Playbook Methods Repository

# **Pair Programming**

Two people problem-solving, with one person in control of the keyboard or screen at a time. Ensemble programming is the term used when more than two people are involved. While this technique is typically adopted by Software Engineers, it is not necessarily limited to this role.

### Remote Agility: **•** High

### Linked Tactic(s): Agile Development

## Why we do it:

Modern programming involves more than typing code. Fundamentally, it is about solving problems. Source code is typically what is used to model the solution.

Having said that, our modeled solution must:

* Be executable by the computer (i.e. syntactically correct)
* Meet business requirements (i.e. implement a specification)
* Be understandable by another software engineer
* Easy to change (i.e. maintainable and extensible)

Consequently, having at least two practitioners model the solution and write the code together dramatically improves the chances that all of the above goals are achieved consistently and reliably.

## 

## When to apply it:

Pair programming can apply to all stages of the product development lifecycle, and is used whenever possible.

Pair programming is especially beneficial in the following situations:

* Onboarding to a new:
  + Technology (language, platform, stack, etc.)
  + Business domain
  + Codebase
* Learning from a more experienced practitioner
* Refactoring legacy code / software
* Initial architecting of greenfield code / software
* Researching: implementing a [spike solution](http://www.extremeprogramming.org/rules/spike.html)
* Debugging: determining the root cause of a bug

## Best Practices & Considerations:

Pair programming is a skill. Therefore, it is acquired through repeated application. The [Dreyfus Model of Skill Acquisition](https://en.wikipedia.org/wiki/Dreyfus_model_of_skill_acquisition) underscores that novices need rules (i.e. a set of instructions, or recipe) that they can refer to and confidently follow. Pairing formats are the rulesets we apply in the case of pair programming.

Practitioners at a certain level of proficiency in pair programming may no longer find pairing formats useful. As long as all parties in a pair programming session have reached this level of proficiency, strict adherence to pairing formats can be abandoned. However, when pairing with novices, advanced beginners, or even competent practitioners of pair programming, the use of a pairing format is encouraged.

Furthermore, certain pairing formats are more suitable to novices and advanced beginners than others.

Pairing formats:

* [Ping-Pong Pairing (forms 1 and 2)](https://sites.google.com/connectedlab.com/wiki/guilds-departments/engineering/guild-programs/technical-coaching/pair-programming#h.dlatugwr52k6)
* [Strong-Style Pairing](https://sites.google.com/connectedlab.com/wiki/guilds-departments/engineering/guild-programs/technical-coaching/pair-programming#h.94mi8xk3dg2m)
* [Driver-Navigator](https://sites.google.com/connectedlab.com/wiki/guilds-departments/engineering/guild-programs/technical-coaching/pair-programming#h.x1o6tnfag30c)
* [Structured Ensemble](https://sites.google.com/connectedlab.com/wiki/guilds-departments/engineering/guild-programs/technical-coaching/pair-programming#h.n7v3avc4qdit)

## Responsible roles:

While pair programming most commonly involves two Software Engineers, the product team and customer benefit greatly when *all product team members* participate in Pair Programming or Ensemble Programming sessions. This includes roles and designations such as:

* Software Engineer
* QA Engineer
* Product Designer
* Product Owner
* Product Manager
* Subject Matter Expert
* Customer

## Tools:

### Online tools/platforms/services

* + Visual Studio Code LiveShare
  + Tuple
  + Zoom or Google Meet Screen Share
  + Online coding platforms (e.g. GitPod, Replit)

## 

## Thoughtworks Examples - Linked

### Internal assets - clinic materials / guild docs

* + [Pair Programming](https://sites.google.com/connectedlab.com/wiki/guilds-departments/engineering/guild-programs/technical-coaching/pair-programming) (Thoughtworks wiki article)

## 

## Learn more: How we do this?

### Templates (docs, decks, sheets, miro, etc.)

* + xx

### How-To Resources (external or internal)

* + xx

### Outside References (articles, books, etc.)

* + [Martin Fowler - On pair programming](https://martinfowler.com/articles/on-pair-programming.html)

### Sub-set Activities

* + xx

## 