Ensemble Programming Cheat Sheet

Created by Coaches and Developers @IndustrialLogic

Version 0.2

MODERN AGILE

Why Ensemble Programming?
"All the brilliant people working on the same thing, at the same time, in the same space...." *– Woody Zuill*

"Ensemble Programming is a continuous integration of ideas." *– Joshua Kerievsky*

"Context Debt will accrue interest, pay it early and often." *– Igor Poltosi*

There are many ways to ensemble successfully. In general, there is one computer, a keyboard and mouse, one or more monitors, a whiteboard, one Driver, and one or more Navigators sharing a work environment like the one below:

Image by Mark Pearl

Ideal Ensemble Size The whole team, consisting of 3-5 people is ideal. Beyond 5 people, an ensemble may encounter difficulties keeping everyone engaged. Doing frequent rotations can address this.

Driver's Role - The Driver operates the keyboard to input/implement ideas made by the Navigator(s).

Navigator(s) Role - Navigators direct the Driver. This allows everyone in the ensemble to interact with the Driver. If that causes chaos, have one Navigator give directions to the Driver– the Navigator serves as the voice of the whole ensemble. Newbies to ensemble programming can ask how best to navigate.

Driver Dos & Don'ts
- Drivers don't navigate. If the Driver is the only one who knows what to do, they should navigate. In general, SMEs navigate.
- Each Driver can share how they prefer to be directed, including asking questions about intent, location, and details. Ultimately, navigators must communicate in a way that allows the Driver to understand and take action.
- If no one is navigating, the Driver must stop typing.

Navigator(s) Dos & Don'ts
- Navigator ideas pass through the Driver's ears and hands into the keyboard and code.
- Navigators must pay attention to the Driver's skill level. If the Driver needs word-by-word instructions, the ensemble must explain in detail what the Driver needs to do.
- When the Driver is more advanced, Navigators then give higher-level instructions, like "commit" or "move that method to the parent class". Over-navigating entails too much direction, and under-navigating too little.
- Don't sit and watch the Driver work. *Everyone learns and contributes in an ensemble.*

Ensemble Responsibilities
- Treat everyone with kindness, consideration, and respect
- Plan, discuss, research, and work out ideas on the whiteboard
- If a Driver begins to navigate and drive simultaneously, someone in the Ensemble should call them on it. It's a no-no.
- The Ensemble continues as people join or leave
- Stakeholders, Managers, Subject Matter Experts, etc. are welcome to join and are not required to drive or navigate.

Leaving and Joining - It is fine for people to leave or join the Ensemble. If they are a Driver, they relinquish that role to the next person in line and the ensemble will adjust its rotation schedule.

Switching Drivers - A Driver switchover should only take a few seconds. Switching Drivers is easiest when the Ensemble has the same setup.

If individuals in the Ensemble have preferred tools and settings, consider using tools like mob.sh so folks can have their own IDE settings..

Remote Ensembles - Here are some setup suggestions for successful remote ensemble programming sessions:

- For collaborative keyboard and mouse sharing, there are several tools available: CodeTogether, Code With Me, LiveShare, Tuple, and others.
- We also use mob.sh to share the code repository when our IDE setup is different across the team.
- For screen sharing several tools are available: Zoom, Gather, Microsoft Teams, TeamViewer, and Webex to name a few.
- There are several whiteboard-type tools for story maps, discovery trees, and high-level design discussions. Some that stand out are Miro, Mural, and Figma.
- Perhaps two most important items are a comfortable chair and a great stainless steel container to stay hydrated. Hands down we recommend the YETI brand.
- A high-quality camera and mic are a must.

Timing and Breaks

- Use an automated timer (e.g., Dillon Kearns' Mobster App) to initiate role changes and breaks.
- Try the Ping-Pong collaboration pattern or switch roles every 7 minutes on average (beginners should switch every 2-4 minutes).
- The whole ensemble should take regularly scheduled breaks. The Pomodoro method, a proven method of taking regular breaks to increase efficiency, suggests taking breaks every 48 minutes.

Bias for Action - When discussing how to solve a problem, get out of the abstract as soon as possible.

- Do not argue for more than 5 minutes.
- If there are multiple ideas just pick one and try it, then try another if necessary.
- Keep the Ensemble moving with this quote from Brian Marick: "An example would be handy right about now".

Value of Ensembles - When done well, ensembles help a team:

- Deliver solutions faster by increasing focus, building skills, and sharing knowledge.
- Produce better quality code because the ensemble reviews the code as it is being written.
- Cross-train its members ergo removing knowledge silos and removing context debt.
- Feel the pain of tedious tasks. This is good, as it biases toward fool-proofing and automation.
- Deliver results faster by reducing the team's "work in progress" and eliminating delays from handoffs with the whole team present.

Pitfalls - A poorly functioning ensemble will produce value slowly. The following are some signs of poor ensemble programming and what do to if you observe these behaviors:

- Excessive discussion or arguing - Run some experiments, then decide which the Ensemble prefers.
- Zoning out - Take a short break with an agreement to focus. Take breaks more often.
- Ignoring roles/timers - Team members should hold each other accountable
- Producing poor designs or not valuing good design - An Ensemble that lacks people with good design skills won't magically produce good designs. To improve the design, the Ensemble should get expert help.
- Conflict between team members - In order to be a well-functioning Ensemble, everyone must embrace kindness, consideration, and respect as the way of working.

Don't Stop the Work - The Ensemble can temporarily delegate a member as a researcher, to find solutions to something they cannot easily figure out. Meanwhile, the Ensemble can work elsewhere in the code. It is important that people feel comfortable asking questions, but if the Ensemble is moving slowly due to a lot of questions, it is better to set aside time outside of the working session to answer questions. Fast throughput is an important goal.

Invite Experts - If an Ensemble gets stuck, they may invite an expert to join and help resolve a problem. Be sure it is an invitation and not a demand (the guest is not required to drive).

Resources

- Mob Programming – A Whole Team Approach, by Woody Zuill and Kevin Meadows
- Code with the Wisdom of the Crowd: Get Better Together with Mob Programming, by Mark Pearl
- The Mob Programming Guidebook, by Maaret Pyhäjärvi
- Mobster - free tool that helps teams manage rotations and breaks (http://mobster.cc)
- Tuple - remote pairing & ensemble app (https://tuple.app/)