



The Partners

- Partners are witnesses to each other's strengths and weaknesses.
- Confidence is a factor in successful pairings - partners must attain a level of comfort with each other.
- A person will pair differently with different partners. Pair programming is a dynamic and adaptive process.

The Driver

- The driver controls the keyboard and mouse.
- The driver records all tests and codes either on paper or in a file.
- The driver is actively talking to the copilot.
- Because the copilot is there to catch errors, the driver can concentrate on creating content, e.g. algorithm development.

The Copilot

- The copilot must be aware of everything that is going on - not a passive role!
- The copilot should be looking ahead and evaluating progress towards the goal.
- The copilot must also monitor details.
- The copilot must be ready to start driving at any time.

Pairing Up



- Do not always pair with the same person.
- Teams should evaluate what has to be done and who should pair to work on the tasks.
- After the tasks are done, new activities are planned and new pairings arranged.

The Bad Seed



- Some people will not work well in pairs:
 - The “Cowboy”
 - The “Passenger”
 - The “Party Animal”
 - The “Strong, Silent Type”

The “Cowboy”



- This person has a big ego and no patience for working with anyone that he/she doesn't consider their equal.
- Always wants to drive.
- Doesn't listen.
- Probably will never learn to work in a pair (or maybe even a team) situation.

The “Passenger”



- This person probably lacks confidence and is worried about looking bad in front of others.
- Never wants to drive.
- As a copilot, they never contribute anything useful.
- Might learn to pair programming as they build up more confidence in their skills.

The “Party Animal”



- Sees pair programming as an opportunity to gossip and talk about issues other than the task at hand.
- Lacks discipline and maturity.
- Has not learned how to concentrate.
- There may be hope for this person if other team members confront the problem immediately.

The “Strong, Silent Type”



- This person is highly skilled technically but lacks communication skills.
- Same effect as the “Cowboy” but for a different reason.
- A good partner may be able to make this work.
- Communication skills can be worked on.

Benefits of Pair Programming

But doesn't pair programming just use twice the resources to do the same amount of work?



Continuous Code Reviews



- The copilot can catch mistakes as they are happening.
- Three types of errors often caught by the copilot are:
 - typos
 - cognitive dropouts
 - invalid assumptions

Cognitive Dropouts



- A programmer can only focus on a limited part of the program at any one time.
- The copilot can not only provide extra cognitive capacity but this capacity is focussed at a different level since they are not concentrating on typing in the code.

Invalid Assumptions



- Easier to catch when two people are comparing their assumptions.
- Communication, in the form of justifying or explaining decisions can often reveal faulty reasoning and invalid assumptions.
- Articulating assumptions is useful in and of itself.

Time Management



- When you are working with a partner you must plan more - when and where to work.
- Since distractions will waste the time of both partners, there is more of a chance that someone will put the team back on track.
- Peer pressure to be disciplined and efficient is effective.

Learning



- Working with a more experienced partner can be a great way to learn about programming, designing, planning, standards and project details.
- Knowledge will spread easily throughout a team that uses pair programming.
- Experienced members of the team will benefit by seeing the project through other eyes.

Pair Programming Hints



- These hints to help you make Pair Programming a worthwhile experience are based on information in the book, *Pair Programming Illuminated*, by Laurie Williams and Robert Kessler (Addison-Wesley, 2003).

Let the driver have time to type.



- Do not immediately interrupt at the exact moment that a typing error is made. This is very, very annoying.
- Keep some paper handy to jot down thoughts about possible errors and then inject these thoughts at appropriate times.

Start your pairing session by establishing a few rules and guidelines for behaviour.



- Everyone works differently.
 - Try not to impose too much of your methodology on others unthinkingly.
- Be patient and try to learn how your partner works best.
 - If your partner is doing something that annoys you then tell them politely and early on in the process. Don't sit there getting more and more angry with your partner - they may not realize what they are doing that is so annoying and may find it very easy to change to make things better if you give them the chance.

Start your pairing session by establishing a few rules and guidelines for behaviour.



- Establish a coding style standard.
- Make sure that you communicate a lot - talking helps in this process.
- More importantly, be sure that you are listening to your partner.

There will be disagreements.



- Disagreements are natural and mean that you and your partner are actively working towards a solution to your problem.
- The trick is to be able to handle disagreements constructively.

There will be disagreements.



- Record your issues and disagreements on paper.
- It is a good idea to organize your thoughts so that you can stop coding and have a brainstorming session with your partner to see why these issues have arisen and what you can do to resolve them.

There will be disagreements.



- Time apart can also help.
- Even just a few minutes on your own can help you get your thoughts in order and start to approach the problem from a fresh angle.

There will be disagreements.



- Sometimes it will be helpful if you can decide to just let one person drive for a while and then review if the pair is happy with the result.
- If not then it is time for the other partner to take over and drive their way for a time. After both partners have had their turns a decision can be made as to which is the path to continue down.

Use test-driven development.

- Write tests before writing code.
- Test and write code incrementally.



If your partner is not listening then take action!

- Either get up and walk away to signal your displeasure or tell your partner that you should both take a break and come back in a few minutes to discuss the situation.
- Don't let the situation get out of control - act immediately and decisively.



Ask questions.

- If you do not understand what your partner is doing or why they are doing it, then question them.
- Both partners have to understand the code (or design, documentation, etc.) so remember not to shut out a partner that questions your code. You might find yourself in the same situation at a later time.



Be considerate.

- Cleanliness is important.
- Fresh breath is important too - you will be working closely together. Breath mints are better than gum - gum chewing can be annoying and is often not seen as an acceptable business habit.



Be considerate.



- Drinking coffee, pop, or juice is fine but food eating should take place at the appropriate time - not during work!
- Remember, if you are annoying your partner or they are annoying you, then you will not be a productive and effective team. A little consideration can go a long way.