Extracted from:

Remote Pairing

Collaborative Tools for Distributed Development

This PDF file contains pages extracted from *Remote Pairing*, published by the Pragmatic Bookshelf. For more information or to purchase a paperback or PDF copy, please visit http://www.pragprog.com.

Note: This extract contains some colored text (particularly in code listing). This is available only in online versions of the books. The printed versions are black and white. Pagination might vary between the online and printed versions; the content is otherwise identical.

Copyright © 2013 The Pragmatic Programmers, LLC.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina

The Pragmatic Programmers

Pragmatic Press

Remote Pairing

Collaborative Tools for Distributed Development

Joe Kutner

Edited by Brian P. Hogan

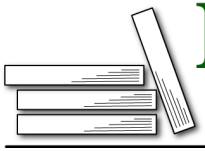
Remote Pairing

Collaborative Tools for Distributed Development

Joe Kutner

The Pragmatic Bookshelf

Dallas, Texas • Raleigh, North Carolina



Pragmatic Bookshelf

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and The Pragmatic Programmers, LLC was aware of a trademark claim, the designations have been printed in initial capital letters or in all capitals. The Pragmatic Starter Kit, The Pragmatic Programmer, Pragmatic Programming, Pragmatic Bookshelf, PragProg and the linking *g* device are trademarks of The Pragmatic Programmers, LLC.

Every precaution was taken in the preparation of this book. However, the publisher assumes no responsibility for errors or omissions, or for damages that may result from the use of information (including program listings) contained herein.

Our Pragmatic courses, workshops, and other products can help you and your team create better software and have more fun. For more information, as well as the latest Pragmatic titles, please visit us at *http://pragprog.com*.

The team that produced this book includes:

Brian P. Hogan (editor)
Candace Cunningham (copyeditor)
David J Kelly (typesetter)
Janet Furlow (producer)
Juliet Benda (rights)
Ellie Callahan (support)

Copyright © 2013 The Pragmatic Programmers, LLC.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher.

Printed in the United States of America.
ISBN-13: 978-1-937785-74-1
Encoded using the finest acid-free high-entropy binary digits.
Book version: P2.0—January 2014

Preface

In March 2013, Yahoo! CEO Marissa Mayer sent a companywide memo that banned employees from working at home. She wrote, "To become the absolute best place to work, communication and collaboration will be important, so we need to work side-by-side."[1]

Mayer's motivations for making this now-infamous decision have been debated ad finem. It may or may not have been the right move for Yahoo!, but one thing is clear: working side-by-side does not require physical presence.

In this book, you'll learn how to collaborate with remote coworkers in ways that are better than sharing the same location. You'll learn about tools so powerful that colocated pair programming teams often use them despite sitting side-by-side. The technologies that make this possible can increase your productivity and the quality of the code you produce. But without them, you may face the problems that concerned Marissa Mayer.

In the months following Mayer's announcement, she defended her decision by saying, "People are more productive when they work alone…they're more collaborative and innovative when they're together."[2] Again, Mayer's premise may be correct, but innovation can happen from any location when you have the right people, processes, and technologies.

Those are three keystones of effective remote pair programming: people, process, and technology. Throughout this book, we'll discuss how you can master the techniques of remote pair programming by using the best tools, integrating those tools into your organizational process, and accommodating each person's individual needs. We'll also discuss scientific research that provides evidence of the benefits or pair programming, and warns of many common pitfalls.

1. http://money.cnn.com/2013/02/25/technology/yahoo-work-from-home/index.html
2. http://tech.fortune.cnn.com/2013/04/19/marissa-mayer-telecommuting/

• Click HERE to purchase this book now. discuss

Remote pair programming can open the door to a more diverse workforce, happier programmers, increased levels of innovation, and the freedom to work from any location. But this book isn't just for programmers outside the office.

Who Should Read This Book?

Every programmer, whether working remotely or from an office, can benefit from the technologies and advice discussed in this book. Being physically remote is an optional part of remote pair programming, and many colocated developers prefer the techniques discussed in this book to sharing a physical space.

