

Fast Career Tack

Curated contents for software engineers.

10X Software Engineer

Learning Path

FABIO CICERCHIA

10x Software Engineer

Curated contents for software engineers.

Fabio Cicerchia

This book is for sale at <http://leanpub.com/10xse>

This version was published on 2021-04-14



Leanpub

This is a [Leanpub](#) book. Leanpub empowers authors and publishers with the Lean Publishing process. [Lean Publishing](#) is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

© 2019 - 2021 Fabio Cicerchia

Contents

Introduction	i
Preface	iii
Audience	iii
Competency Levels	iii
Contents of This Book	iv
How to Use this Book	viii
10xSE Academy	viii
Level: Advanced Beginner	1
Schedule	2
Week 1: Build Up Dictionary	3
Week 2: Ground Rules & Manifestos	20
Coding Standards	20
Etiquette	21
Job	21
Manifestos	22
Roadmaps	23
Roles	23
Skills	24
VCS	24
Books	24

CONTENTS

Week 3: SDLC	28
Agile	28
Time Management	28
XP	29
Mix	29
Books	29
Week 4: Algorithms	32
Articles	32
Videos	33
Week 5: OOP	36
Basics	36
Calisthenics	36
Clean Code	37
Cohesion & Coupling	37
SOLID	37
Videos	37
Books	38
Week 6: Design Patterns	41
Creational	41
Structural	42
Behavioral	42
Books	44
Week 7: Design Patterns	47
UML	47
Books	47
Week 8: Asynchronous Programming	50
Basics	50
Messages	51
Week 9: Testing	54
Arrange Act Assert	54

CONTENTS

Bottlenecks	54
Mocks & Stubs	54
Mutation	55
Test Pyramid	55
Test Driven Development	55
Mix	55
Videos	56
Books	56
Week 10: Refactoring	59
Articles	59
Code Smells	60
Refactoring Techniques	62
Books	68
Week 11: Refactoring	71
Software Development AntiPatterns	71
Software Architecture AntiPatterns	72
Project Management AntiPatterns	73
Books	75
Week 12: DB	78
ACID properties	78
Consistency	78
Joins	79
Normalisation	79
Sharding	79
Mix	80
Books	80
Week 13: Extra Resources	83
RegEx	83
VCS	86
SEO	89
Books	93

CONTENTS

Level: Intermediate	94
Schedule	94
Week 14: Build Up Dictionary	96
Week 15: Caching	112
Antipatterns	112
Best Practice	112
ESI	113
Invalidation	113
Security	113
Strategies	113
Web	114
Videos	114
Books	115
Week 16: DDD, Functional & CQRS/ES	118
Books	118
Week 17: DDD, Functional & CQRS/ES	121
DDD	121
Functional	122
CQRS	123
Event Sourcing	123
Videos	124
Books	124
Week 18: Networking	127
HTTP	127
IP	127
Time	128
Books	128
Week 19: Cloud	131
Cloud	131
Best Practices	132

CONTENTS

Books	132
Week 20: DevOps	135
Deployments	135
Infrastructure	135
MVP	136
SRE	136
Tools	136
Mix	137
Books	137
Week 21: Security	141
SSL/TLS	141
Mix	141
Books	142
Week 22: Architecture	145
Distributed	145
Documentation	145
Enterprise	145
Event-Driven	146
Hexagonal	146
Microservices	146
The Twelve Factors App	147
Books	148
Week 23: Architecture	151
Mix	151
Videos	151
Books	152
Week 24: Jobs	155
Freelancing	155
Job Offer	155
Ladder	156
Preparation	156

CONTENTS

Questions	156
Quit	157
Remote	158
Resume	158
Books	159
Level: Advanced	162
Schedule	162
Week 24: Build Up Dictionary	164
Week 25: OS	168
Virtualization	168
Container	168
Shell	169
Mixed	169
Week 26: Productivity	176
Cognitive Biases	176
Critical Thinking	176
Decision Making	176
Destructive Approaches	177
Stress	177
Time Management	178
Mix	178
Videos	179
Books	179
Week 27: Standards & Best Practices	183
Architecture	183
Data	183
Design	184
HA	184
Observability	184
Scalability	184

CONTENTS

Security	185
Week 28: Standards & Best Practices	188
Books	188
Week 29: Management	192
Approaches	192
Leadership	192
Product/Project Management	193
Public Speaking	193
Roles	194
Quit	194
Mix	195
Books	195
Week 30: Management	198
Books	198
Week 31: Management	201
Videos	201
Books	202
 Appendices	 205
Appendix A: Extra Learning Paths	206
Appendix B: Exercises	208
Week 2	208
Week 4	209
Week 5	210
Week 6	211
Week 11	211
Week 12	212
Week 15	213
 Appendix C: More IT Books	 214

CONTENTS

Appendix D: IT Trends 215

Appendix E: Tech People 216

Introduction

Hello, I am Fabio Cicerchia, a Passionate Solutions Architect and Application Developer with 15+ years of experience. Always enjoying creating quality web applications and web portals using cutting-edge technologies.

Working in several positions, from Software Developer to Frontend/Backend Developer, from Sysadmin to Team Leader, allowed me to work on each layer of a web application, covering the whole life-cycle from initial requirements gathering to design, planning, coding, testing, documentation, deployment, and maintenance.

I've started to move my first steps in the programming world, at the "late" age of 15 years old...

And I decided to undertake that kind of career without asking myself too many questions.

I've started, actually, developing software during my years spent in high school (obviously in the computer science specialisation) and then decided to focus totally on the web. I continue nowadays to spend my spare time learning new things (from methodologies to new technologies) keeping myself up-to-date or at least trying to do it.

Because, especially in this field, who hesitates is lost. Let's acknowledge this.

My career started as a freelancer, and then settled down in various companies, in various roles (still as a developer), in various industries such as e-commerce, marketing, web agency, analytics.

I continued moving up the ladder, switching to management and reaching the "peak" as CTO and then going back to the technical track.

Now I find myself years later from those my first steps, with HTML and VB, and still, remember the day in which I've opened the

website [HTML.it](http://www.html.it)¹ to start to learn the rudiments of what brought me where I am now.

A lot of satisfaction and few regrets.

Therefore I've decided to sum up all my 15+ years of experience about the things I've learned, tips and suggestions collected over time, with the hope that might be useful to you.

Consider this ebook as a starting point and not as a list of things to be blindly followed. Bear in mind that lots of things may be subjective, but I tried to be as much objective and unbiased as possible.

'Nuff said, happy reading!

Contacts

- [Web Site](http://www.html.it)²
- [Twitter: @fabiocicerchia](https://twitter.com/fabiocicerchia)³
- [LinkedIn](http://linkedin.com/in/fabiocicerchia/)⁴

¹<http://www.html.it>

²<https://fabiocicerchia.it>

³<https://twitter.com/fabiocicerchia>

⁴<http://linkedin.com/in/fabiocicerchia/>

Preface

Audience

The ebooks will cover all the stages of a developer career, from the beginning to the advanced roles, the main core is focusing on improving your career when you've already started.

Therefore this is mainly for junior and mid-level programmers, but it has good points and useful information even for experienced developers as well.

The technical skillset as a software developer is in high demand since technology is a huge part of our lives and no company can afford to survive without IT. Companies do not need just software developers, they need software engineers with soft skills and breadth of knowledge.

Competency Levels

In the book will be used 3 of the levels defined by the Dreyfus model:

- Advanced Beginner
- Competence
- Proficient

The **Dreyfus model of skill acquisition** is a model of how learners acquire skills through formal instruction and practising, used in the fields of education and operations research. Brothers Stuart and Hubert Dreyfus

proposed the model in 1980 in an 18-page report on their research at the University of California, Berkeley, Operations Research Center for the United States Air Force Office of Scientific Research. The model proposes that a student passes through five distinct stages and was originally determined as: novice, competence, proficiency, expertise, and mastery.

- https://en.wikipedia.org/wiki/Dreyfus_model_of_skill_acquisition

The book won't cover the *Novice* level, because it is required to have some knowledge about programming.

Also, it won't be covered about the *Expert* level, because the book will provide enough knowledge to be competent in many areas of the IT field, but giving the knowledge to be expert in all of them won't be realistically possible.

Contents of This Book

In each chapter, I provide a list of useful contents (articles, videos and/or books) to be actioned during the week. It is recommended to follow the order provided as the schedule will allow to build up the required knowledge.

I do not take credit about those external resources, but I relate and agree with. In this, you can find tips and suggestion extracted from my own career and from what I learned so far.

This is not a bible nor a reference manual, I strongly recommend you to do not follow any suggestion blindly without understanding the reasons behind what you'll read.

I hope while reading and maybe assimilating (un)consciously a few concepts, you'll find a way to improve yourself.

That's the aim of this book!



Disclaimer: Copyright & Legal & IP

Most of the data contained are of public domain, available on Google Search. The ebook, the website <https://10xse.academy/> and the course itself is built on the 10x SE Learning Path: scouting, gathering, categorisation, structuring outline, maintenance, architecture & infrastructure, and so on. The links provided to the external content, and the external content itself, belongs to the content's author(s) and they are provided as-is: more details on [T&C](#)⁵. Should you wish to have your website removed can be done in the [removal page](#)⁶.

⁵<https://10xse.academy/wpautoterms/terms-and-conditions/>

⁶<https://10xse.academy/website-removal>



Disclaimer: 10x Myth

I do believe in the 10x, in the sense that one developer could achieve x-times more of others developers in certain circumstances, for example, previous knowledge of the domain, previous knowledge of the language/framework, knowledge of the project, some good time management skills.

There's so much hype about the "myth" of the 10x engineer. Please do read more about it:

- [□ The origins of the 10x developer⁷](#)
- [□ What makes a 10x programmer/software engineer?⁸](#)
- [□ The 10x Programmer: Is Individual Productivity Overrated?⁹](#)
- [□ The "10X Engineer" Has Officially Become a Meme¹⁰](#)
- [□ How To Become A 10X Engineer¹¹](#)
- [□ You'll never be a 10x Developer¹²](#)

⁷<https://medium.com/ingeniouslysimple/the-origins-of-the-10x-developer-2e0177ecef60>

⁸<https://www.quora.com/What-makes-a-10x-programmer-software-engineer/answer/Edward-De-Jong>

⁹<https://thenewstack.io/the-veracity-and-relevance-of-the-10x-programmer/>

¹⁰<https://www.7pace.com/blog/10x-engineers>

¹¹<https://blog.codegiant.io/how-to-become-a-10x-engineer-492fa3f57101>

¹²<https://medium.com/dev-genius/youll-never-be-a-10x-developer-3312c1f003ed>



Disclaimer: External Links and Reading Time

All the displayed reading time are just an approximation based on average reading times, most people read around 250/300 wpm, so as a baseline I've used 250 words per minute, and an average page length of 450 words per page. This is pretty standard relaxed time, to read one page it'll take into consideration 1 minute and 48 seconds, not too slow not too fast. If you're a faster reader you can devour all those links in less time than suggested, if you're slower than that who cares, as long as you can get the concepts :)

You might be wondering why to create a book on external content, can't I create my own?! Sure, but... Those external contents are there for a reason, to be shared. In this way, you'll know many Developers or Architects or CTOs who are sharing their knowledge, rather than just share only my own. You could bookmark those blogs, subscribe to their feeds, read the other books, check the related news or similar books, get the most up to date content from their latest video on YouTube, or attend to a conference where they have a talk. Coming from different roles, different backgrounds, different styles. Cross-contamination of ideas and skills, it's great to learn from others and grow faster than expected.

Since the whole book is based on a curated collection of external resources, despite the effort taken to make sure all the links are always available, it might happen that some of them are not available any more.

Do not worry about it! I've got you covered, so does the Internet Archive.

By using the [Wayback Machine](#)¹³ you can access to a snapshot of the page and read the stored content. If you want to know more about the web archiving functionality there's a [Help Center](#)¹⁴ section full of details.

¹³<https://archive.org/web>

How to Use this Book

- Always start from the lowest level, even if you're more experienced.
You'll never know you'll find something interesting.
- Use the Checklists to mark your progress.
Each item has a checkbox () in front of, used it with a pencil to record what you have read/studied.

10xSE Academy

The ebook is the foundation of the learning path, so I've decided to improve it and add an online course with assignments, exercises, book reviews, grades, newsletter, peer reviews and much more.

I can offer two special discounts just for the ebook readers:

- **14.99 EUR (instead of 47.88 - save 32.89) for 1 year subscription access** for all courses and materials, even future ones.
- **24.99 EUR for lifetime access** for all courses and materials, even future ones.

To redeem the offer just go to <https://10xse.academy/redeem-ebook> and enter the code Z98DQ8A0S30.

¹⁴<https://help.archive.org/hc/en-us/categories/360000553851-The-Wayback-Machine>

Level: Advanced Beginner

Level Definition

The novice evolves by figuring out the mistakes in his work. The newly, “promoted” advanced beginner dwells into the world of troubleshooting. Unfortunately, the hasty mindset is not lost, and the individual still aims to acquire results fast, in this case gain knowledge and information. An example would be when a coder with years of experience starts learning a new program language, he could be a master in PHP but an advanced beginner in Python. Scrolling through the documentation will not lead to productive results.

