the healer
d: Investigation d20+1=15 vs DC 12 ->
Success
=> Cramped room above fishmonger's.
[N:Tomas|healer|underground]

N(Tomas): "You look terrible."
```

```
PC(Sable): "Mira needs patching. Crossbow
bolt."
PC(Kael): "Name your price."

@(Kael) Negotiate
d: Persuasion d20+3=9 vs DC 14 -> Fail
=> Full price. [Party:Crew|Gold-25]

! Tomas treats the wound
=> [PC:Mira|HP+15|-wounded|bandaged]

S20 *Safe house, dawn*

@(Kael) Secure the room
=> [L:Safe House|hidden|secure for now]

@(Sable) Examine stolen documents
d: Investigation d20+1=18 vs DC 14 ->
Success
=> Sunstone shipment in 4 days.
Destination: abandoned temple.
[Thread:Sunstone Shipment|Open]
[Timer:Shipment Arrives 4]

PC(Kael): "Then we get there first."

[Thread:Baron Holt's Retaliation|Open]
[Clock:Holt's Search 1/6]

(note: great session. temple prep next
week.)
```

6.6 Combat Example (Detailed)

A full combat encounter showing initiative, multiple participants, and world events:

````
### S22 *Ambush on the coast road*

```
! Bandits spring from the tree line —
four of them, armed
[F:Bandits|×4|HP 11 each|pos:Close]

[COMBAT]
(init: Mira 19, Bandit Leader 17, Kael
14, Bandits 12, Sable 8)

Rd1
@(Mira) Draw greatsword, charge the
leader
d: d20+6=21 vs AC 13 -> Hit, 2d6+4=15 dmg
=> Devastating opening strike: 15 of his
27 HP. [F:Bandit Leader|HP-15]

@(Bandit Leader) "Kill them! Take the
satchel!"
d: d20+4=17 vs AC 18 (Mira) -> Miss
=> Mira deflects with her bracer.

@(Kael) Shortbow at the nearest bandit
d: d20+7=19 vs AC 12 -> Hit, 1d6+3d6+3=16
dmg
=> Drops instantly. [F:Bandits|×3]
````
````

```
@(Bandits) Two of them rush Sable
d: d20+3=15 vs AC 12 -> Hit, 1d6+1=5 dmg
d: d20+3=11 vs AC 12 -> Miss
=> One blade gets through. [PC:Sable|
HP-5]

@(Sable) 5-foot step back, cast Burning
Hands
d: DEX saves — Bandit 1: d20+1=8, Bandit
2: d20+1=14 vs DC 14
=> 3d6=11 fire dmg. Bandit 1 takes full
(dead), Bandit 2 half.
   [F:Bandits|×2|one wounded]

Rd2
@(Mira) Finish the leader
d: d20+6=18 vs AC 13 -> Hit, 2d6+4=11 dmg
=> [F:Bandit Leader|HP 0|dead]

! Last bandit drops his weapon
N(Bandit): "I yield! Please!"

@(Kael) "Who sent you?"
(note: Sam wants to intimidate, Alex
wants to just tie him up)
@(Kael) Tie him up while Mira looms
d: Intimidation (Mira, assist) d20+3=17
vs DC 12 -> Success
=> He talks. The Baron hired them to
intercept the party.
```

```
[Thread:Baron Holt's Retaliation|
confirmed — he knows]
[Clock:Holt's Search 3/6]
[/COMBAT]

=> Party takes stock. Sable bandages
herself.
[PC:Sable|HP 13/18]
[N:Captured Bandit|terrified|cooperative]
```

7. Best Practices

7.1 Good Practices ✔️

Do: Attribute actions in multi-character scenes

```
@(Kael) Pick the lock
@(Sable) Watch the corridor
```

Do: Separate player actions from world events

```
@(Mira) Charge across the bridge
! The bridge starts to collapse
```

Do: Keep actions and rolls connected

```
@(Kael) Pick the lock
d: d20+7=19 vs DC 15 -> Success
=> The door swings open.
```

Do: Use tags for persistent elements

```
[N:Baron Holt|hostile|powerful]
[L:Safe House|hidden|docks]
```

Do: Record who was absent

```
[Absent] Sam (Mira stays at camp)
```

Do: Prioritize mechanics over prose when falling behind

```
Fast:  @(K) lock d:19≥15 ->S => open
Slow:  Kael knelt before the ancient
lock, his picks glinting...
```

The fast version captures the game. The slow version can be written later.

7.2 Bad Practices ✗

Don't: Bury mechanics in prose

```
✗ Kael tried to pick the lock and rolled
a 19 which beat the DC

✓ @(Kael) Pick the lock
  d: d20+7=19 vs DC 15 -> Success
```

Don't: Forget attribution in group scenes

```
✗ @ Attack the orc    (who?)