Traditional pair programming requires sharing a computer, which means you're also sharing a keyboard, mouse, and everything else you touch. The close proximity also means you share germs, viruses, and odors. But pair programming with remote techniques, even if you sit within speaking distance of your pairing partner, can make the experience more comfortable and productive.

The technologies covered in this book favor a certain type of developer—one who is comfortable with the command line. The techniques and tools you'll learn about are not limited to the terminal, but command-line tools are favored because they tend to require less bandwidth and handle high-latency networks better than many other mediums.

As a result, some of the techniques we'll discuss simply won't work with frameworks like .NET or iOS. But it's still possible, and desirable, to employ remote pair programming on applications that use these frameworks. All of the tools you'll need are discussed in this book, and many of the tools that are not applicable can be tweaked to suit your needs. For example, we'll discuss how to create an Elastic Compute Cloud (EC2) instance running Linux, but the same principles apply if you need a server running Windows. The book even contains alternate paths in many chapters for readers that are running Windows. No matter what platform you favor or what technologies you're working with, this book will have something for you.

If you're a remote worker, or if the people you work with are dispersed across the country, then you'll gain the most from this book. We'll discuss how to solve problems of latency, bandwidth, security, and connectivity. But the same techniques apply regardless of your location.

• Click HERE to purchase this book now. discuss

Why Should You Read This Book?

Pair programming can be emotionally draining. You have to cooperate with other programmers, their personal preferences, and their schedules. But making these compromises can greatly increase the quality of the products you create, and it can hone your technical skills. You have to ensure that you don't burn out, and with the right techniques you won't.

The most common causes of pair-programming burnout are technical issues. If you spend an hour before a pairing session just trying to get connected or if your connection drops midsession, then you're going to grow weary. Likewise, if you're using a textual editor that gives one half of the pairing team too much control, you might grow tired of feeling that you are not contributing to the work product.

In this book, you'll learn about networking techniques, textual editors, and many other tools that make the practice of pairing fun. They will eliminate the technical challenges that drain your energy.

What's in This Book?

This book covers three major paradigms of remote pair programming: text-only collaboration, screen sharing, and integrated development environment (IDE) collaboration.

We'll begin with a general discussion on pair programming in Chapter 1, *Introduction to Pair Programming*, on page ?. You'll learn some rules of etiquette, scientific research, and how these apply to a remote environment.

We'll address the first major paradigm in Chapter 2, *Collaborating with Text Only*, on page ?. You'll learn to use command-line tools for sharing and collaboratively editing a code base with another programmer across the Internet. We'll follow up with Chapter 3, *Using the Cloud to Connect*, on page ?, where you'll learn some tools and techniques for making reliable and secure connections with your partner.

In Chapter 4, *Collaborating with Shared Screens*, on page ?, we'll discuss the next major paradigm. You'll learn how to share your entire screen, but you'll also learn how to share just the parts you're using. Each method is appropriate in certain situations, which you'll learn about. Many commercial tools make screen sharing easy, but we'll focus on the free and open source technologies.

In Chapter 5, *Building a Pairing Server*, on page ?, we'll put some tools together and build a complete pairing server that can be easily re-created,

• Click HERE to purchase this book now. discuss

updated, shared, and destroyed. It can also create a more balanced experience for you and your partner.

We'll cover the last paradigm in Chapter 6, *Collaborating with an IDE*, on page ?. You'll learn about a robust pairing technology that runs within an IDE and how you can use it to reduce lag, increase responsiveness, and collaborate through many different interfaces.

Finally, we'll take a look at some real-world examples in Chapter 7, *Remote Pairing in the Wild*, on page ?. You'll learn how some of the most experienced programmers are using remote pairing to make their work better. We'll also discuss patterns you can use to provide structure in your pairing sessions.

What Do You Need to Use This Book?