- <https://www.360pmo.com/the-five-dreyfus-model-stages/>



Duration: 13 weeks (~3 months)
Average per week: ~11 hours

Schedule

- Week 1: Build Up Dictionary
- Week 2: Ground Rules & Manifestos
- Week 3: SDLC
- Week 4: Algorithms
- Week 5: OOP
- Week 6: Design Patterns
- Week 7: Design Patterns
- Week 8: Asynchronous Programming
- Week 9: Testing
- Week 10: Refactoring
- Week 11: Refactoring
- Week 12: DB
- Week 13: Extra Resources

Week 1: Build Up Dictionary

0-9

2FA - Two Factor Authentication

The 2FA is an additional layer of security you can set up to keep your account secure. It requires a unique one-time use code.

A

A/B Testing

A/B testing is an experimentation method by comparing two (or more) variants of a webpage or app against each other to determine which one performs better.

AAA - Authentication Authorization Accounting

AAA refers to Authentication, Authorization and Accounting.

ACID - Atomicity Consistency Isolation Durability

An ACID database system guarantees that transactions are processed reliably, following the 4 properties: Atomicity, Consistency, Isolation, Durability.

Active Record

Active Record pattern is a pattern used to store object data in the database. There's a direct relationship with the database schema and the basic CRUD operations.

Agile

Agile is an iterative approach to project management that helps to deliver value to the customers faster.

Algorithm

A logical approach which is a well-defined list of steps that allows a computer to solve a problem.

Antipattern

Antipatterns are apparently appropriate solutions to problems, but in reality, are ineffective or result in unexpected consequences.

API - Application Programming Interface

An API is an interface that lets your service communicate with external services without them knowing the implementation details.

B

Big Data

Big data is a term that describes a large volume of data, that can be analysed for better decision making insights.

Blockchain

Blockchain is a shared, distributed and immutable ledger that facilitates the process of recording transactions.

Bug

A bug is an error, flaw or fault in a computer program that causes an incorrect or unexpected result.

C

Cache

A cache is a data storage layer which allows you to quickly serve previously retrieved or computed data.

CDN - Content Delivery Network

A CDN is a geographically distributed platform that helps reducing delays in web page content loading by reducing the physical distance between the server and the user.

Chatbots

A chatbot is an artificial intelligence software that can simulate a conversation with a user via messaging applications.

CLS - Cumulative Layout Shift

CLS is a metric for measuring visual stability of the web page, it helps quantify how often users experience unexpected layout shifts.

Code Review

A code review is the process (manual or automatic) of checking the source code for mistakes and improve the overall quality.

Container

A container is a set of processes that are isolated from the rest of the system. They are portable and consistent through different environments.

CORS - Cross-Origin Resource Sharing

CORS is a mechanism that grants the browsers access to resources outside the scope of the current origin.

CRUD - Create Read Update Delete

CRUD are the four basic functions that models should be able to do to implement persistent storage.

D

Database Normalization

Database Normalization is a technique for eliminating data redundancy.

Deadlock

A deadlock occurs when two threads are holding the locked variable that the other thread wants, nothing occurs, and the threads remain deadlocked.

Design Patterns

A design pattern is a general repeatable solution to a common problem in software design.

DOMContentLoaded

The DOMContentLoaded event is triggered when the HTML has been completely loaded and parsed, without waiting for assets to load.

E

EAV - Entity Attribute Value

Entity-attribute-value is a data model used to store a variable number of entity attributes in a table's space-efficient manner.

Environment Variable

An environment variable is a named, global, shared variable that contains data used by one or more applications.

F

FID - First Input Delay

FID is a metric for measuring the time from when a user first interacts with a page.

FCP - First Contentful Paint

FCP is a metric for measuring the time from when the page starts loading to when any part of the page's content is rendered on the screen.

G

GDPR - General Data Protection Regulation

The GDPR is a privacy and security law, drafted by the European Union, it defines obligations to worldwide organizations in case they're involved with data related to people in the EU.

GTD - Getting Things Done

Getting Things Done is a time management method, it is based on recording tasks and activities and breaking them down in actionable work.

H

Hydration

Most ORMs are performing a hydration process when converting database results into objects, it usually involves reading on-the-fly a record (or additional fields) and then make them available in the record object.

I

Idempotence

An HTTP method is idempotent if an identical request can be made more than once without any side-effects leaving the server in the same state.

Information Architecture

Information Architecture focuses on organizing and effectively structuring content.

Invariant

An invariant is a property of the program state that is always conceptually true.

IoT - Internet of Things

The Internet of Things describes the network of physical objects with embedded technologies to connect and exchange data with other devices and systems over the internet.

K

Kanban

Kanban is a framework used for agile software development, where work items are visually available for the team members to know the state of every piece of work at any time.

Key-Value

A key-value database is a type of non-relational database that uses simple key-value pairs to store data.

L

LCP - Largest Contentful Paint

LCP is a metric reporting the render time of the largest image or text block visible within the viewport.

Lean

Lean is a management philosophy inspired by Toyota practices to minimise risk and waste while maximizing customer value.

Legacy

A legacy system is an old system still in use, that usually drives the business, that is out of date or in need of replacement.

M

Machine Learning

Machine learning is a data analysis method that allows systems to learn from data, identify patterns and make decisions with minimal human intervention.

MFA - Multi-Factor Authentication

MFA may use three or more checks to verify and authenticate customer identity.

Mobile-First

A mobile-first approach involves designing a website starting with the mobile version, with your mobile users in mind, and then adapt to larger screens.

Mockup

Mockups are essentially wireframes with an added surface layer that communicates the visual design to suggest what the final design will look like.

Murphy's Law

If something can go wrong, it will.

MVP - Minimum Viable Product

An MVP is a version of a new product which allows collecting assumptions about customers with the least effort.

N

NoSQL - Not Only SQL

NoSQL databases are non-tabular (document, key-value, wide-column, and graph) and store data differently than relational tables.

O

OOP - Object Oriented Programming

OOP is a programming model that organizes software design around objects, rather than functions and logic, that has unique attributes and behaviour.

ORM - Object Relational Mapping

ORM allows writing SQL queries using the object-oriented paradigm of a programming language.

OWASP - Open Web Application Security Project

The OWASP is a non-profit foundation that works to improve the security of software.

P

Pair Programming

Pair programming is a collaborative way of developing code in pair and involves a driver and a navigator.

PoC - Proof of Concept

A Proof of Concept is a demonstration to verify that certain concepts of a project or product are feasible and worthy enough to justify the expenses needed to support them.

PWA - Progressive Web App

PWA is a web app built to look and feel an actual native app.

Q

QA - Quality Assurance

Quality Assurance is a process for preventing mistakes and defects when delivering products or services to customers.

R

Race Condition

A race condition occurs when two threads write a shared variable at the same time.

RDBMS - Relational Database Management System

Relational Database Management System is an advanced version of a DBMS, stores the data in the form of tables, rows and columns.

RegEx

A regular expression allows you to create patterns that help match, locate, and manage text.

Remote Work

Remote work allows professionals to work outside of a traditional office environment.

REST - REpresentational State Transfer

REST is an architectural style for distributed APIs.

RPC - Remote Procedure Call

A RPC is a protocol that let you communicate with external services without knowing the implementation details.

Reverse Engineering

Reverse Engineering is a process of acquiring the knowledge of how a software works by recovering the design and specifications of a product from an analysis of its code.

Rubber Duck

Rubber duck debugging is a method of debugging code, by forcing one to explain it, line-by-line, to the duck.

S

Scrum

Scrum is a framework that allows people to address complex problems while delivering products.

SEO - Search Engine Optimization

SEO is the practice of increasing a website traffic quantity/quality through organic search engine results.

Spikes

A spike is a user story for which the team cannot estimate the effort needed, so it will be executed in a time-boxed exploration to learn about the issue or the possible solutions.

SOAP - Simple Object Access Protocol

SOAP is an XML-based protocol for accessing web services over HTTP.

SOLID

SOLID is an acronym for the five object-oriented design principles to develop a software that is easy to maintain and extend: Single-responsibility principle, Open-closed principle, Liskov substitution principle, Interface segregation principle, Dependency Inversion Principle.

SPA - Single Page Application

A SPA is a web application that dynamically rewrites the current web page with new data from the webserver based on user interactions.

SQL - Structured Query Language

SQL is a programming language used in a relational database.

SSL/TLS

SSL/TLS works by binding websites to a cryptographic key pair via certificates.

T

TBT - Total Blocking Time

TBT is a metric for measuring the total amount of time between FCP and TTI where the main thread was blocked for long enough to prevent input responsiveness.

Testing

Testing is the activity (manual or automated) of checking whether the actual results match the expected results and to ensure that the software system is free from defects.

TTFB - Time To First Byte

TTFB is the amount of time it takes to receive the first byte of an HTTP request from the webserver.

TTI - Time to Interactive

TTI is a metric for measuring the time from when the page starts loading to when its main sub-resources have loaded and it is capable of reliably responding to user input quickly.

V

VCS - Version Control System

Version control systems are tools that help to manage changes over time.

W

Waterfall

The waterfall model is a strictly linear and sequential of phases, where each phase depends on the deliverables of the previous one.

Web Vitals

Web Vitals are metrics developed by Google to measure quality signals for delivering great user experience.

Wireframe

A wireframe is a schematic model that is useful to communicate about the structure of the software or website.

WSDL - Web Services Description Language

WSDL is an XML format for describing services endpoints and message formats.

X

XP - Extreme Programming

Extreme Programming is a software development methodology to improve software quality and responsiveness to changing customer requirements.

Week 2: Ground Rules & Manifestos



Time needed: 20 hours

Coding Standards

1. [Semantic Versioning 2.0.0](#)¹⁵
Semantic Versioning 2.0.0 | Semantic Versioning
2. [Arlo's Commit Notation](#)¹⁶
GitHub - RefactoringCompos/ArlosCommitNotation: A notation for small commits messages that show the risk involved in each step
3. [Awesome Guidelines](#)¹⁷
GitHub - Kristories/awesome-guidelines: A curated list of high quality coding style conventions and standards.
4. [HTTP headers for the responsible developer](#)¹⁸
HTTP headers for the responsible developer - Twilio
5. [Pragmatic Programming Cheat Sheet](#)¹⁹
Pragmatic Programming Cheat Sheet by marconlsantos - Download free from Cheatography - Cheatography.com: Cheat Sheets For Every Occasion

¹⁵<https://semver.org/>

¹⁶<https://github.com/RefactoringCompos/ArlosCommitNotation>

¹⁷<https://github.com/Kristories/awesome-guidelines>

¹⁸<https://www.twilio.com/blog/a-http-headers-for-the-responsible-developer>

¹⁹<https://cheatography.com/marconlsantos/cheat-sheets/pragmatic-programming/>

Etiquette

1. [Netiquette](#)²⁰
Netiquette : Florida Atlantic University
2. [50 Amazing Office Etiquette Tips to Transform Your Company Culture](#)²¹
50 Amazing Office Etiquette Tips to Transform Your Company Culture - Small Business Trends
3. [Dev etiquettes that you must not ignore](#)²²
Dev etiquettes that you must not ignore | by Madhav Bahl | codeburst
4. [Seven principles of pair programming etiquette](#)²³
Seven principles of pair programming etiquette | by Juntao Qiu | ITNEXT
5. [Developer Etiquette – Code Review and Pull Request Comments](#)²⁴
Pull Request Etiquette - A set of simple rules for your code review · Erik Zaadi

Job

1. [6 Ways To Get Noticed At Work](#)²⁵
6 Ways To Get Noticed At Work - Business Insider Business Insider logo Close icon Loading Menu icon Search icon Business Insider logo Account icon Account icon Business Life News Reviews Search icon Insider logo Close icon Business Life News All Account icon World globe Facebook Icon Twitter icon LinkedIn icon YouTube icon Instagram icon Business

²⁰<https://www.fau.edu/oit/student/netiquette.php>

²¹<https://smallbiztrends.com/2017/06/office-etiquette.html>

²²<https://codeburst.io/dev-etiquettes-that-you-must-not-ignore-619e1bb490b8>

²³<https://itnext.io/seven-principles-of-pair-programming-etiquette-74a2b3b233b0>

²⁴<https://erikzaadi.com/2019/09/29/pull-request-etiquette-a-set-of-simple-rules-for-your-code-review/>

²⁵<https://www.businessinsider.com/6-ways-to-get-noticed-at-work-2013-8>

- Insider logo Close icon Chevron icon Chevron icon Facebook icon Email icon Link icon Twitter icon LinkedIn icon Fliboard icon More icon Close icon Loading Close icon
2. [How to Make Yourself Indispensable at Work²⁶](#)
How to Make Yourself Indispensable at Work
 3. [Seven Ways to Be a Good Employee and Make Your Boss Happy²⁷](#)
Seven Ways to Be a Good Employee and Make Your Boss Happy
 4. [The Top 10 Signs That You Are An Impostor At Work²⁸](#)
The Top 10 Signs That You Are An Impostor At Work
 5. [Internal Developer Training: Doing It Right²⁹](#)
Internal Developer Training: Doing It Right — Smashing Magazine Clear Search Back to top
 6. [6 Stupid Mistakes Smart Developers Should Make³⁰](#)
6 Stupid Mistakes Smart Developers Should Make - SitePoint SitePoint
 7. [Five Career Mistakes That Might Be Holding You Back³¹](#)
Five Career Mistakes That Might Be Holding You Back