✓ @(Mira) Attack the orc
```

Don't: Use ! for consequences of player actions

```
✗ @(Kael) Kick the door open
  ! The door smashes inward

✓ @(Kael) Kick the door open
  => The door smashes inward
```

Don't: Try to capture every word of dialogue

```
✗ (Three pages of exact conversation)

✓ PC(Kael): "I don't trust the Baron."
  (note: long argument about whether to
go to the dinner)
  => Party decides to attend, but armed.
```

Don't: Lose track of who's where during split party

```
✗ S22 @(Kael) searches the library
  S23 @(Sable) follows the suspect
  (Who is where? When?)

✓ T1-S22 *Kael at the library*
  T2-S22 *Sable tailing the suspect*
```

8. Templates

8.1 Campaign Template (Digital)

```
title:
ruleset:
genre:
gm:
players:
pcs:
start_date:
last_update:
themes:
tone:
notes:
```

```
# [Campaign Title]

## Session 1
*Date: | Duration: | Scribe: *
*Players: *

### S1 *Starting scene*

Your play log here...
```

8.2 Campaign Template (Analog)

```
=== Campaign Log: [Title] ===
[Title]
[Ruleset]
[Genre]
[GM]
[Players]
[PCs]
[Start Date]
[Last Update]
[Themes]
[Tone]
[Notes]

=== Session 1 ===
[Date]
[Duration]
[Scribe]
[Players]

S1 *Starting scene*

Your play log here...
```

8.3 Session Template

Digital:

```
## Session X
*Date: | Duration: | Scenes: *
*Players: *
*Scribe: *

**Recap:**

**Goals:**

### S1 *Scene description*

...

### End of Session X

```
[Advance:Name|Class Level|+gained]
[Party:Crew|XP+]
(hook:)
(note:)
```
```

Analog:

```
=== Session X ===
[Date]
[Duration]
[Players]
[Scribe]
[Absent]
```

```
[Recap]
[Goals]

S1 *Scene description*

--- End of Session X ---
[Advance] Name: Class Level (+gained)
[Party:Crew|XP+]
(hook: )
(note: )
```

8.4 Quick Scene Template

```
S# *Location, time*
@(Name) Action
d: roll -> outcome
=> What happens

! World event
=> Consequence
```

8.5 Combat Template

```
[COMBAT]
(init: Name X, Name X, Name X)
```

```
Rd1
@(Name) Action
d: roll -> outcome
=> [F:Enemy|effect]

@(Enemy) Action
d: roll -> outcome
=> [PC:Name|effect]

Rd2
...
[/COMBAT]
```

9. Adapting to Your System

The core symbols work with any TTRPG. Only the resolution details change based on your system.

9.1 System-Specific Roll Notation

Powered by the Apocalypse (PbtA)

```
@(Kael) Read a Tense Situation
d: 2d6+Sharp=9 -> Strong Hit (10+)
d: 2d6+Cool=7 -> Weak Hit (7-9)
d: 2d6+Hot=4 -> Miss (6-)
```

Forged in the Dark (FitD)

```
@(Kael) Prowl past the Bluecoats
d: 3d6=5,4,2 (take highest=5) -> Partial
Success (4-5)
d: 4d6=6,6,4,1 (two 6s) -> Critical
```

D&D / d20 Systems

```
@(Mira) Attack the ogre
d: d20+6=21 vs AC 16 -> Hit, 1d12+4=13
```

```
dmg

! Ogre attacks Kael
d: d20+6=18 vs AC 15 (Kael) -> Hit,
2d8+4=14 dmg
```

Year Zero Engine (Forbidden Lands, Alien, etc.)

```
@(Kael) Sneak past the guards
d: 4 Base + 2 Skill = 3,5,6,1,4,6 -> 2
successes (6s)
=> [PC:Kael|Gear -1 (rolled 1)]
```

Fate / Fudge

```
@(Sable) Create Advantage: Distracting
Illusion
d: 4dF=+2 (++0-) +Deceive(3) = +5 vs
passive +2 -> Success with Style
=> Free invoke on "Mesmerized Guards"
aspect
```

Call of Cthulhu / BRP

```

```

```
@(Kael) Spot Hidden
d: d100=34 vs Skill 55 -> Regular Success
d: d100=08 vs Skill 55 -> Extreme Success
d: d100=72 vs Skill 55 -> Fail
```

9.2 Handling Edge Cases

Advantage / Disadvantage

```
@(Kael) Attack with advantage (flanking)
d: 2d20=17,8 (take higher=17) vs AC 14 ->
Hit
```

Group Checks

