How trees connect inside

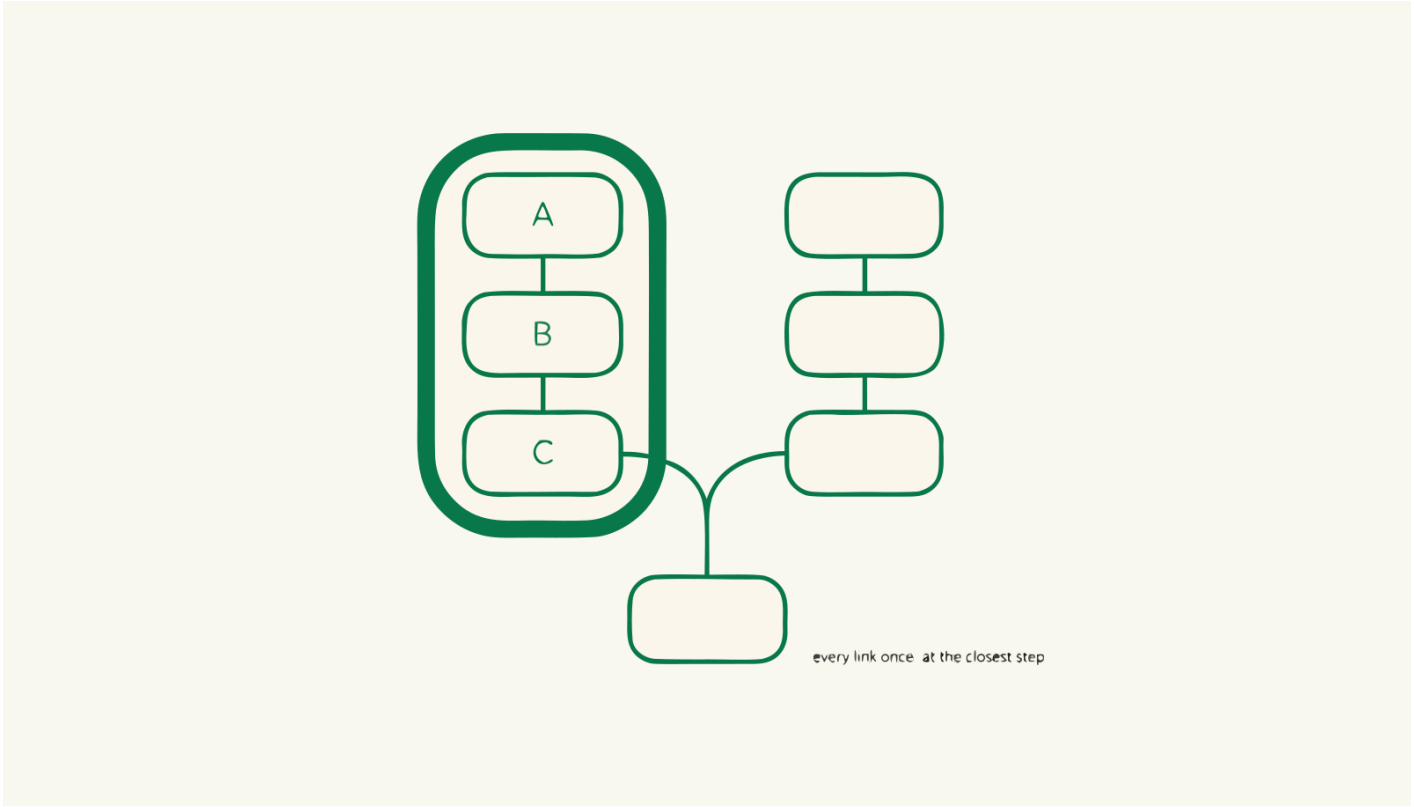
Inside a tree, the prerequisite chains from the skills page become visible paths. Arrows run from each prerequisite up to the skill it unlocks, so the tree isn't just a grouping — it's a route map. Reading one is mostly instinct once you know two conventions:

The path is the prerequisite chain. You start at the bottom of the tree and work up. You can't take a skill before the skills beneath it, the same way you can't take the third rung of a ladder first. If a skill high in the tree looks impossibly hard, the explanation is almost always a missing skill somewhere below it.

Chains stay clean — no double-linking. The rule from the skills chapter does its real work here. If B depends on A and C depends on B, then C connects only through B; there is never an extra arrow from A straight to C. Every dependency is expressed exactly once, at the closest link. This is what keeps even a large tree readable instead of becoming a plate of spaghetti.

Because prerequisites live on the skills themselves (defined once, organisation-wide), a tree never invents connections. It reveals the ones that already exist among the skills it chose to show. Pull "React I" and "React II" into any tree, and they arrive connected — the tree didn't link them, the skills were always linked.

A well-formed tree ends up with a recognisable anatomy: a few foundational skills near the bottom that much of the area rests on, branches where specialisations diverge, and the most advanced skills at the tips. The skills at the very top aren't special — there's no notion of "exit skills," they're just the current ends of the branches. Trees keep growing.



Revision #2
Created 2026-07-08 16:08:49 UTC by Admin
Updated 2026-07-10 09:30:13 UTC by Admin