Manifestos

1. [Manifesto for Agile Software Development³²](#)
Manifesto for Agile Software Development

²⁶<https://lifehacker.com/how-to-make-yourself-indispensable-at-work-1113590784>

²⁷<https://lifehacker.com/seven-ways-to-be-a-good-employee-and-make-your-boss-happy-1622335033>

²⁸<https://www.forbes.com/sites/kathycaprino/2013/08/14/the-top-10-signs-that-you-are-an-impostor-at-work>

²⁹<https://www.smashingmagazine.com/2014/09/internal-developer-training-doing-it-right/>

³⁰<https://www.sitepoint.com/6-stupid-mistakes-smart-developers-should-make/>

³¹<https://lifehacker.com/five-career-mistakes-that-might-be-holding-you-back-1596535994>

³²<https://agilemanifesto.org/>

2. [Manifesto for Software Craftsmanship](#)³³
Manifesto for Software Craftsmanship
3. [Refactoring Manifesto](#)³⁴
Refactoring Manifesto - Because the world needs better code
4. [Software disenchantment](#)³⁵
Software disenchantment @ tonsky.me
5. [The Reactive Manifesto](#)³⁶
The Reactive Manifesto

Roadmaps

1. [Developer Roadmaps](#)³⁷
Developer Roadmaps

Roles

1. [Job Titles & Levels: What Every Software Engineer Needs to Know](#)³⁸
Job Titles & Levels: What Every Software Engineer Needs to Know — Holloway
2. [How to Find Your Career Path](#)³⁹
How to Find Your Career Path
3. [The Taxonomy of Terrible Programmers](#)⁴⁰
The Taxonomy of Terrible Programmers – Aaronontheweb

³³<http://manifesto.softwarecraftsmanship.org/>

³⁴<https://refactoringmanifesto.org/>

³⁵<https://tonsky.me/blog/disenchantment/>

³⁶<https://www.reactivemanifesto.org/>

³⁷<https://roadmap.sh/>

³⁸<https://www.holloway.com/s/trh-job-titles-levels-fundamentals-for-software-engineering>

³⁹<https://lifehacker.com/top-10-ways-to-find-your-career-path-1628537579>

⁴⁰<http://www.aaronstannard.com/the-taxonomy-of-terrible-programmers/>

4. [□ The full-stack employee⁴¹](#)
The full-stack employee. Defining a new class of hybrid worker. | by Chris Messina | Chris Messina | Medium

Skills

1. [□ Evergreen Skills for Software Developers⁴²](#)
GitHub - romenrg/evergreen-skills-developers: List of evergreen skills, based on software development best practices & cross-framework principles, that should serve as a fair assessment of skilled software engineers / developers

VCS

1. [□ 6 Version Control Systems Reviewed⁴³](#)
6 Version Control Systems Reviewed — Smashing Magazine
Clear Search Back to top
2. [□ CS Visualized: Useful Git Commands⁴⁴](#)

Books

1. [□ The Mythical Man-Month⁴⁵](#)

⁴¹<https://medium.com/chris-messina/the-full-stack-employee-ed0db089f0a1>

⁴²<https://github.com/romenrg/evergreen-skills-developers>

⁴³<https://www.smashingmagazine.com/2008/09/the-top-7-open-source-version-control-systems/>

⁴⁴<https://dev.to/lydiahallie/cs-visualized-useful-git-commands-37p1>

⁴⁵<https://www.amazon.com/Mythical-Man-Month-Essays-Software-Engineering/dp/0201835959>

2. □ [Computer Science Distilled](#)⁴⁶
Computer Science Distilled

⁴⁶<https://sourcemaking.com/computer-science-distilled>

Notes

(free space)

Notes

(free space)

Week 3: SDLC



Time needed: 13 hours

Agile

1. [Explaining Agile](#)⁴⁷
Explaining Agile
2. [Agile Patterns](#)⁴⁸
Agile Patterns - Dzone Refcardz

Time Management

1. [The Pomodoro Technique®](#)⁴⁹
The Pomodoro Technique® - proudly developed by Francesco Cirillo | Cirillo Consulting GmbH
2. [Types of Procrastination \(And How To Fix Procrastination And Start Doing\)](#)⁵⁰

⁴⁷<https://www.forbes.com/sites/stevedenning/2016/09/08/explaining-agile/>

⁴⁸<https://dzone.com/refcardz/agile-patterns>

⁴⁹<https://francescocirillo.com/pages/pomodoro-technique>

⁵⁰<https://www.lifehack.org/articles/productivity/types-procrastination-and-how-you-can-fix-them.html>

XP

1. [Essential XP: Emergent Design](#)⁵¹
Essential XP: Emergent Design
2. [BeckDesignRules](#)⁵²
BeckDesignRules

Mix

1. [Zero trust architecture design principles](#)⁵³
GitHub - ukncsc/zero-trust-architecture: Principles to help you design and deploy a zero trust architecture
2. [Principles of secure development & deployment](#)⁵⁴
GitHub - ukncsc/secure-development-and-deployment: NCSC Guidance for secure development and deployment

Books

1. [Agile Estimating and Planning](#)⁵⁵
2. [Agile Patterns](#)⁵⁶
Agile Patterns - Dzone Refcardz

⁵¹<https://ronjeffries.com/xprog/classics/expemergentdesign/>

⁵²<https://martinfowler.com/bliki/BeckDesignRules.html>

⁵³<https://github.com/ukncsc/zero-trust-architecture>

⁵⁴<https://github.com/ukncsc/secure-development-and-deployment>

⁵⁵<https://www.amazon.com/Agile-Estimating-Planning-Mike-Cohn/dp/0131479415>

⁵⁶<https://dzone.com/refcardz/agile-patterns>

Notes

(free space)

Notes

(free space)

Week 4: Algorithms



Time needed: 2 hours

Articles

1. [Algorithms {fundamental techniques}](#)⁵⁷
Algorithms - Wikibooks, open books for an open world
2. [Algorithms](#)⁵⁸
Algorithms - GeeksforGeeks
3. [IDEA – nonverbal algorithm assembly instructions](#)⁵⁹
IDEA – nonverbal algorithm assembly instructions
4. [Why do students fail in Algorithms and Data Structure Interviews for Top Companies? | by Shubham Gautam | Medium](#)⁶⁰
Why do students fail in Algorithms and Data Structure Interviews for Top Companies? | by Shubham Gautam | Medium
5. [14 Patterns to Ace Any Coding Interview Question | Hacker Noon](#)⁶¹
14 Patterns to Ace Any Coding Interview Question | Hacker Noon

⁵⁷<https://en.wikibooks.org/wiki/Algorithms>

⁵⁸<https://www.geeksforgeeks.org/fundamentals-of-algorithms/>

⁵⁹<https://idea-instructions.com/>

⁶⁰<https://medium.com/@shubhamkumargautam/why-do-students-fail-in-algorithms-and-data-structure-interviews-for-top-companies-4fcca7ce7580>

⁶¹<https://hackernoon.com/14-patterns-to-ace-any-coding-interview-question-c5bb3357f6ed>

6. [Data Structures 101: Graphs — A Visual Introduction for Beginners | by Estefania Cassingena Navone | freeCodeCamp.org | Medium](#)⁶²
Data Structures 101: Graphs — A Visual Introduction for Beginners | by Estefania Cassingena Navone | freeCodeCamp.org | Medium
7. [Sorting Algorithms - LAMFO](#)⁶³
Sorting Algorithms - LAMFO
8. [Big-O Algorithm Complexity Cheat Sheet \(Know Thy Complexities!\) @ericdrowell](#)⁶⁴
Big-O Algorithm Complexity Cheat Sheet (Know Thy Complexities!) @ericdrowell
9. [Time Complexity of Algorithms— Big O Notation Explained In Plain English | by Yong Cui | The Startup | Medium](#)⁶⁵
Time Complexity of Algorithms— Big O Notation Explained In Plain English | by Yong Cui | The Startup | Medium

Videos

1. [MIT 6.006 Introduction to Algorithms, Fall 2011](#)⁶⁶

⁶²<https://medium.com/free-code-camp/data-structures-101-graphs-a-visual-introduction-for-beginners-6d88f36ec768>

⁶³<https://lamfo-unb.github.io/2019/04/21/Sorting-algorithms/>

⁶⁴<https://www.bigocheatsheet.com/>

⁶⁵<https://medium.com/swlh/time-complexity-of-algorithms-big-o-notation-explained-in-plain-english-e12a11dc4a4f>

⁶⁶https://www.youtube.com/playlist?list=PLU14u3cNGP61Oq3tWYp6V_F-5jb5L2iHb

Notes

(free space)

Notes

(free space)

Week 5: OOP



Time needed: 15 hours

Basics

1. [Objects Should Be Immutable](#)⁶⁷
Objects Should Be Immutable
2. [Getters/Setters. Evil. Period.](#)⁶⁸
Getters/Setters. Evil. Period.
3. [Seven Virtues of a Good Object](#)⁶⁹
Seven Virtues of a Good Object
4. [Why NULL is Bad?](#)⁷⁰
Why NULL is Bad?

Calisthenics

1. [Object Calisthenics](#)⁷¹
Object Calisthenics | William Durand

⁶⁷<https://www.yegor256.com/2014/06/09/objects-should-be-immutable.html>

⁶⁸<https://www.yegor256.com/2014/09/16/getters-and-setters-are-evil.html>

⁶⁹<https://www.yegor256.com/2014/11/20/seven-virtues-of-good-object.html>

⁷⁰<https://www.yegor256.com/2014/05/13/why-null-is-bad.html>

⁷¹<https://williamdurand.fr/2013/06/03/object-calisthenics/>

Clean Code

1. [Clean Code Cheat Sheet](#)⁷²

Cohesion & Coupling

1. [High Cohesion, Loose Coupling](#)⁷³
High Cohesion, Loose Coupling – A Sleek Geek Blog

SOLID

1. [SOLID, GRASP, and Other Basic Principles of Object-Oriented Design](#)⁷⁴
SOLID, GRASP, and Other Basic Principles of Object-Oriented Design - DZone Web Dev

Videos

1. [Joshua Thijssen: Paradoxes and theorems every developer should know](#) [Video @ DPC2017](#)⁷⁵ | [Slides](#)⁷⁶ 43:28

⁷²<https://www.bbv.ch/wp-content/uploads/2020/02/200-bbv-Software-Testing-Clean-Code-Cheat-Sheet.pdf>

⁷³<https://thebojan.ninja/2015/04/08/high-cohesion-loose-coupling/>

⁷⁴<https://dzone.com/articles/solid-grasp-and-other-basic-principles-of-object-o>

⁷⁵<https://www.youtube.com/watch?v=JBUIQnVfBQ>

⁷⁶<https://speakerdeck.com/jaytaph/paradoxes-and-theorems-every-developer-should-know-3>

Books

1. [□ Algorithms in a Nutshell⁷⁷](#)
Amazon.com: Algorithms in a Nutshell: A Practical Guide (9781491948927): Heineman, George T., Pollice, Gary, Selkow, Stanley: Books
2. [□ Code Review Patterns and Anti-Patterns⁷⁸](#)
Code Review Patterns and Anti-Patterns - Dzone Refcardz
3. [□ InfoQ eMag: Technical Debt and Software Craftsmanship⁷⁹](#)
InfoQ eMag: Technical Debt and Software Craftsmanship

⁷⁷<https://www.amazon.com/Algorithms-Nutshell-Desktop-Quick-Reference/dp/1491948922>

⁷⁸<https://dzone.com/refcardz/code-review-patterns-and-anti-patterns>

⁷⁹<https://www.infoq.com/minibooks/emag-technical-debt/>

Notes

(free space)

Notes

(free space)

Week 6: Design Patterns



Time needed: 15 hours

Creational

1. [Creational patterns](#)⁸⁰
Creational patterns
2. [Abstract Factory](#)⁸¹
Abstract Factory Design Pattern
3. [Builder](#)⁸²
Builder Design Pattern
4. [Factory Method](#)⁸³
Factory Method Design Pattern
5. [Object Pool](#)⁸⁴
Object Pool Design Pattern
6. [Prototype](#)⁸⁵
Prototype Design Pattern
7. [Singleton](#)⁸⁶
Singleton Design Pattern

