Pair Programming

Add Your Suggestions Here

What has worked for you? What tools do you use for screen sharing? Please add your tips and experiences! We will all benefit.

What is pair programming?

Pair programming is two (or more) people working on the same code at the same time.

If you're working in person, you sit at the same computer and take turns typing. If you're pairing remotely, you use a screen sharing application.

Typically one person is the "driver" - the person with the keyboard - and one person is the "navigator" - who suggests the next step to take. Both people talk about design choices, syntax and logical errors, etc. The programmers trade roles frequently.

How To Pair Program describes how to do a pairing session, step-by-step.

When and why might you pair program?

The advantages of pair programming are that we write better code with fewer mistakes, we're more familiar with more of the code base, working together can be more fun, and it tends to make new contributors become productive more quickly and stay with the community.

Critical or complex code

It's best to have more than one person review sensitive or complicated code, especially if it could affect other parts of OpenMRS. If it's something you'd request a code review for, it's a good candidate for pair programming.

Solving tough problems

Two brains are better than one.

Different skill sets

Perhaps you have different levels of expertise or different kinds of expertise. You might know all about Java, and your partner might know about HL7. A pairing session can get you up to speed quickly-- and that means more productive coders working on OpenMRS.

Expert-Expert

Coding with a colleague can be a great experience. You can get more done, learn some new tricks, or see other ways to think about a problem.

Novice-Novice

Two brains are better than one.

Software Tools

You need

- a way to talk, preferably voice chat (like Skype)
- a way to share a screen
- a way to share control of the computer if at all possible - so you're both actively working with the code

Plan on spending 30-60 minutes on getting your tools to work the first time you pair.

Screen sharing tools depend on what operating systems you're using and how much configuration you want to do.

The best way to find tools is to search for "remote pairing tools" on Google or Stack Overflow. (Consider limiting search results to the past year, as some go back 6+ years.)

Here are some good posts:

- Evan Light lists several configurations he's tried.
- StackOverflow posts:
    - Can you pair program remotely?

    - (2008) Good free screen-sharing app?
    - (2008) What's a great tool for remote pair development?
- pair.io gives you a one-button, collaboration-friendly dev environment for your GitHub repo . Still in alpha - could be great, especially with our GitHub switchover.
- RemotePairProgramming.com

Software Tool Reviews

Please add your reviews below.

iChat (Mac-Mac)

jriley: Worked the best for us so far. Occasional refresh errors on the screen, but responsive.

ScreenShare app (Mac - installed to /System/Library/CoreServices/ScreenSharing.app)

jriley: We couldn't get our machines to connect. iChat uses this to share -- it's easier to use iChat.

join.me (web)

jriley: Ok for viewing a screen, terrible for remote control. It ensured the navigator navigated instead of typed.

Adobe Connect - the OpenMRS U chat room - screen share only, no shared control

jriley: ok for viewing a screen, especially by multiple people. Without the shared keyboard and mouse, it's more of a lecture. You'll have to find ways to keep it a collaboration.

Untested:
Skype screen share

Chicken of the VNC (Mac)

TeamViewer - (Mac, Linux, Windows)

TightVNC (Windows and Linux)

Social Tools

Pair programming is about collaboration, which makes it different from programming alone.

//TODO:

//ping-pong programming - one writes test, one writes code to make it pass http://c2.com/cgi/wiki?PairProgrammingPingPongPattern

//make sure you're listening && collaborating (by...)

//if navigating, esp. while pairing remotely, get used to the pauses while the driver carries out the plan. They aren't stuck, they're thinking.

Tips

- Use an IDE you both know
- Collaborating is hard work. Plan sessions of only 1-2 hours.

Interesting Links

Unlikely Tools for Pair Programming - presentation from Open Source Bridge conference

Harnessing the Good Intentions of Others for your OSS projects - an OSCON presentation - how to lower the barrier between a volunteer expressing interest and contributing.

There are many introductions to pair programming and other agile development techniques - see Wikipedia or do Google search.