An important tenet of pair programming is that it should accommodate a wide range of people and preferences. You should be able to pair-program with a partner who uses different tools than yours. That's why this book emphasizes cross-platform solutions. Many of the tools we'll discuss can be used from Mac, Linux, and even Windows. However, some of the tools favor certain platforms over others.

If you're running on Mac or Linux, you'll need to have a terminal-based editor installed. The examples we'll use favor Vim, but Emacs or any other solution will work. You'll also need a package manager, which is provided for you on most Linux systems, but you will need to install Homebrew on Mac OS X.[3]

If you're running Windows, you'll need a shell environment that supports SSH. Two great options are PuTTY,[4] a free implementation of SSH, and the Secure Shell plug-in for the Chrome browser.[5]

Regardless of your operating system, you'll need to install Vagrant, a virtual machine manager. Vagrant uses Oracle's VirtualBox to create virtual environments, so begin by downloading the VirtualBox installer from the Oracle site, and run it.[6] To install Vagrant, download and install the binary package for your platform from the Vagrant website, and run it.[7]

We can check that Vagrant was installed correctly by running the `vagrant` command like this:

3. http://brew.sh/
4. http://www.chiark.greenend.org.uk/~sgtatham/putty/
5. https://chrome.google.com/webstore/detail/secure-shell/pnhechapfaindjhompbnflcldabbghjo?hl=en
6. http://www.virtualbox.org/wiki/Downloads
7. http://downloads.vagrantup.com/

```
$ vagrant --version
Vagrant version 1.2.3
```

We'll use Vagrant throughout the book to create and configure pair-programming environments.

The next tool we'll need is RubyGems, which comes preinstalled with many Linux distributions (including the one we'll use with Vagrant) and even Mac OS X. You can check that it's available on your system by running this command:

```
$ gem -v
2.1.5
```

If the `gem` command does not work, you can install RubyGems with your system's package manager. On Debian-based Linux systems, run this command:

```
$ apt-get install ruby1.9.1
```

On Windows, download and follow the instructions for RubyInstaller.[8]

Those are the prerequisites, but there is one more thing you might want.

Having a Partner Is Optional

All of the examples in this book can be run without a partner, and we'll discuss tricks for testing these techniques without the assistance of another human. In most cases you won't even need a second computer, but in a few examples you may find it helpful to have an extra machine.

There will, however, come a time when you want to put the techniques in this book into action. Fortunately, a number of resources can help you find a partner. But the most convenient partner is probably a coworker.

When it comes to your coworkers, the best way to initiate a pairing session is to create an environment that is conducive to collaboration, and let partnership form organically. If the tools you need are right in front of you, then a session often evolves from a simple conversation about a piece of code. If your tools are not ready, the energy you need to pair will often fade before your environment is set up.

To be more deliberate when creating a session, you might head to your office's common area, log in to a chat room, or stick your head over a cubical wall and say, "Hey, anyone wanna pair?" It may work, but it's not the most formal

8. http://rubyinstaller.org/

way of starting a session. On the other hand, scheduling pairing sessions in advance may not work for some teams. Ultimately, the best way to get coworkers into a remote pair programming session will be different for every work environment. If your office is already using a system to schedule meetings and other events, then it might make sense to follow those same guidelines.

If you don't have a coworker or colleague to pair with, you can try using the #pairwithme tag on Twitter. Post a tweet describing what you'd like to work on, and a partner may reach out to you. Try something like "I want to dig into the Rails source code. Anyone want to #pairwithme?" Or you can look for programmers who have already posted a request to pair. There are some great aggregators for these requests.

The website at http://pair-with-me.herokuapp.com/ simply collects tweets that use the #pairwithme hash tag so you can browse or search them. There are also some technology-specific websites. The Ruby Pair website is helpful for linking up on projects using the Ruby programming language, while the Ember Pairs website is helpful for pairing on the Ember.js project.[9,10] The Pair Program with Me website provides dozens of other resources for finding a partner.[11]

Let's pair up.

9. http://rubypair.com/
10. http://www.emberpairs.com/
11. http://www.pairprogramwith.me/