```
! The party tries to sneak through the
camp
d(Kael): Stealth d20+7=21 -> S
d(Sable): Stealth d20+0=8 -> F
d(Mira): Stealth d20-1=6 -> F
=> Group check: only 1 of 3 succeeds, so
the party fails. They're spotted.
```

Reactions and Interrupts

```

```

```
@(Bandit) Fires crossbow at Sable
@(Mira) [Reaction] Shield — step in front
d: — (no roll, class feature) -> Sable
takes no damage
=> Bolt glances off Mira's shield.
[PC:Mira|reaction used]
```

Concentration and Sustained Effects

```
@(Sable) Maintain concentration on Hold
Person
d: CON save d20+1=14 vs DC 10 (damage
taken) -> Success
=> Spell holds. The bandit is still
frozen.
```

Appendices

A. Symbol Reference

A.1 Core Symbols

Symbol	Meaning	Example
`@` / `@(Name)`	Player character action	`@(Kael) Pick the lock`
`@(A > B)`	Collaborative action (Lead > Assist)	`@(Mira > Kael) Assist`
`!`	GM / world event	`! The ceiling collapses`
`d:`	Roll or rule that gives a verdict	`d: d20+5=18 vs DC 15 -> Hit`
`tbl:`	Random result that gives content	`tbl: Loot d100=42 -> A broken sword`
`->`	Resolution outcome	`-> Success`
`=>`	Consequence	`=> The door opens quietly`

A.2 Comparison Operators

- `≥` or `>=` — Meets/beats target
- `<` — Falls short of target
- `vs` — Neutral comparison

- `-> S` / `-> F` — Success / Fail
- `+[…]` / `-[…]` before the dice — Roll context

A.3 Tracking Tags

- `[N:Name|tags]` — NPC (first mention)
- `[N:Name|cat:tag,tag]` — NPC with categorized tags
- `[#N:Name]` — NPC (reference)
- `[L:Name|tags]` — Location
- `[E:Name]` — Notable event
- `[Thread:Name|state]` — Story thread
- `[Goal:Name|state]` — Shared objective
- `[Quest:Name|state]` — Major milestone
- `[Loot:Name|tags]` — Unassigned party item
- `[PC:Name|stats]` — Player character
- `[Party:Name|resources]` — Party-level tracking, named after the group
- `[Faction:Name|tier:X|standing:Y]` — Faction (power and relationship)
- `[Advance:Name|Class Level|+gained]` — Character advancement event
- `[F:Enemy|stats]` — Foe in combat
- `[N:Name | tag | tag ]` — Multi-line tag structure

A.4 Progress Tracking

- `[Clock:Name X/Y]` — Clock (fills up, usually a threat)
- `[Track:Name X/Y]` — Progress track (fills up, usually a goal)
- `[Timer:Name X]` — Countdown (reaches zero, time's up)

A.5 Random Generation

- `tbl: Name roll -> result` — Table or generator result
- `tbl: Name [A, B, C]` — Filtered option set
- `tbl: Name (d#) \n 1: Res` — Inline table definition

- `tbl: Name \n Axis: res` — Multi-line result block

A.6 Structure

- `S#` — Scene number
- `S#a` — Flashback scene
- `T#-S#` — Scene in a parallel storyline (split party)
- `S#.#` — Montage / time cut

A.7 Narrative

- Inline: `=> Prose here`
- PC dialogue: `PC(Name): "Speech"`
- NPC dialogue: `N(Name): "Speech"`
- Block: `\--- text ---\`

A.8 Meta

- `(note: ...)` — Out-of-character commentary
- `(rule: ...)` — GM Ruling / House Rule
- `(post: ...)` — Post-session reconstruction marker
- `(hook: ...)` — Next session hook (used in Session End)
- `[OOC: ...]` — Table-level Status (Safety, Breaks, AFK)

A.9 Combat

- `[COMBAT]` / `[/COMBAT]` — Combat block markers
- `Rd1`, `Rd2` — Round markers
- `@(NPC)` — Enemy turn within combat
- `[F:Enemy|stats]` — Foe tracking

B. Oracle Notation for GM-less Play

Some group RPGs operate without a traditional GM. Games like Ironsworn in co-op mode, Fiasco, The Quiet Year, or any game using

an oracle (such as Mythic GME) with multiple players share narrative authority instead of assigning it to one person.

In these contexts, the `!` symbol is not used — there is no GM to push events. Instead, the world speaks through oracles, and any player can ask.

B.1 Oracle Questions

Use `?` when a player asks the game world a question:

```
? Is anyone guarding the bridge?
-> Yes, but... (d6=4)
=> Guards are present, but distracted by
an argument.
```

The `->` symbol resumes its dual role from solo play: it resolves both mechanics (dice outcomes) and oracle answers (world questions).

B.2 Oracle Formats

- **Yes/No:** `-> Yes`, `-> No`
- **With modifiers:** `-> Yes, but...`, `-> No, and...`
- **Degree results:** `-> Strong yes`, `-> Weak no`
- **Custom:** `-> Partially`, `-> With a cost`

B.3 Combining Oracle and Attributed Actions

```
@(Kael) Sneak past the camp
d: Stealth 2d6+Shadow=8 -> Weak Hit
(Ironsworn)
=> I pass through, but something notices
```

```
me.

