Tech Lead Basics

Version 2026 · verified 2026-11-02

Your output is now the team's output — 1:1s, delegation levels, mentoring vs sponsoring, unblocking, and the hero, bottleneck and absentee traps.

Contents

Mental model 2

What a tech lead owns 2

Running a 1:1 3

Delegation levels 4

Mentoring vs sponsoring 5

Unblocking and prioritising 5

Protecting focus time 6

Anti-patterns and growing the next lead 7

Further reading 8

Mental model

Your output is now the team's output. An hour spent unblocking two engineers, reviewing a design, or making a decision nobody else could make is the job — not a distraction from "real work". Fournier's core lesson for the role is the willingness to step away from the code; leads who measure themselves by their own commits end up as the team's bottleneck.

What a tech lead owns

Owns	Does not own
Technical direction and decisions	Hiring, firing, pay
Project delivery and risk	Performance reviews (input only)
Architecture of the team's area	Every line of code
Unblocking and pace	Product priorities (shapes them)

The role varies by company, so write it down. Larson's *Tech Lead* archetype scopes complex work, coordinates the team towards it, and unblocks along the way; Fournier's chapter adds understanding the architecture, leading technical decisions and communicating. Agree the split with your engineering manager (EM) and product manager (PM) in week one — gaps between three leads are where projects quietly die.

```
Role split (agreed 2026-11-02)
TL : design reviews, tech decisions, risks
EM : people, hiring, career, process
```

```
PM : what and why, priorities, stakeholders
Shared: roadmap, incident follow-ups
```

> **Gotcha:** “Tech lead” without direct reports still gets asked about people problems. Pass performance and pay questions to the EM, and tell the person you did.

Running a 1:1

Item	Practice
Owner of the agenda	Them; yours goes last
Cadence	Weekly or biweekly, 30–60 min
Format	Shared running doc
Status updates	Not here — use standup or the board
End with	Action items, reviewed next time

A 1:1 is their time to raise what does not fit in a group: blockers, frustrations, growth. Google’s re:Work 1:1 guide recommends a shared agenda both people add to, a regular slot, and ending with next steps. If a 1:1 turns into a status report, the person has nothing else they trust you with yet.

```
## 2026-11-02
- [them] Stuck on auth migration scope
- [them] Want to lead next design review
- [me] Feedback: last PR description
```

```
- Next: I intro you to the auth owner
- Next: you draft the review agenda
- "Anything I should do that I'm not?"
```

Delegation levels

Level	Who decides
Tell / Sell	You decide (and explain)
Consult	You decide after asking them
Agree	You decide together
Advise	They decide; you give input
Inquire / Delegate	They decide; tell you after, or not at all

Delegating is not binary. Management 3.0's seven levels — Tell, Sell, Consult, Agree, Advise, Inquire, Delegate — let you hand over a decision area at the level the person is ready for, then move it up. Say the level out loud; most delegation failures are a mismatch between the level you meant and the one they heard.

```
"You own the caching design. I'd like to
see it before it's built (Advise) — after
that, the implementation calls are yours
and you don't need to check with me
(Delegate)."
```

> **Warning:** Taking work back the first time it goes wrong teaches the team that delegation is temporary. Coach the fix instead.

Mentoring vs sponsoring

Mode	You give	Best for
Mentoring	Advice from your experience	New to the role
Coaching	Questions; they find the answer	Building a skill
Sponsoring	Your influence and visibility	Reaching the next level

Larson notes the most effective staff engineers pair a moderate amount of mentorship with considerably more sponsorship. Lara Hogan defines sponsoring as using your influence to advance someone: putting their name forward for a project, a talk or a promotion case. Most of that work is invisible to the person you sponsor, and you do not need to be a manager to do it.

```
In the planning meeting:
"Priya debugged the queue backlog last
month — she should lead the redesign.
I'll review her design, but it's hers."
```

Unblocking and prioritising

Signal	Action
Ticket idle for days	Ask what it’s waiting on
Same question asked twice	Write it down once

Signal	Action
Decision stuck in a thread	Decide, or name who decides by when
Too many in-flight items	Stop starting; finish

A team rarely fails for lack of effort; it fails waiting — for a decision, a review, another team. Your highest-leverage habit is scanning for waiting work each day. LeadDev's first-time tech lead advice sorts decisions by stakes: decide small ones yourself, take input on medium ones, and write a proposal for big ones.

```
Daily scan (10 min)
1. PRs waiting > 1 day for review?
2. Tickets blocked on another team?
3. Open questions with no owner?
4. Anything I promised and haven't done?
```

Protecting focus time

Practice	Protects
Meeting-free blocks on the calendar	Your design and review time
Office hours for questions	Everyone else's focus
Rotating interrupt duty	One person absorbs pings
Off the critical path	The deadline, from your meetings

Leads collect meetings, so code you own becomes the slowest code in the sprint. Keep yourself off the critical path: pick work that matters but

can wait a day, such as tooling, spikes or reviews. Protect the team's focus the same way — batch questions, keep one person on interrupts.

```
Mon-Fri 09:00-11:00  Focus (decline)
Tue, Thu 15:00-15:30 Office hours
This sprint: interrupts -> Sam
My tickets: tooling only, no deadlines
```

Anti-patterns and growing the next lead

Anti-pattern	Symptom
Hero	You fix every incident; nobody else can
Bottleneck	Every decision and review waits on you
Absentee	In meetings all day; team drifts

Each anti-pattern is the old job leaking into the new one, or the new job crowding out the team. The test is the same for all of them: what happens if you take two weeks off? Growing the next lead — handing out design reviews, incident command, and decisions at rising delegation levels — is how you pass that test and how you become free to take on a bigger scope.

```
Bus-factor check (quarterly)
- Who else can deploy?          ____
- Who else can run an incident? ____
```

```
- Who else can review the core? ____

Any blank -> delegate it this quarter
```

Further reading

- The Manager's Path — Camille Fournier
- Staff Engineer — Staff archetypes
- Lara Hogan — What does sponsorship look like?
- Management 3.0 — Delegation Poker
- Google re:Work — Hold effective 1:1 meetings