⁸⁰https://sourcemaking.com/design_patterns/creational_patterns

⁸¹https://sourcemaking.com/design_patterns/abstract_factory

⁸²https://sourcemaking.com/design_patterns/builder

⁸³https://sourcemaking.com/design_patterns/factory_method

⁸⁴https://sourcemaking.com/design_patterns/object_pool

⁸⁵https://sourcemaking.com/design_patterns/prototype

⁸⁶https://sourcemaking.com/design_patterns/singleton

Structural

1. [Structural patterns](#)⁸⁷
Structural patterns
2. [Adapter](#)⁸⁸
Adapter Design Pattern
3. [Bridge](#)⁸⁹
Bridge Design Pattern
4. [Composite](#)⁹⁰
Composite Design Pattern
5. [Decorator](#)⁹¹
Decorator Design Pattern
6. [Facade](#)⁹²
Facade Design Pattern
7. [Flyweight](#)⁹³
Flyweight Design Pattern
8. [Private Class Data](#)⁹⁴
Private Class Data
9. [Proxy](#)⁹⁵
Proxy Design Pattern

Behavioral

1. [Behavioral patterns](#)⁹⁶
Behavioral patterns

⁸⁷https://sourcemaking.com/design_patterns/structural_patterns

⁸⁸https://sourcemaking.com/design_patterns/adapter

⁸⁹https://sourcemaking.com/design_patterns/bridge

⁹⁰https://sourcemaking.com/design_patterns/composite

⁹¹https://sourcemaking.com/design_patterns/decorator

⁹²https://sourcemaking.com/design_patterns/facade

⁹³https://sourcemaking.com/design_patterns/flyweight

⁹⁴https://sourcemaking.com/design_patterns/private_class_data

⁹⁵https://sourcemaking.com/design_patterns/proxy

⁹⁶https://sourcemaking.com/design_patterns/behavioral_patterns

2. [Chain of Responsibility](#)⁹⁷
Chain of Responsibility
3. [Command](#)⁹⁸
Command Design Pattern
4. [Interpreter](#)⁹⁹
Interpreter Design Pattern
5. [Iterator](#)¹⁰⁰
Iterator Design Pattern
6. [Mediator](#)¹⁰¹
Mediator Design Pattern
7. [Memento](#)¹⁰²
Memento Design Pattern
8. [Null Object](#)¹⁰³
Null Object Design Pattern
9. [Observer](#)¹⁰⁴
Observer Design Pattern
10. [State](#)¹⁰⁵
State Design Pattern
11. [Strategy](#)¹⁰⁶
Strategy Design Pattern
12. [Template Method](#)¹⁰⁷
Template Method Design Pattern
13. [Visitor](#)¹⁰⁸
Visitor Design Pattern

⁹⁷https://sourcemaking.com/design_patterns/chain_of_responsibility

⁹⁸https://sourcemaking.com/design_patterns/command

⁹⁹https://sourcemaking.com/design_patterns/interpreter

¹⁰⁰https://sourcemaking.com/design_patterns/iterator

¹⁰¹https://sourcemaking.com/design_patterns/mediator

¹⁰²https://sourcemaking.com/design_patterns/memento

¹⁰³https://sourcemaking.com/design_patterns/null_object

¹⁰⁴https://sourcemaking.com/design_patterns/observer

¹⁰⁵https://sourcemaking.com/design_patterns/state

¹⁰⁶https://sourcemaking.com/design_patterns/strategy

¹⁰⁷https://sourcemaking.com/design_patterns/template_method

¹⁰⁸https://sourcemaking.com/design_patterns/visitor

Books

1. [Clean Code](#)¹⁰⁹

¹⁰⁹<https://www.amazon.com/Clean-Code-Handbook-Software-Craftsmanship/dp/0132350882>

Notes

(free space)

Notes

(free space)

Week 7: Design Patterns



Time needed: 13 hours

UML

1. [A Comprehensive Guide to 14 Types of UML Diagram](#)¹¹⁰

Books

1. [Design Patterns](#)¹¹¹

¹¹⁰<https://warren2lynch.medium.com/a-comprehensive-guide-to-14-types-of-uml-diagram-affcc688377e>

¹¹¹<https://www.amazon.com/Design-Patterns-Elements-Reusable-Object-Oriented/dp/0201633612>

Notes

(free space)

Notes

(free space)

Week 8: Asynchronous Programming



Time needed: 2 hours

Basics

1. [General asynchronous programming concepts¹¹²](#)
General asynchronous programming concepts - Learn web development | MDN
2. [Async/Await - Best Practices in Asynchronous Programming | Microsoft Docs¹¹³](#)
Async/Await - Best Practices in Asynchronous Programming | Microsoft Docs
3. [When to Use \(and Not to Use\) Asynchronous Programming: 20 Pros Reveal the Best Use Cases - DZone DevOps¹¹⁴](#)
When to Use (and Not to Use) Asynchronous Programming: 20 Pros Reveal the Best Use Cases - DZone DevOps
4. [Asynchronous and Parallel Programming in C# .NET | by Thanh Le | Medium¹¹⁵](#)

¹¹²<https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Asynchronous/Concepts>

¹¹³<https://docs.microsoft.com/en-us/archive/msdn-magazine/2013/march/async-await-best-practices-in-asynchronous-programming>

¹¹⁴<https://dzone.com/articles/when-to-use-and-not-to-use-asynchronous-programmin>

¹¹⁵<https://medium.com/@letienthanh0212/asynchronous-and-parallel-programming-in-c-net-1e0f14e1db80>

Asynchronous and Parallel Programming in C# .NET | by
Thanh Le | Medium

1. [Reactive in practice: Concurrency, parallelism, asynchrony – IBM Developer](#)¹¹⁶

Reactive in practice: Concurrency, parallelism, asynchrony –
IBM Developer Close Favorite this Thumbs up Show more
icon Show more icon Show more icon Show more icon Show
more icon Show more icon Show more icon Show more icon
Show more icon Show more icon Show more icon Show more
icon Show more icon Share this on Facebook Share this on
Twitter Share this on LinkedIn Share this on WeChat Arrow
down Arrow up Close Modal

Messages

1. [Messaging patterns](#)¹¹⁷

Messaging patterns - Cloud Design Patterns | Microsoft Docs

¹¹⁶<https://developer.ibm.com/languages/java/tutorials/reactive-in-practice-4/>

¹¹⁷<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/messaging>

Notes

(free space)

Notes

(free space)

Week 9: Testing



Time needed: 14 hours

Arrange Act Assert

1. [3A – Arrange, Act, Assert¹¹⁸](#)
3A - Arrange, Act, Assert - XP123

Bottlenecks

1. [Big List Of 20 Common Bottlenecks¹¹⁹](#)
Big List of 20 Common Bottlenecks - High Scalability -

Mocks & Stubs

1. [Mocks Aren't Stubs¹²⁰](#)
Mocks Aren't Stubs
2. [TestDouble¹²¹](#)
TestDouble

¹¹⁸<https://xp123.com/articles/3a-arrange-act-assert/>

¹¹⁹<http://highscalability.com/blog/2012/5/16/big-list-of-20-common-bottlenecks.html>

¹²⁰<https://martinfowler.com/articles/mocksArentStubs.html>

¹²¹<https://martinfowler.com/bliki/TestDouble.html>

Mutation

1. [Mutation Testing in Software Testing: Mutant Score & Analysis Example¹²²](#)
Mutation Testing in Software Testing: Mutant Score & Analysis Example

Test Pyramid

1. [The Practical Test Pyramid¹²³](#)
The Practical Test Pyramid
2. [Types Of Software Testing: Different Testing Types With Details¹²⁴](#)
Types of Software Testing: Different Testing Types with Details

Test Driven Development

1. [Introduction to Test Driven Development \(TDD\)¹²⁵](#)
Introduction to Test Driven Development (TDD)
2. [test && commit || revert¹²⁶](#)
test && commit || revert. As part of Limbo on the Cheap, we...
| by Kent Beck | Medium

Mix

1. [Testing Microservices: an Overview of 12 Useful Techniques](#)

¹²²<https://www.guru99.com/mutation-testing.html>

¹²³<https://martinfowler.com/articles/practical-test-pyramid.html>

¹²⁴<https://www.softwaretestinghelp.com/types-of-software-testing/>

¹²⁵<http://agiledata.org/essays/tdd.html>

¹²⁶https://medium.com/@kentbeck_7670/test-commit-revert-870bbd756864

- Part 1¹²⁷

Testing Microservices: an Overview of 12 Useful Techniques
- Part 1

2. [Testing Microservices: Examining the Tradeoffs of Twelve Techniques - Part 2](#)¹²⁸

Testing Microservices: Examining the Tradeoffs of Twelve
Techniques - Part 2

Videos

1. Kevlin Henney: Enterprise Programming Tricks For Clean Code [Video](#)¹²⁹ | [Slides](#)¹³⁰
2. Sander Hoogendoorn: How Thinking Small is Changing Software Development Big Time [Video @ GOTO 2019](#)¹³¹ | [Slides](#)¹³²
3. [Mixed Paradigms: The Method to Madness. Venkat Subramaniam, Agile developer, Inc](#)¹³³
Mixed Paradigms: The Method to Madness. Venkat Subramaniam, Agile developer, Inc - YouTube

Books

1. [Test Driven Development: By Example](#)¹³⁴

¹²⁷<https://www.infoq.com/articles/twelve-testing-techniques-microservices-intro/>

¹²⁸<https://www.infoq.com/articles/twelve-testing-techniques-microservices-tradeoffs/>

¹²⁹<https://www.youtube.com/watch?v=dC9vdQkU-xI>

¹³⁰<https://www.slideshare.net/Kevlin/clean-coders-hate-what-happens-to-your-code-when-you-use-these-enterprise-programming-tricks-77305014>

¹³¹<https://www.youtube.com/watch?v=YCQMIF9QXM>

¹³²<https://www.slideshare.net/aahoogendoorn/its-a-small-world-after-all-how-thinking-small-changes-software-big-time>

¹³³<https://www.youtube.com/watch?v=QYBRifsWHD0>

¹³⁴<https://www.amazon.com/Test-Driven-Development-By-Example/dp/0321146530>

Notes

(free space)

Notes

(free space)

Week 10: Refactoring



Time needed: 16 hours

Articles

1. [□ The Art of Enbugging¹³⁵](#)
2. [□ 9 Anti-Patterns Every Programmer Should Be Aware Of¹³⁶](#)
9 Anti-Patterns Every Programmer Should Be Aware Of
3. [□ Provable Refactorings¹³⁷](#)
GitHub - digdeeproots/provable-refactorings: A collection of refactoring recipes that are provably safe. They never accidentally introduce nor fix a bug, including one that you don't know exists. They maintain all behavior, including unknown or unspecified behavior. To accomplish this, each recipe is concrete and language-specific.
4. [□ The Most Dangerous Word In Software Development¹³⁸](#)
The Most Dangerous Word In Software Development – A List Apart
5. [□ Understanding Fake Agile¹³⁹](#)
Understanding Fake Agile

¹³⁵https://www2.ccs.neu.edu/research/demeter/related-work/pragmatic-programmer/jan_03_enbug.pdf

¹³⁶<https://sahandsaba.com/nine-anti-patterns-every-programmer-should-be-aware-of-with-examples.html>

¹³⁷<https://github.com/digdeeproots/provable-refactorings>

¹³⁸<https://alistapart.com/blog/post/the-most-dangerous-word-in-software-development/>

¹³⁹<https://www.forbes.com/sites/stevedenning/2019/05/23/understanding-fake-agile>

6. [Why Do Managers Hate Agile?](#)¹⁴⁰
Why Do Managers Hate Agile?

Code Smells

Bloaters

1. [Long Method](#)¹⁴¹
Long Method
2. [Large Class](#)¹⁴²
Large Class
3. [Primitive Obsession](#)¹⁴³
Primitive Obsession
4. [Long Parameter List](#)¹⁴⁴
Long Parameter List
5. [Data Clumps](#)¹⁴⁵
Data Clumps

Object-Orientation Abusers

1. [Switch Statements](#)¹⁴⁶
Switch Statements
2. [Temporary Field](#)¹⁴⁷
Temporary Field
3. [Refused Bequest](#)¹⁴⁸
Refused Bequest

¹⁴⁰<https://www.forbes.com/sites/stevedenning/2015/01/26/why-do-managers-hate-agile/>

¹⁴¹<https://sourcemaking.com/refactoring/smells/long-method>

¹⁴²<https://sourcemaking.com/refactoring/smells/large-class>

¹⁴³<https://sourcemaking.com/refactoring/smells/primitive-obsession>

¹⁴⁴<https://sourcemaking.com/refactoring/smells/long-parameter-list>

¹⁴⁵<https://sourcemaking.com/refactoring/smells/data-clumps>

¹⁴⁶<https://sourcemaking.com/refactoring/smells/switch-statements>

¹⁴⁷<https://sourcemaking.com/refactoring/smells/temporary-field>

¹⁴⁸<https://sourcemaking.com/refactoring/smells/refused-bequest>

4. [Alternative Classes with Different Interfaces](#)¹⁴⁹
Alternative Classes with Different Interfaces

Change Preventers

1. [Divergent Change](#)¹⁵⁰
Divergent Change
2. [Shotgun Surgery](#)¹⁵¹
Shotgun Surgery
3. [Parallel Inheritance Hierarchies](#)¹⁵²
Parallel Inheritance Hierarchies

Dispensables

1. [Comments](#)¹⁵³
Comments
2. [Duplicate Code](#)¹⁵⁴
Duplicate Code
3. [Lazy Class](#)¹⁵⁵
Lazy Class
4. [Data Class](#)¹⁵⁶
Data Class
5. [Dead Code](#)¹⁵⁷
Dead Code
6. [Speculative Generality](#)¹⁵⁸
Speculative Generality

