Nice Pairing

Or, How To Maximise Pair Programming Value

Or, How To Make Things Easier On The People Who Work With You

Me

Former programming teacher

Now developing software at Lonely Planet

@adelsmee

adel.smee@lonelyplanet.com.au

Amateur food obsessive

http://pearshapedrecipes.tumblr.com

Pair Programming

What is it?

Assumptions

1. You, your company and your team understand the value of pair programming and support it.
2. You have a strategy in place to implement it.

Effective Pairing

- Two heads are better
- Reduces the silo effect and dilutes the diva effect
- Reduces bugs and code rewrites
- Fastest way to induct new team members
- Shares the best knowledge of the team with everyone
- Distributes code/infrastructure ownership across the team
- It is fun!

Charlotte: Future Software Developer

Wasteful Pairing

- What happens when pairing goes bad?
  - **Option 1:** Impotent frustration. Distraction. Waste of time. Waste of money. Propagates poor habits.
  - **Option 2:** Work with what you have. Examine your own behaviour.
    - How are you helping?
    - How are you not helping?
    - What can you do to change?

What's An Archetype?

- An archetype is a simplified model.
- Useful to mirror our behaviour when we can't see it ourselves.
- Pairing archetypes:
  1. Highlights – work to these strengths.
  2. Lowlights – minimise the impact.
  3. Check Yourself – techniques to emphasize the highlights and minimise the lowlights.

Finger-Operated Coder

Highlights

- Keyboard shortcuts legend
- Ideas generator
- Fast typist

Lowlights

- Keyboard hog
- Fidgeter
- Trouble explaining ideas
- Has to type out mistakes

Check Yourself (Driver)

- Take your hands off the keyboard.
- Spend more time navigating.
- Use your words/pen/paper instead of typing.

Check Yourself (Navigator)

- Keep pushing your pair to explain/talk/interact with more than just their hands.
- Get your turn in the driver's seat (e.g. pomodoro).

The Thinker

Highlights

- Always worth listening to
- Sees issues before they arise

Lowlights

- Thinking or sleeping?
- May look like not paying attention
- May not share all the good stuff

Check Yourself (Driver)

- Take a moment to tell your navigator you are in thinking mode.
- Find a technique that helps you to explain what is going on in your head (pen & paper/models).

Check Yourself (Navigator)

- Don't try and fill the dead air, give your pair time and space to think.
- Ask questions to get the conversation flowing.
-

The Cheerleader

Highlights

- Easy to work with
- Supportive when experimenting
- Confidence building

Lowlights

- Not critical
- Too focused on immediate problem
- Spends too much time navigating

Check Yourself (Driver)

- Use ping pong or pomodoro to ensure even driving time.
- Remind yourself to focus more on the code, less on the person you are coding with.

Check Yourself (Navigator)

- Ask for critical opinions.

The Brainiac

Highlights

- A great learning resource
- Able to anticipate issues before they arise
- Writes a lot of good code

Lowlights

- Keyboard hog
- If not inclined to share knowledge can be frustrating
- Can get carried away trying new things

Check Yourself (Driver)

- Always, always remember the smarter your team gets as a whole the better value you are to your company.
- Listen!

Check Yourself (Navigator)

- Ask questions, learn everything you can.
- Limit your pairing time to avoid becoming a spectator.

The N00b

Highlights

- Explaining code to a n00b can uncover bugs & refactoring opportunities
- N00b questions can highlight gaps in partner's knowledge
- Generates excitement about new stuff

Lowlights

- Requires patience from your pair
- Can slow down development in the short term

Check Yourself (Driver)

- Balance questions with listening.
- Try not to get too lost.
- Take responsibility for your own learning.

Check Yourself (Navigator)

- Accept the fact that development will be a little slower.
- Limit pairing with The N00b if you find teaching draining.

The Surfer

Highlights

- Good for mental break
- Passes the time when running tests
- Good resource for new tech

Lowlights

- The Internet is a Playground but you're at work!
- Kills the flow

Check Yourself (Driver)
- Don't share unless invited to.
- Isolate your cool stuff to company spam channel.

Check Yourself (Navigator)
- Keep directing attention back to the problem at hand.

The Talker

Highlights

- Excellent at describing problems and solutions
- Great brainstormer
- Good at extracting requirements

Lowlights

- Can dominate the pair
- Ideas hog
- Distracting to quieter partner

Check Yourself (Driver)

- Use the keyboard as well as your mouth.
- Remind yourself to listen, listen, listen.

Check Yourself (Navigator)

- Get your pair to put ideas on paper.
- Ask for what you need – "gimme a minute to think"

The Rock

Highlights

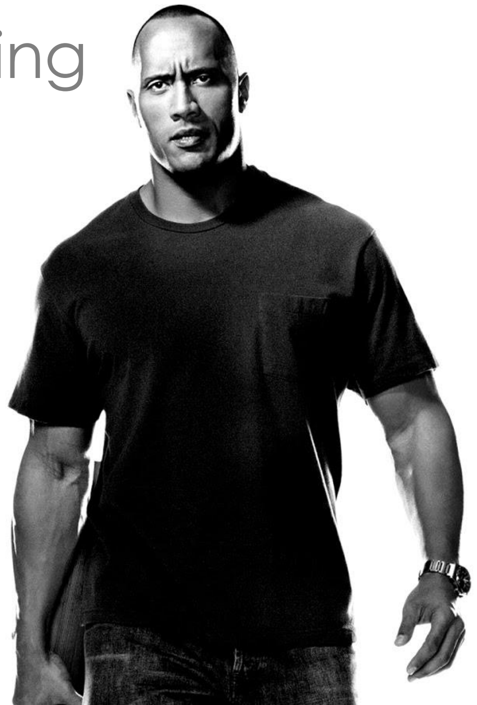
- Writes reliable code to best practices
- Minimises tech debt
- Excellent learning resource

Lowlights

- Creates scaffolding for throw-away code
- May lose sight of pragmatic programming
- Inflexible

Check Yourself (Driver)
- YAGNI.
- Remind yourself, again, of the downsides of premature
optimisation.

Check Yourself (Navigator)
- Keep checking in "what is our purpose",
"do we need that".



- Devs:
  - Can you see yourself in these archetypes?
  - Can you see your team in these archetypes?
  - What, if any, archetypes could you add?
  - How can you catch yourself in the act?
- Dev managers:
  - Run a tech session for your dev teams on Nice Pairing.
  - What characteristics does your team display:
    - Are these characteristics adding value, or diminishing it?
    - Brainstorm with your team on how to emphasize the good and reduce the bad.

Links

Nice Pairing blog
http://engineering.lonelyplanet.com/2013/08/09/Nice-Pair---Pair-Programming-Archetypes.html

Pair Programming explained
http://www.extremeprogramming.org/rules/pair.html
http://guide.agilealliance.org/guide/pairing.html

Pair Programming advocates
http://www.scribd.com/doc/25304465/null
http://www.versionone.com/Agile101/Pair_Programming.asp
http://www.airpair.com/pair-programming

The Pomodoro Technique
http://pomodorotechnique.com/

Ping Pong Programming
http://c2.com/cgi/wiki?PairProgrammingPingPongPattern