?(Sable asks) What noticed him?
tbl: Oracle d100=67 -> "A creature,
hostile"
=> A wolf, half-starved, blocking the
path ahead.
```

When multiple players share oracle duties, optionally attribute who asked:

```
?(Sable) Is the ruin inhabited?
?(Kael) What do we find inside?
```

B.4 When to Use This Appendix

Use oracle notation when:

- Playing a GM-less RPG (Ironsworn co-op, Fiasco, The Quiet Year, etc.)
- Using a GM emulator (Mythic GME, CRGE, MUNE) with a group
- Any session where no single player has narrative authority

Do **not** mix `!` and `?` in the same session. If you have a GM, use `!`. If you don't, use `?`. Mixing them creates ambiguity about who has narrative authority.

C. Symbol Design Philosophy

Partylog's symbols were chosen for specific reasons:

- `@` **(Action)**: Represents the actor. In group play, `@(Name)` attributes the action. Visually distinct from markdown operators.

- `!` **(World Event)**: The exclamation mark carries urgency — something is happening *to* the party. It mirrors GM energy: declarative, external, not requested. This replaces the solo `?` (oracle question) as a core symbol because in group play, the world doesn't wait to be asked. It acts.
- `d:` **(Dice/Resolution)**: Clear abbreviation for dice rolls and mechanics. Optionally attributed with `d(Name):`.
- `->` **(Resolution)**: Arrow showing "this leads to outcome." In standard group play, it resolves dice mechanics. In GM-less play (Appendix B), it also resolves oracle answers.
- `=>` **(Consequence)**: Double arrow showing cascading effects. Always represents what happens *as a result* of actions, events, and rolls.

Markdown Compatibility: All symbols work cleanly in code blocks and don't conflict with markdown formatting. Always wrap notation in code blocks when using digital markdown.

The `!` / `?` distinction: This is the fundamental design choice that separates Partylog from solo notation. In solo play, the world only speaks when asked (`?`). In group play, the world speaks when the GM decides (`!`). These represent opposite models of narrative authority — pull vs. push — and using distinct symbols makes the model explicit.

Credits & License

© 2025-2026 Roberto Bisceglie

Partylog is forked from **Lonelog** v1.0.0 by Roberto Bisceglie and kept aligned with Lonelog v1.7.0. It shares Lonelog's core philosophy, tag system, and structural conventions, adapted for group TTRPG play.

Version History:

- v 1.1.0: Round markers are `Rd#`, matching the Combat Add-on; initiative is an `(init: …)` note; HP changes are written as changes, not arithmetic. Aligned with Lonelog v1.7: `[Party:]` is named after the group (`[Party:Crew|Gold 150]`); group sizes are `×N` values, never part of the name; truthful comparisons with outcomes after `->`, roll context before the dice, accumulating tags with `old>new`, `[E:]` for events and `[Clock:]` for counters, `gen:` folded into `tbl:`, parallel storylines.
- v 1.0.0: Public launch
- v 0.5.0: Added Safety Tool Protocol (§4.5), Logical Tag Flow recommendation (§4.1), and clarified the Scribe's role as a diegetic observer (§1.5).
- v 0.4.0: Added Faction tracking (`[Faction:]`), PC advancement events (`[Advance:]`), Session End structure (§5.4), Interlude blocks (§5.5), and `(post:)` / `(hook:)` meta notes. Fixed section 4.1 numbering.
- v 0.3.0: Added "Group Dynamic" extensions: Collaborative Actions (`>`), Loot tracking, Goals/Quests, and formalized Rulings (`rule:`) / OOC status.
- v 0.2.0: Added Roll Context, Tag Categories, Multi-Line Tags, and expanded Random Table support (imported from Lonelog v1.3.0)
- v 0.1.0: Initial draft, forked from Lonelog v1.0.0

This work is licensed under the **Creative Commons Attribution-ShareAlike 4.0 International License**.

You are free to:

- Share — copy and redistribute the material
- Adapt — remix, transform, and build upon the material

Under these terms:

- Attribution — Give appropriate credit
- ShareAlike — Distribute adaptations under the same license