¹⁴⁹<https://sourcemaking.com/refactoring/smells/alternative-classes-with-different-interfaces>

¹⁵⁰<https://sourcemaking.com/refactoring/smells/divergent-change>

¹⁵¹<https://sourcemaking.com/refactoring/smells/shotgun-surgery>

¹⁵²<https://sourcemaking.com/refactoring/smells/parallel-inheritance-hierarchies>

¹⁵³<https://sourcemaking.com/refactoring/smells/comments>

¹⁵⁴<https://sourcemaking.com/refactoring/smells/duplicate-code>

¹⁵⁵<https://sourcemaking.com/refactoring/smells/lazy-class>

¹⁵⁶<https://sourcemaking.com/refactoring/smells/data-class>

¹⁵⁷<https://sourcemaking.com/refactoring/smells/dead-code>

¹⁵⁸<https://sourcemaking.com/refactoring/smells/speculative-generality>

Couplers

1. [Feature Envy](#)¹⁵⁹
Feature Envy
2. [Inappropriate Intimacy](#)¹⁶⁰
Inappropriate Intimacy
3. [Message Chains](#)¹⁶¹
Message Chains
4. [Middle Man](#)¹⁶²
Middle Man

Other Smells

1. [Incomplete Library Class](#)¹⁶³
Incomplete Library Class

Refactoring Techniques

Composing Methods

1. [Extract Method](#)¹⁶⁴
Extract Method
2. [Inline Method](#)¹⁶⁵
Inline Method
3. [Extract Variable](#)¹⁶⁶
Extract Variable

¹⁵⁹<https://sourcemaking.com/refactoring/smells/feature-envy>

¹⁶⁰<https://sourcemaking.com/refactoring/smells/inappropriate-intimacy>

¹⁶¹<https://sourcemaking.com/refactoring/smells/message-chains>

¹⁶²<https://sourcemaking.com/refactoring/smells/middle-man>

¹⁶³<https://sourcemaking.com/refactoring/smells/incomplete-library-class>

¹⁶⁴<https://sourcemaking.com/refactoring/extract-method>

¹⁶⁵<https://sourcemaking.com/refactoring/inline-method>

¹⁶⁶<https://sourcemaking.com/refactoring/extract-variable>

4. [Inline Temp](#)¹⁶⁷
Inline Temp
5. [Replace Temp with Query](#)¹⁶⁸
Replace Temp with Query
6. [Split Temporary Variable](#)¹⁶⁹
Split Temporary Variable
7. [Remove Assignments to Parameters](#)¹⁷⁰
Remove Assignments to Parameters
8. [Replace Method with Method Object](#)¹⁷¹
Replace Method with Method Object
9. [Substitute Algorithm](#)¹⁷²
Substitute Algorithm

Moving Features between Objects

1. [Move Method](#)¹⁷³
Move Method
2. [Move Field](#)¹⁷⁴
Move Field
3. [Extract Class](#)¹⁷⁵
Extract Class
4. [Inline Class](#)¹⁷⁶
Inline Class
5. [Hide Delegate](#)¹⁷⁷
Hide Delegate

¹⁶⁷<https://sourcemaking.com/refactoring/inline-temp>

¹⁶⁸<https://sourcemaking.com/refactoring/replace-temp-with-query>

¹⁶⁹<https://sourcemaking.com/refactoring/split-temporary-variable>

¹⁷⁰<https://sourcemaking.com/refactoring/remove-assignments-to-parameters>

¹⁷¹<https://sourcemaking.com/refactoring/replace-method-with-method-object>

¹⁷²<https://sourcemaking.com/refactoring/substitute-algorithm>

¹⁷³<https://sourcemaking.com/refactoring/move-method>

¹⁷⁴<https://sourcemaking.com/refactoring/move-field>

¹⁷⁵<https://sourcemaking.com/refactoring/extract-class>

¹⁷⁶<https://sourcemaking.com/refactoring/inline-class>

¹⁷⁷<https://sourcemaking.com/refactoring/hide-delegate>

6. [Remove Middle Man](#)¹⁷⁸
Remove Middle Man
7. [Introduce Foreign Method](#)¹⁷⁹
Introduce Foreign Method
8. [Introduce Local Extension](#)¹⁸⁰
Introduce Local Extension

Organizing Data

1. [Self Encapsulate Field](#)¹⁸¹
Self Encapsulate Field
2. [Replace Data Value with Object](#)¹⁸²
Replace Data Value with Object
3. [Change Value to Reference](#)¹⁸³
Change Value to Reference
4. [Change Reference to Value](#)¹⁸⁴
Change Reference to Value
5. [Replace Array with Object](#)¹⁸⁵
Replace Array with Object
6. [Duplicate Observed Data](#)¹⁸⁶
Duplicate Observed Data
7. [Change Unidirectional Association to Bidirectional](#)¹⁸⁷
Change Unidirectional Association to Bidirectional
8. [Change Bidirectional Association to Unidirectional](#)¹⁸⁸
Change Bidirectional Association to Unidirectional

¹⁷⁸<https://sourcemaking.com/refactoring/remove-middle-man>

¹⁷⁹<https://sourcemaking.com/refactoring/introduce-foreign-method>

¹⁸⁰<https://sourcemaking.com/refactoring/introduce-local-extension>

¹⁸¹<https://sourcemaking.com/refactoring/self-encapsulate-field>

¹⁸²<https://sourcemaking.com/refactoring/replace-data-value-with-object>

¹⁸³<https://sourcemaking.com/refactoring/change-value-to-reference>

¹⁸⁴<https://sourcemaking.com/refactoring/change-reference-to-value>

¹⁸⁵<https://sourcemaking.com/refactoring/replace-array-with-object>

¹⁸⁶<https://sourcemaking.com/refactoring/duplicate-observed-data>

¹⁸⁷<https://sourcemaking.com/refactoring/change-unidirectional-association-to-bidirectional>

¹⁸⁸<https://sourcemaking.com/refactoring/change-bidirectional-association-to-unidirectional>

9. [Replace Magic Number with Symbolic Constant](#)¹⁸⁹
Replace Magic Number with Symbolic Constant
10. [Encapsulate Field](#)¹⁹⁰
Encapsulate Field
11. [Encapsulate Collection](#)¹⁹¹
Encapsulate Collection
12. [Replace Type Code with Class](#)¹⁹²
Replace Type Code with Class
13. [Replace Type Code with Subclasses](#)¹⁹³
Replace Type Code with Subclasses
14. [Replace Type Code with State/Strategy](#)¹⁹⁴
Replace Type Code with State/Strategy
15. [Replace Subclass with Fields](#)¹⁹⁵
Replace Subclass with Fields

Simplifying Conditional Expressions

1. [Decompose Conditional](#)¹⁹⁶
Decompose Conditional
2. [Consolidate Conditional Expression](#)¹⁹⁷
Consolidate Conditional Expression
3. [Consolidate Duplicate Conditional Fragments](#)¹⁹⁸
Consolidate Duplicate Conditional Fragments
4. [Remove Control Flag](#)¹⁹⁹
Remove Control Flag

¹⁸⁹<https://sourcemaking.com/refactoring/replace-magic-number-with-symbolic-constant>

¹⁹⁰<https://sourcemaking.com/refactoring/encapsulate-field>

¹⁹¹<https://sourcemaking.com/refactoring/encapsulate-collection>

¹⁹²<https://sourcemaking.com/refactoring/replace-type-code-with-class>

¹⁹³<https://sourcemaking.com/refactoring/replace-type-code-with-subclasses>

¹⁹⁴<https://sourcemaking.com/refactoring/replace-type-code-with-state-strategy>

¹⁹⁵<https://sourcemaking.com/refactoring/replace-subclass-with-fields>

¹⁹⁶<https://sourcemaking.com/refactoring/decompose-conditional>

¹⁹⁷<https://sourcemaking.com/refactoring/consolidate-conditional-expression>

¹⁹⁸<https://sourcemaking.com/refactoring/consolidate-duplicate-conditional-fragments>

¹⁹⁹<https://sourcemaking.com/refactoring/remove-control-flag>

5. [Replace Nested Conditional with Guard Clauses](#)²⁰⁰
Replace Nested Conditional with Guard Clauses
6. [Replace Conditional with Polymorphism](#)²⁰¹
Replace Conditional with Polymorphism
7. [Introduce Null Object](#)²⁰²
Introduce Null Object
8. [Introduce Assertion](#)²⁰³
Introduce Assertion

Simplifying Method Calls

1. [Rename Method](#)²⁰⁴
Rename Method
2. [Add Parameter](#)²⁰⁵
Add Parameter
3. [Remove Parameter](#)²⁰⁶
Remove Parameter
4. [Separate Query from Modifier](#)²⁰⁷
Separate Query from Modifier
5. [Parameterize Method](#)²⁰⁸
Parameterize Method
6. [Replace Parameter with Explicit Methods](#)²⁰⁹
Replace Parameter with Explicit Methods
7. [Preserve Whole Object](#)²¹⁰
Preserve Whole Object

²⁰⁰<https://sourcemaking.com/refactoring/replace-nested-conditional-with-guard-clauses>

²⁰¹<https://sourcemaking.com/refactoring/replace-conditional-with-polymorphism>

²⁰²<https://sourcemaking.com/refactoring/introduce-null-object>

²⁰³<https://sourcemaking.com/refactoring/introduce-assertion>

²⁰⁴<https://sourcemaking.com/refactoring/rename-method>

²⁰⁵<https://sourcemaking.com/refactoring/add-parameter>

²⁰⁶<https://sourcemaking.com/refactoring/remove-parameter>

²⁰⁷<https://sourcemaking.com/refactoring/separate-query-from-modifier>

²⁰⁸<https://sourcemaking.com/refactoring/parameterize-method>

²⁰⁹<https://sourcemaking.com/refactoring/replace-parameter-with-explicit-methods>

²¹⁰<https://sourcemaking.com/refactoring/preserve-whole-object>

8. [Replace Parameter with Method Call](#)²¹¹
Replace Parameter with Method Call
9. [Introduce Parameter Object](#)²¹²
Introduce Parameter Object
10. [Remove Setting Method](#)²¹³
Remove Setting Method
11. [Hide Method](#)²¹⁴
Hide Method
12. [Replace Constructor with Factory Method](#)²¹⁵
Replace Constructor with Factory Method
13. [Replace Error Code with Exception](#)²¹⁶
Replace Error Code with Exception
14. [Replace Exception with Test](#)²¹⁷
Replace Exception with Test

Dealing with Generalisation

1. [Pull Up Field](#)²¹⁸
Pull Up Field
2. [Pull Up Method](#)²¹⁹
Pull Up Method
3. [Pull Up Constructor Body](#)²²⁰
Pull Up Constructor Body
4. [Push Down Method](#)²²¹
Push Down Method

²¹¹<https://sourcemaking.com/refactoring/replace-parameter-with-method-call>

²¹²<https://sourcemaking.com/refactoring/introduce-parameter-object>

²¹³<https://sourcemaking.com/refactoring/remove-setting-method>

²¹⁴<https://sourcemaking.com/refactoring/hide-method>

²¹⁵<https://sourcemaking.com/refactoring/replace-constructor-with-factory-method>

²¹⁶<https://sourcemaking.com/refactoring/replace-error-code-with-exception>

²¹⁷<https://sourcemaking.com/refactoring/replace-exception-with-test>

²¹⁸<https://sourcemaking.com/refactoring/pull-up-field>

²¹⁹<https://sourcemaking.com/refactoring/pull-up-method>

²²⁰<https://sourcemaking.com/refactoring/pull-up-constructor-body>

²²¹<https://sourcemaking.com/refactoring/push-down-method>

5. [Push Down Field](#)²²²
Push Down Field
 6. [Extract Subclass](#)²²³
Extract Subclass
 7. [Extract Superclass](#)²²⁴
Extract Superclass
 8. [Extract Interface](#)²²⁵
Extract Interface
 9. [Collapse Hierarchy](#)²²⁶
Collapse Hierarchy
 10. [Form Template Method](#)²²⁷
Form Template Method
 11. [Replace Inheritance with Delegation](#)²²⁸
Replace Inheritance with Delegation
 12. [Replace Delegation with Inheritance](#)²²⁹
Replace Delegation with Inheritance
-

Books

1. [Refactoring to Patterns](#)²³⁰
Refactoring to Patterns: Kerievsky, Joshua: 0785342213355:
Amazon.com: Books

²²²<https://sourcemaking.com/refactoring/push-down-field>

²²³<https://sourcemaking.com/refactoring/extract-subclass>

²²⁴<https://sourcemaking.com/refactoring/extract-superclass>

²²⁵<https://sourcemaking.com/refactoring/extract-interface>

²²⁶<https://sourcemaking.com/refactoring/collapse-hierarchy>

²²⁷<https://sourcemaking.com/refactoring/form-template-method>

²²⁸<https://sourcemaking.com/refactoring/replace-inheritance-with-delegation>

²²⁹<https://sourcemaking.com/refactoring/replace-delegation-with-inheritance>

²³⁰<https://www.amazon.com/Refactoring-Patterns-Joshua-Kerievsky/dp/0321213351>

Notes

(free space)

