Things to consider when it comes to creating material that will help others learn how to plan Samman style Learning Hours.

Planning **Samman-style Learning Hours** (as per the work of Emily Bache and the Samman Technical Coaching method) requires a thoughtful balance between structure, engagement, and relevance. This style is highly focused on collaborative, practical learning, especially for developers or technical teams. Here's what to consider when creating **learning material** for this format:

CORE PRINCIPLES TO KEEP IN MIND

1. **Short, Focused, Repeatable Sessions**
    - Typically 60–90 minutes long
    - Designed for repeated weekly/bi-weekly cadence
    - Not a one-off workshop — aim to build a habit of learning
2. **Learning by Doing**
    - Emphasis on **hands-on coding**, not just listening
    - Use **katas, exercises, or mob programming** to engage all participants
3. **Skill-Building with a Purpose**
    - Target **technical practices** like refactoring, TDD, pair/mob programming, design, etc.
    - Tailor content to real needs of the team (e.g., improving legacy code, testability, etc.)

PLANNING YOUR MATERIAL

1. Define Clear Learning Objectives

- What should participants be better at after this session?
- Examples:
    - "Understand the value of small commits"
    - "Practice the Red-Green-Refactor loop"
    - "Use mocking frameworks appropriately"

---

2. Choose Appropriate Exercises

- Use code **katas** (e.g., FizzBuzz, Gilded Rose, Mars Rover)
- Legacy code exercises with tests missing are especially valuable
- Adjust difficulty to match experience level of the team
- Prefer exercises with:
    - Clear boundaries
    - Opportunity for test-first and refactoring
    - Domain that doesn't distract from the core learning

---

3. Prepare Starter Material

- Pre-cloneable repos with exercise skeletons
- Tests prepared (possibly failing)
- README or guide outlining the task
- Ensure all tooling is easy to run (no painful setup)

---

4. Support the Mob Programming Structure

- Use **strong-style pairing**: the person at the keyboard follows instructions from the navigator
- Rotate roles every 4–5 minutes
- Assign a **facilitator** to keep time and guide reflection

---

5. Create Prompts and Checkpoints

- Prepare questions to spark discussion:
    - “What’s a better name for this function?”
    - “Could we remove duplication here?”
    - “Would a test help us clarify this behavior?”
- Use “pause and reflect” moments mid-session

---

6. Include Time for Retrospective

- Allocate ~10 minutes at the end
- Ask:
    - What did we learn?
    - What was confusing?
    - What would we try differently next time?

CONTENT CREATION TIPS

- **Keep instructions minimal** — avoid over-documenting
- Provide just enough context to get started
- Use **progressive disclosure**: reveal complexity step-by-step
- Have variations of the exercise to adjust in real-time (e.g., add constraints like "no conditionals")

MINDSET FOR FACILITATORS

- Your role is **to guide, not lecture**
- Be prepared to step back and let the group solve things
- Focus on **psychological safety** — learning works best when people feel safe to fail
- Be curious: adapt based on how the team responds

CHECKLIST FOR A SESSION

Task	Notes
Defined learning goal?	Clear, specific, actionable
Exercise selected?	Matches goals and skill level
Repo/materials ready?	Easy to access and run
Roles clarified?	Facilitator, driver, navigator
Reflection time planned?	At end, or during checkpoints
Optional: Backup exercise?	In case of faster group pace