Notes

(free space)

Week 11: Refactoring



Time needed: 14 hours

Software Development AntiPatterns

1. [The Blob](#)²³¹
The Blob
2. [Continuous Obsolescence](#)²³²
Continuous Obsolescence
3. [Lava Flow](#)²³³
Lava Flow
4. [Ambiguous Viewpoint](#)²³⁴
Ambiguous Viewpoint
5. [Functional Decomposition](#)²³⁵
Functional Decomposition
6. [Poltergeists](#)²³⁶
Poltergeists
7. [Boat Anchor](#)²³⁷
Boat Anchor

²³¹<https://sourcemaking.com/antipatterns/the-blob>

²³²<https://sourcemaking.com/antipatterns/continuous-obsolence>

²³³<https://sourcemaking.com/antipatterns/lava-flow>

²³⁴<https://sourcemaking.com/antipatterns/ambiguous-viewpoint>

²³⁵<https://sourcemaking.com/antipatterns/functional-decomposition>

²³⁶<https://sourcemaking.com/antipatterns/poltergeists>

²³⁷<https://sourcemaking.com/antipatterns/boat-anchor>

8. [Golden Hammer](#)²³⁸
Golden Hammer
9. [Dead End](#)²³⁹
Dead End
10. [Spaghetti Code](#)²⁴⁰
Spaghetti Code
11. [Input Kludge](#)²⁴¹
Input Kludge
12. [Walking through a Minefield](#)²⁴²
Walking through a Minefield
13. [Cut-And-Paste Programming](#)²⁴³
Cut-And-Paste Programming
14. [Mushroom Management](#)²⁴⁴
Mushroom Management

Software Architecture AntiPatterns

1. [Autogenerated Stovepipe](#)²⁴⁵
Autogenerated Stovepipe
2. [Stovepipe Enterprise](#)²⁴⁶
Stovepipe Enterprise
3. [Jumble](#)²⁴⁷
Jumble
4. [Stovepipe System](#)²⁴⁸
Stovepipe System

²³⁸<https://sourcemaking.com/antipatterns/golden-hammer>

²³⁹<https://sourcemaking.com/antipatterns/dead-end>

²⁴⁰<https://sourcemaking.com/antipatterns/spaghetti-code>

²⁴¹<https://sourcemaking.com/antipatterns/input-kludge>

²⁴²<https://sourcemaking.com/antipatterns/walking-through-minefield>

²⁴³<https://sourcemaking.com/antipatterns/cut-and-paste-programming>

²⁴⁴<https://sourcemaking.com/antipatterns/mushroom-management>

²⁴⁵<https://sourcemaking.com/antipatterns/autogenerated-stovepipe>

²⁴⁶<https://sourcemaking.com/antipatterns/stovepipe-enterprise>

²⁴⁷<https://sourcemaking.com/antipatterns/jumble>

²⁴⁸<https://sourcemaking.com/antipatterns/stovepipe-system>

5. [Cover Your Assets](#)²⁴⁹
Cover Your Assets
6. [Vendor Lock-In](#)²⁵⁰
Vendor Lock-In
7. [Wolf Ticket](#)²⁵¹
Wolf Ticket
8. [Architecture By Implication](#)²⁵²
Architecture By Implication
9. [Warm Bodies](#)²⁵³
Warm Bodies
10. [Design By Committee](#)²⁵⁴
Design By Committee
11. [Swiss Army Knife](#)²⁵⁵
Swiss Army Knife
12. [Reinvent The Wheel](#)²⁵⁶
Reinvent The Wheel
13. [The Grand Old Duke of York](#)²⁵⁷
The Grand Old Duke of York

Project Management AntiPatterns

1. [Blowhard Jamboree](#)²⁵⁸
Blowhard Jamboree
2. [Analysis Paralysis](#)²⁵⁹
Analysis Paralysis

²⁴⁹<https://sourcemaking.com/antipatterns/cover-your-assets>

²⁵⁰<https://sourcemaking.com/antipatterns/vendor-lock-in>

²⁵¹<https://sourcemaking.com/antipatterns/wolf-ticket>

²⁵²<https://sourcemaking.com/antipatterns/architecture-by-implication>

²⁵³<https://sourcemaking.com/antipatterns/warm-bodies>

²⁵⁴<https://sourcemaking.com/antipatterns/design-by-committee>

²⁵⁵<https://sourcemaking.com/antipatterns/swiss-army-knife>

²⁵⁶<https://sourcemaking.com/antipatterns/reinvent-the-wheel>

²⁵⁷<https://sourcemaking.com/antipatterns/the-grand-old-duke-of-york>

²⁵⁸<https://sourcemaking.com/antipatterns/blowhard-jamboree>

²⁵⁹<https://sourcemaking.com/antipatterns/analysis-paralysis>

3. [Viewgraph Engineering](#)²⁶⁰
Viewgraph Engineering
4. [Death By Planning](#)²⁶¹
Death By Planning
5. [Fear of Success](#)²⁶²
Fear of Success
6. [Corncob](#)²⁶³
Corncob
7. [Intellectual Violence](#)²⁶⁴
Intellectual Violence
8. [Irrational Management](#)²⁶⁵
Irrational Management
9. [Smoke and Mirrors](#)²⁶⁶
Smoke and Mirrors
10. [Project Mismanagement](#)²⁶⁷
Project Mismanagement
11. [Throw It over the Wall](#)²⁶⁸
Throw It over the Wall
12. [Fire Drill](#)²⁶⁹
Fire Drill
13. [The Feud](#)²⁷⁰
The Feud
14. [E-mail Is Dangerous](#)²⁷¹
E-mail Is Dangerous

²⁶⁰<https://sourcemaking.com/antipatterns/viewgraph-engineering>

²⁶¹<https://sourcemaking.com/antipatterns/death-by-planning>

²⁶²<https://sourcemaking.com/antipatterns/fear-of-success>

²⁶³<https://sourcemaking.com/antipatterns/corncob>

²⁶⁴<https://sourcemaking.com/antipatterns/intellectual-violence>

²⁶⁵<https://sourcemaking.com/antipatterns/irrational-management>

²⁶⁶<https://sourcemaking.com/antipatterns/smoke-and-mirrors>

²⁶⁷<https://sourcemaking.com/antipatterns/project-mismanagement>

²⁶⁸<https://sourcemaking.com/antipatterns/throw-it-over-the-wall>

²⁶⁹<https://sourcemaking.com/antipatterns/fire-drill>

²⁷⁰<https://sourcemaking.com/antipatterns/the-feud>

²⁷¹<https://sourcemaking.com/antipatterns/e-mail-is-dangerous>

Books

1. [Refactoring](#)²⁷²

²⁷²<https://www.amazon.com/Refactoring-Improving-Design-Existing-Code/dp/0134757599>

Notes

(free space)

Notes

(free space)

Week 12: DB



Time needed: 8 hours

ACID properties

1. [A Primer on ACID Transactions: The Basics Every Cloud App Developer Must Know](#)²⁷³
A Primer on ACID Transactions: The Basics Every Cloud App Developer Must Know - The Distributed SQL Blog
hamburger-white

Consistency

1. [Eventual Consistency vs Strong Consistency | by Vivek Kumar Singh | System Design Blog | Medium](#)²⁷⁴
Eventual Consistency vs Strong Consistency | by Vivek Kumar Singh | System Design Blog | Medium

²⁷³<https://blog.yugabyte.com/a-primer-on-acid-transactions/>

²⁷⁴<https://medium.com/system-design-blog/eventual-consistency-vs-strong-consistency-b4de1f92534d>

Joins

1. [A Visual Explanation of SQL Joins](#)²⁷⁵
A Visual Explanation of SQL Joins

Normalisation

1. [Normalization of Database](#)²⁷⁶
1NF, 2NF, 3NF and BCNF in Database Normalization | Study-tonight

Sharding

1. [Five sharding data models and which is right](#)²⁷⁷
Five sharding data models and which is right
2. [Four Data Sharding Strategies We Analyzed in Building a Distributed SQL Database](#)²⁷⁸
Four Data Sharding Strategies We Analyzed in Building a Distributed SQL Database - The Distributed SQL Blog ham-burger-white
3. [Understanding Database Sharding](#)²⁷⁹
Understanding Database Sharding | DigitalOcean DigitalOcean home DigitalOcean Homepage

²⁷⁵<https://blog.codinghorror.com/a-visual-explanation-of-sql-joins/>

²⁷⁶<https://www.studytonight.com/dbms/database-normalization.php>

²⁷⁷<https://www.citusdata.com/blog/2017/08/28/five-data-models-for-sharding/>

²⁷⁸<https://blog.yugabyte.com/four-data-sharding-strategies-we-analyzed-in-building-a-distributed-sql-database/>

²⁷⁹<https://www.digitalocean.com/community/tutorials/understanding-database-sharding>

Mix

1. [Understanding Window Functions](#)²⁸⁰
Understanding Window Functions
 2. [Why Order By With Limit and Offset is Slow - Faster Pagination in Mysql](#)²⁸¹
Why Order By With Limit and Offset is Slow - Faster Pagination in Mysql
 3. [Managing Hierarchical Data in MySQL Using the Adjacency List Model](#)²⁸²
Managing Hierarchical Data in MySQL Using the Adjacency List Model
 4. [101 Tips to MySQL Tuning and Optimization](#)²⁸³
101 Tips to MySQL Tuning and Optimization
 5. [A Review of Graph Databases](#)²⁸⁴
A Review of Graph Databases
-

Books

1. [97 Things Every SQL Developer Should Know](#)²⁸⁵
97 Things Every SQL Developer Should Know: Beaulieu, Alan: 9780596804336: Amazon.com: Books

²⁸⁰<https://tapoueh.org/blog/2013/08/understanding-window-functions/>

²⁸¹<https://www.eversql.com/faster-pagination-in-mysql-why-order-by-with-limit-and-offset-is-slow/>

²⁸²<https://www.mysqltutorial.org/mysql-adjacency-list-tree/>

²⁸³<https://www.monitis.com/blog/101-tips-to-mysql-tuning-and-optimization/>

²⁸⁴<https://nebula-graph.io/posts/review-on-graph-databases/>

²⁸⁵<https://www.amazon.com/Things-Every-Developer-Should-Know/dp/0596804334>

Notes

(free space)

Notes

(free space)

Week 13: Extra Resources



Time needed: 6 hours

RegEx

1. [Debuggex: Online visual regex tester. JavaScript, Python, and PCRE.](#)²⁸⁶
Debuggex: Online visual regex tester. JavaScript, Python, and PCRE.
2. [RegExr: Learn, Build, & Test RegEx](#)²⁸⁷

²⁸⁶<https://www.debuggex.com/>

²⁸⁷<https://regex.com/>

Notes

(free space)

Notes

(free space)

VCS

1. [Learn Git Branching](#)²⁸⁸
Learn Git Branching
 2. [Git Command Explorer](#)²⁸⁹
Git Explorer
 3. [GitHub Cheat Sheet](#)²⁹⁰
GitHub - tiimgreen/github-cheat-sheet: A list of cool features of Git and GitHub.
-

²⁸⁸<https://learngitbranching.js.org/>

²⁸⁹<https://gitexplorer.com/>

²⁹⁰<https://github.com/tiimgreen/github-cheat-sheet>

Notes

(free space)

Notes

(free space)

SEO

1. [On-Page SEO Checklist for 2020](#)²⁹¹
The On-Page SEO Checklist for 2021
2. [UI Testing Best Practices](#)²⁹²
GitHub - NoriSte/ui-testing-best-practices: The largest UI testing best practices list (last update: January 2021)
3. [A Breakdown of HTML Usage Across ~8 Million Pages \(& What It Means for Modern SEO\)](#)²⁹³
A Breakdown of HTML Usage Across ~8 Million Pages (& What It Means for Modern SEO) - Moz Moz Search Resources
Menu icon-close Search Moz
4. [34 Ways To Improve SEO Rankings in 2020](#)²⁹⁴
34 Ways To Improve SEO Rankings in 2021
5. [The Complete 51-Point SEO Checklist For 2020](#)²⁹⁵
The Complete 51-Point SEO Checklist For 2021 [Updated]
6. [The Complete SEO Checklist For 2020](#)²⁹⁶
The Complete SEO Checklist For 2021
7. [The SEMrush Website Migration Checklist](#)²⁹⁷
Website Migration Checklist: Everything You Need to Know
8. [The Ultimate SEO Checklist for 2020](#)²⁹⁸
The Ultimate SEO Checklist for 2021 (66 Checks + PDF Download)
9. [I Used The Web For A Day On A 50 MB Budget](#)²⁹⁹
I Used The Web For A Day On A 50 MB Budget — Smashing Magazine
Clear Search Back to top

²⁹¹<https://www.gotchseo.com/on-page-seo/>

²⁹²<https://github.com/NoriSte/ui-testing-best-practices>

²⁹³<https://moz.com/blog/a-breakdown-of-html-usage-across-8-million-pages>

²⁹⁴<https://www.quicksprout.com/ways-to-improve-seo-ranking/>

²⁹⁵<https://www.clickminded.com/seo-checklist/>

²⁹⁶<https://backlinko.com/seo-checklist>

²⁹⁷<https://www.semrush.com/blog/website-migration-checklist/>

²⁹⁸<https://www.reliablesoft.net/seo-checklist/>

²⁹⁹<https://www.smashingmagazine.com/2019/07/web-on-50mb-budget/>

10. [Building the most inaccessible site possible with a perfect Lighthouse score³⁰⁰](#)
Building the most inaccessible site possible with a perfect Lighthouse score - Manuel Matuzović
11. [The On-Page SEO Cheat Sheet³⁰¹](#)
The On-Page SEO Cheat Sheet
12. [How to Create a Site Structure That Will Enhance SEO³⁰²](#)
How to Create a Site Structure That Will Enhance SEO

³⁰⁰<https://www.matuzo.at/blog/building-the-most-inaccessible-site-possible-with-a-perfect-lighthouse-score/>

³⁰¹<https://neilpatel.com/2015/07/07/the-on-page-seo-cheat-sheet/>

³⁰²<https://neilpatel.com/blog/site-structure-enhance-seo/>

Notes

(free space)

Notes

(free space)

Books

1. □ [Git Patterns and Anti-Patterns Scaling from Workgroup to Enterprise](#)³⁰³
Git Patterns and Anti-Patterns - Dzone Refcardz
2. □ [Regular Expressions A Look at Characters, Types, Operators, and More](#)³⁰⁴
Regular Expressions - Dzone Refcardz

³⁰³<https://dzone.com/refcardz/git-patterns-and-anti-patterns>

³⁰⁴<https://dzone.com/refcardz/regular-expressions>

Level: Intermediate

Level Definition

We are in the middle ground of the model. An individual falls into this category when he is fully capable of troubleshooting and solving problems on their own, as well as planning their future actions while avoiding previous mistakes. The practitioner will still experience trouble when it comes to pinpointing the exact details to focus on. The IT sphere works mainly in teams in order to smoothen out these processes.

- <https://www.360pmo.com/the-five-dreyfus-model-stages/>



Duration: 11 weeks (~3 months)
Average per week: ~15 hours

Schedule

- Week 14: Build Up Dictionary
- Week 15: Caching
- Week 16: DDD, Functional & CQRS/ES

- Week 17: DDD, Functional & CQRS/ES
- Week 18: Networking
- Week 19: Cloud
- Week 20: DevOps
- Week 21: Security
- Week 22: Architecture
- Week 23: Architecture
- Week 24: Jobs

Week 14: Build Up Dictionary

A

APM - Application Performance Management

APMs are solutions to monitor applications to ensure performance and availability by generally collecting the application and server metrics to alert you when crossing thresholds.

B

Backend For Frontend

A Backend for Frontend pattern is tightly coupled to a specific user experience and helps to create backends for client-facing mobile or web apps.

Blue-Green Deployment

Blue-green deployment is a technique that reduces risk and downtime by running two environments and redirecting traffic to the latest one.

Brook's Law

Adding manpower to a late software project makes it later.

Burnout

Burnout is a state of mental exhaustion caused by excessive and prolonged stress.

C

Canary

Canary release is a technique used for rolling out new changes to a small subset of users before making it available to everyone, generally used to control and minimise the impact.

CAP Theorem

The CAP theorem states that a distributed system can deliver only two of three desired characteristics at the same moment: Consistency, Availability, Partition tolerance.

CI/CD - Continuous Integration / Continuous Deployment

A CI/CD pipeline automates your software delivery process, by building code, running tests and deploying the application.

CIDR - Classless Inter-Domain Routing

A CIDR is a set of IP standards used for creating unique identifiers for network devices.

Cohesion

Cohesion refers to what a module can do: Low would mean that it is too broad, High means that is very focused.

Contravariance

A property is contravariant if it reverses the ordering of types, which orders types from more generic to more specific.

Coupling

Coupling refers to how dependent two modules towards each other: Low would mean that changing something major should not affect the other module, High means that is difficult to change without side-effects.

Covariance

A property is covariant if it preserves the ordering of types, which orders types from more specific to more generic.

CQRS - Command Query Responsibility Segregation

CQRS it is a pattern used for splitting the read model (query) from the write model (command).

CSP - Content Security Policy

Content Security Policy an extra layer of security implemented in the browser that helps to detect and mitigate certain types of attacks (like XSS).

D

Data Lake

A data lake is a centralized repository that allows storing structured/unstructured data at scale.

Data Mesh

Data Mesh is a layer that abstracts the complexities of connecting, managing and supporting access to data and allows the data to be available and discoverable for the applications that needed access to it.

Data Warehouse

A data warehouse is a central repository of information that can be analyzed and aggregated.

DaaS - Data as a Service

DaaS can be considered as a subset of SaaS as it builds on the concept that its data product can be provided to the user on demand.

DBaaS - Database as a Service

DBaaS is a cloud service that lets users access and uses a cloud database system without having to manage it directly.

DevOps

DevOps is a mix of cultural practices and tools that increases an organization's ability to deliver applications and services.

DDD - Domain Driven Design

Domain-Driven Design is an approach to complex software where there's a focus on the core domain, via collaboration (and by speaking a ubiquitous language) with domain practitioners there will be a clear understanding of the data models and context in which they're applied.

E

Emergent Design

Emergent design is the ability to adapt and evolve a software design to new concepts or changes.

Event Sourcing

Event Sourcing ensures that all changes to application state are stored as a sequence of events, they can be used to reconstruct past states and also cope with retroactive changes.

Event Storming

Event Storming is a workshop-based method to find out details of the domain of a software program.

F

FaaS - Function as a Service

Function as a Service is based on functions that can be triggered by events, most of the times it's called serverless architecture.

Feature Toggle

A feature toggle is a mechanism that allows functionality to be turned enabled/disabled via a flag.

Forward Secrecy

(Perfect) Forward Secrecy means that the keys used to encrypt and decrypt information will be changed frequently, so if the latest key is compromised, it exposes only a small portion of the user's data.

Functional Programming

Functional programming is a way of thinking about software development by following the following concepts: pure functions, recursion, referential transparency, first-class & higher-order function, immutable variables.

G

GRASP - General Responsibility Assignment Software Patterns

GRASP consists of a series of guidelines to solve common problems in object-oriented design: controller, creator, indirection, information expert, low coupling, high cohesion, polymorphism, protected variations, and pure fabrication.

H

Hofstadter's Law

It always takes longer than you expect, even when you take into account Hofstadter's Law.

HSTS - HTTP Strict Transport Security

HSTS is a technology that secures HTTPS web servers against downgrade attacks.

I

IaaS - Infrastructure as a Service

Infrastructure as a Service is offered by cloud vendors and provides access to computing resources such as servers, storage and networking.

IaC - Infrastructure as Code

Infrastructure as Code is the management of infrastructure in a descriptive and versioned manner.

K

Kerckhoffs's Principle

In cryptography, a system should be secure even if everything about the system, except for a small piece of information - the key - is public knowledge.

Knuth's optimization principle

Premature optimization is the root of all evil.

KPI - Key Performance Indicators

The KPIs are the critical key indicators of progress toward an intended goal.

L

Lambda

A lambda is a small anonymous function that can be passed as an argument.

Linus's Law

Given enough eyeballs, all bugs are shallow.

M

Microservices

Microservices are an architectural approach to building applications as a composition of small independent services.

Micro Frontends

Micro Frontend architecture is a design approach to decompose a frontend app into mini-apps.

Moore's Law

The power of computers per unit cost doubles every 24 months. The most popular version states: The number of transistors on an integrated circuit will double in about 18 months.

MTBF - Mean Time Between Failures

MTBF is the average time between repairable failures, the metric tracks the availability and reliability of a product.

MTTA - Mean Time To Acknowledge

MTTA is the average time it takes from when an alert is triggered to when work begins on the issue, the metric tracks the team responsiveness.

MTTF - Mean Time To Failure

MTTF is the average time between non-repairable failures.

MTTR - Mean Time To Recovery

MTTR is the average time it takes to recover a system failure.

Mutation Testing

Mutation Testing is a kind of testing where certain statement in the code are mutated to check whether the test cases are covering those changes.

N**Ninety-ninety rule**

The first 90% of the code takes 10% of the time. The remaining 10% takes the other 90% of the time.

Norvig's Law

Any technology that surpasses 50% penetration will never double again (in any number of months).

O

Observability

Observability is a technical solution that enables to actively debug a system by analysing their properties and patterns.

OLAP - Online Analytical Processing

OLAP systems have the primary objective of data analysis and not data processing.

OLTP - Online Transaction Processing

OLTP systems have the primary objective is data processing and not data analysis.

P

PaaS - Platform as a Service

PaaS is a complete cloud environment where develop and deploy software onto.

Pareto Principle

For many phenomena, 80% of consequences stem from 20% of the causes.

Pipelines

A CI/CD pipeline automates the software delivery process: builds the code, runs the tests, and deploys a new version of the application.

R

Refactoring

Refactoring is the technique for restructuring an existing code and altering its internal structure without changing its external behaviour.

RPO - Recovery Point Objective

An RPO describes the interval of time during a disruption before the quantity of data lost exceeds the tolerance defined.

RTO - Recovery Time Objective

An RTO is the duration of time within which a business process must be restored after a disaster to avoid unacceptable consequences.

RUM - Real User Monitoring

Real User Monitoring is a passive monitoring technique that collects and analyses user interactions to improve the end-user experience.

S

SaaS - Software as a Service

Software as a Service is an on-demand software hosted by the provider who gives access to the product usually via a subscription model.

Secret Management

Secrets management refers to tools and methods for managing digital authentication credentials (secrets, such as password and API keys).

Serverless

Serverless computing is a cloud model where the cloud provider is responsible for executing a piece of code by dynamically allocating the resources, sometimes referred to as FaaS.

Service Mesh

A service mesh is an infrastructure layer designed to handle a high volume of network communications among application services, generally, it provides service discovery, load balancing, encryption, observability, traceability, authentication & authorization, and support for the circuit breaker pattern.

SLA - Service Level Agreement

An SLA is an agreement between the provider and the client about measurable metrics.

SLI - Service Level Indicator

An SLI is the actual measurement of an SLO.

SLO - Service Level Objective

An SLO is an agreement about a specific metric within an SLA.

T

TDD - Test Driven Development

TDD is a process that relies on the repetition of a short development cycle: turn the requirements into specific test cases, the minimal code will be written so that the tests pass, then (optionally) the code will be refactored.

Technical Debt

Technical debt refers to the consequences due to poorly written code and compromises during the development, taking in consideration the effort that has to be done to “repay” the debt to go back to acceptable levels.

U

Uptime

The website uptime is the time that a website or web service is available to the users over a given period.

V

Velocity

The velocity, in an “agile” iteration, is the sum of all the story points associated with each completed user stories during that iteration.

W

Wirth’s law

Software gets slower faster than hardware gets faster.

Y

Yak Shaving

Yak shaving is a term that refers to a series of nested (never-ending) tasks that need to be performed before a project can progress to its next stage.

Yoda Conditions

The Yoda conditions is a programming style where the variable and constant will be inverted in a conditional expression.

Week 15: Caching



Time needed: 19 hours

Antipatterns

1. [Preventing the Dogpile Effect](#)³⁰⁵
Preventing the Dogpile Effect
2. [The Caching Antipattern](#)³⁰⁶
The Caching Antipattern

Best Practice

1. [Caching guidance - Best practices for cloud applications | Microsoft Docs](#)³⁰⁷
Caching guidance - Best practices for cloud applications | Microsoft Docs
2. [Caching best practices & max-age gotchas - JakeArchibald.com](#)³⁰⁸
Caching best practices & max-age gotchas - JakeArchibald.com

³⁰⁵<https://www.sobstel.org/blog/preventing-dogpile-effect/>

³⁰⁶<https://www.hidefsoftware.co.uk/2016/12/25/the-caching-antipattern/>

³⁰⁷<https://docs.microsoft.com/en-us/azure/architecture/best-practices/caching>

³⁰⁸<https://jakearchibald.com/2016/caching-best-practices/>

ESI

1. [ESI Language Specification 1.0³⁰⁹](#)
ESI Language Specification 1.0

Invalidation

1. [An Introduction to Cache Invalidation — FOSHttpCache Documentation³¹⁰](#)
An Introduction to Cache Invalidation — FOSHttpCache Documentation

Security

1. [Cache Poisoning Software Attack | OWASP Foundation³¹¹](#)
Cache Poisoning Software Attack | OWASP Foundation

Strategies

1. [A Beginner's Guide to Memoization with JavaScript | by Mahdhi Rezvi | Jan, 2021 | Bits and Pieces³¹²](#)
A Beginner's Guide to Memoization with JavaScript | by Mahdhi Rezvi | Jan, 2021 | Bits and Pieces
2. [Introduction and Strategies To Handle Challenges in Caching | by Aastikta Sharma | Jan, 2021 | Better Programming³¹³](#)

³⁰⁹<https://www.w3.org/TR/esi-lang/>

³¹⁰<https://foshttpcache.readthedocs.io/en/stable/invalidation-introduction.html>

³¹¹https://owasp.org/www-community/attacks/Cache_Poisoning

³¹²<https://blog.bitsrc.io/a-beginners-guide-to-memoization-with-javascript-59d9c818f4c8>

³¹³<https://betterprogramming.pub/introduction-and-strategies-to-handle-challenges-in-caching-c619d51882c0>

Introduction and Strategies To Handle Challenges in Caching
| by Aastikta Sharma | Jan, 2021 | Better Programming

3. [Cache Coherence Techniques](#)³¹⁴
Cache Coherence Techniques
4. [A Hitchhiker's Guide to Caching Patterns | Hazelcast](#)³¹⁵
A Hitchhiker's Guide to Caching Patterns | Hazelcast

Web

1. [HTTP caching - HTTP | MDN](#)³¹⁶
HTTP caching - HTTP | MDN
2. [Web Caching Basics: Terminology, HTTP Headers, and Caching Strategies | DigitalOcean DigitalOcean home DigitalOcean Homepage](#)³¹⁷
Web Caching Basics: Terminology, HTTP Headers, and Caching Strategies | DigitalOcean DigitalOcean home DigitalOcean Homepage
3. [HTTP Cache Headers - A Complete Guide](#)³¹⁸
HTTP Cache Headers - A Complete Guide

Videos

1. [“Caching at Netflix: The Hidden Microservice” by Scott Mansfield](#)³¹⁹
2. [Best Practices for Caching](#)³²⁰

³¹⁴<http://joomla.di.unipi.it/~vannesch//SPA%202011-12/Silvia-cache-coherence-.pdf>

³¹⁵<https://hazelcast.com/blog/a-hitchhikers-guide-to-caching-patterns/>

³¹⁶<https://developer.mozilla.org/en-US/docs/Web/HTTP/Caching>

³¹⁷<https://www.digitalocean.com/community/tutorials/web-caching-basics-terminology-http-headers-and-caching-strategies>

³¹⁸<https://www.keycdn.com/blog/http-cache-headers>

³¹⁹<https://www.youtube.com/watch?v=Rzdxgx3RC0Q>

³²⁰<https://www.youtube.com/watch?v=iNH6APQzIog>

Books

1. □ [Getting Started with Varnish Cache: Accelerate Your Web Applications](#)³²¹
2. □ [Redis in Action](#)³²²

³²¹<https://www.amazon.com/Getting-Started-Varnish-Cache-Applications/dp/149197222X>

³²²<https://redislabs.com/ebook/redis-in-action/>

Notes

(free space)

Notes

(free space)

Week 16: DDD, Functional & CQRS/ES



Time needed: 17 hours

Books

1. Domain-Driven Design: Tackling Complexity in the Heart of Software³²³

³²³<https://www.amazon.com/Domain-Driven-Design-Tackling-Complexity-Software/dp/0321125215>

Notes

(free space)

Notes

(free space)

Week 17: DDD, Functional & CQRS/ES



Time needed: 11 hours

DDD

1. [DDD Reference](#)³²⁴
Domain-Driven Design Reference: Definitions and Pattern Summaries - Domain Language
2. [Summary of a four days DDD training](#)³²⁵
Summary of a four days DDD training | by Thomas Ferro | Medium
3. [The beginner's guide to BDD \(behaviour-driven development\)](#)³²⁶
The beginner's guide to BDD (behaviour-driven development)

³²⁴<https://domainlanguage.com/product/domain-driven-design-reference/>

³²⁵<https://medium.com/@t.ferro184/summary-of-a-four-days-ddd-training-74103a6d99a1>

³²⁶<https://inviqa.com/blog/bdd-guide>

Functional

1. [Benefits of Functional Programming by Example](#)³²⁷
Benefits of Functional Programming by Example | by Nick McCurdy | Medium
2. [Don't Be Scared Of Functional Programming](#)³²⁸
Don't Be Scared Of Functional Programming — Smashing Magazine Clear Search Back to top
3. [So You Want to be a Functional Programmer \(Part 1\)](#)³²⁹
So You Want to be a Functional Programmer (Part 1) | by Charles Scalfani | Medium
4. [So You Want to be a Functional Programmer \(Part 2\)](#)³³⁰
So You Want to be a Functional Programmer (Part 2) | by Charles Scalfani | Medium
5. [So You Want to be a Functional Programmer \(Part 3\)](#)³³¹
So You Want to be a Functional Programmer (Part 3) | by Charles Scalfani | Medium
6. [So You Want to be a Functional Programmer \(Part 4\)](#)³³²
So You Want to be a Functional Programmer (Part 4) | by Charles Scalfani | Medium
7. [So You Want to be a Functional Programmer \(Part 5\)](#)³³³
So You Want to be a Functional Programmer (Part 5) | by Charles Scalfani | Medium

³²⁷<https://medium.com/@nickmccurdy/benefits-of-functional-programming-by-example-76f1135b0b18>

³²⁸<https://www.smashingmagazine.com/2014/07/dont-be-scared-of-functional-programming/>

³²⁹<https://medium.com/@cscalfani/so-you-want-to-be-a-functional-programmer-part-1-f15e387e536>

³³⁰<https://medium.com/@cscalfani/so-you-want-to-be-a-functional-programmer-part-2-7005682cec4a>

³³¹<https://medium.com/@cscalfani/so-you-want-to-be-a-functional-programmer-part-3-1b0fd14eb1a7>

³³²<https://medium.com/@cscalfani/so-you-want-to-be-a-functional-programmer-part-4-18fbe3ea9e49>

³³³<https://medium.com/@cscalfani/so-you-want-to-be-a-functional-programmer-part-5-c70adc9cf56a>

8. [So You Want to be a Functional Programmer \(Part 6\)](#)³³⁴
So You Want to be a Functional Programmer (Part 6) | by Charles Scalfani | Medium

CQRS

1. [CQRS: What? Why? How?. CQRS is a useful pattern to reason... | by Stéphane Derosiaux | Medium](#)³³⁵
CQRS: What? Why? How?. CQRS is a useful pattern to reason... | by Stéphane Derosiaux | Medium
2. [CQRS pattern - Azure Architecture Center | Microsoft Docs](#)³³⁶
CQRS pattern - Azure Architecture Center | Microsoft Docs
3. [CQRS](#)³³⁷
CQRS

Event Sourcing

1. [Event Sourcing](#)³³⁸
Event Sourcing
2. [Event Sourcing and CQRS - Event Store Blog](#)³³⁹
Event Sourcing and CQRS - Event Store Blog
3. [What they don't tell you about event sourcing | by Hugo Rocha | Medium](#)³⁴⁰
What they don't tell you about event sourcing | by Hugo Rocha | Medium

³³⁴<https://medium.com/@cscalfani/so-you-want-to-be-a-functional-programmer-part-6-db502830403>

³³⁵<https://medium.com/@sderosiaux/cqrs-what-why-how-945543482313>

³³⁶<https://docs.microsoft.com/en-us/azure/architecture/patterns/cqrs>

³³⁷<https://martinfowler.com/bliki/CQRS.html>

³³⁸<https://martinfowler.com/eaDev/EventSourcing.html>

³³⁹<https://www.eventstore.com/blog/event-sourcing-and-cqrs>

³⁴⁰<https://medium.com/@hugo.oliveira.rocha/what-they-dont-tell-you-about-event-sourcing-6afc23c69e9a>

Videos

1. □ [Pim Elshoff: Technically DDD Video @ DPC2018³⁴¹ | Slides³⁴²](#)
2. □ [Greg Young - CQRS and Event Sourcing - Code on the Beach 2014 - YouTube³⁴³](#)

Books

1. □ [The InfoQ eMag: Domain-Driven Design in Practice³⁴⁴](#)
The InfoQ eMag: Domain-Driven Design in Practice
2. □ [Domain Driven Design Quickly³⁴⁵](#)
Domain Driven Design Quickly
3. □ [Domain-Driven Design Object-Orientation Done Right³⁴⁶](#)
Domain-Driven Design - Dzone Refcardz
4. □ [SOA Patterns Service-Orient Your Enterprise³⁴⁷](#)
SOA Patterns - Dzone Refcardz
5. □ [The Anatomy Of Domain-Driven Design - Booklet³⁴⁸](#)
Anatomy Of... by Scott Millett et al. [PDF/iPad/Kindle]

³⁴¹<https://www.youtube.com/watch?v=JpcNeeetijo>

³⁴²<https://speakerdeck.com/pelshoff/technically-ddd-v3>

³⁴³<https://www.youtube.com/watch?v=JHGkaShoyNs>

³⁴⁴<https://www.infoq.com/minibooks/emag-domain-driven-design/>

³⁴⁵<https://www.infoq.com/minibooks/domain-driven-design-quickly/>

³⁴⁶<https://dzone.com/refcardz/getting-started-domain-driven>

³⁴⁷<https://dzone.com/refcardz/soa-patterns>

³⁴⁸<https://leanpub.com/theanatomyofdomain-drivendesign>

Notes

(free space)

Notes

(free space)

Week 18: Networking



Time needed: 17 hours

HTTP

1. [HTTP for servers](#)³⁴⁹
HTTP for servers

IP

1. [Understanding IP Addressing and CIDR Charts](#)³⁵⁰
Understanding IP Addressing and CIDR Charts — RIPE Network Coordination Centre
2. [CIDR.xyz](#)³⁵¹
CIDR.xyz
3. [Understanding IP Addressing: Everything You Ever Wanted To Know](#)³⁵²

³⁴⁹<http://www.and.org/texts/server-http>

³⁵⁰<https://www.ripe.net/about-us/press-centre/understanding-ip-addressing>

³⁵¹<https://cidr.xyz/>

³⁵²<http://pages.di.unipi.it/ricci/501302.pdf>

Time

1. [UTC is enough for everyone...right?³⁵³](#)
UTC is Enough for Everyone, Right?
-

Books

1. [The DevOps Handbook³⁵⁴](#)

³⁵³<https://zachholman.com/talk/utc-is-enough-for-everyone-right>

³⁵⁴<https://www.amazon.com/Devops-Handbook-World-Class-Reliability-Organizations/dp/1942788002>

Notes

(free space)

Notes

(free space)

Week 19: Cloud



Time needed: 23 hours

Cloud

1. [Cloud Design Patterns](#)³⁵⁵
Cloud design patterns - Azure Architecture Center | Microsoft Docs
2. [How to Succeed with Cloud-native Applications](#)³⁵⁶
How to Succeed with Cloud-native Applications | by Tijl Dullers | FAUN | Medium
3. [Don't get locked up into avoiding lock-in](#)³⁵⁷
Don't get locked up into avoiding lock-in
4. [Cloud Computing Tutorial for Beginners: What is & Architecture](#)³⁵⁸
Cloud Computing Tutorial for Beginners: What is & Architecture
5. [Top 25 Cloud Computing Service Provider Companies \(2021\)](#)³⁵⁹
Top 25 Cloud Computing Service Provider Companies (2021)

³⁵⁵<https://docs.microsoft.com/en-us/azure/architecture/patterns/>

³⁵⁶<https://medium.com/faun/how-to-succeed-with-cloud-native-applications-f22ecd3f746>

³⁵⁷<https://martinfowler.com/articles/oss-lockin.html>

³⁵⁸<https://www.guru99.com/cloud-computing-for-beginners.html>

³⁵⁹<https://www.guru99.com/cloud-computing-service-provider.html>

6. [You Are Locked In ... Deal With It! | Cloudscaling](#)³⁶⁰
[You Are Locked In ... Deal With It! | Cloudscaling](#)

Best Practices

1. [AWS Well-Architected Framework - Operational Excellence Pillar](#)³⁶¹
 2. [AWS Well-Architected Framework - Security Pillar](#)³⁶²
 3. [AWS Well-Architected Framework - Reliability Pillar](#)³⁶³
 4. [AWS Well-Architected Framework - Performance Efficiency Pillar](#)³⁶⁴
 5. [AWS Well-Architected Framework - Cost Optimization Pillar](#)³⁶⁵
-

Books

1. [Cloud Architecture Patterns](#)³⁶⁶
2. [97 Things Every Cloud Engineer Should Know](#)³⁶⁷

³⁶⁰<http://cloudscaling.com/blog/openstack/you-are-locked-in-deal-with-it/>

³⁶¹<https://docs.aws.amazon.com/wellarchitected/latest/operational-excellence-pillar/wellarchitected-operational-excellence-pillar.pdf>

³⁶²<https://docs.aws.amazon.com/wellarchitected/latest/security-pillar/wellarchitected-security-pillar.pdf>

³⁶³<https://docs.aws.amazon.com/wellarchitected/latest/reliability-pillar/wellarchitected-reliability-pillar.pdf>

³⁶⁴<https://docs.aws.amazon.com/wellarchitected/latest/performance-efficiency-pillar/wellarchitected-performance-efficiency-pillar.pdf>

³⁶⁵<https://docs.aws.amazon.com/wellarchitected/latest/cost-optimization-pillar/wellarchitected-cost-optimization-pillar.pdf>

³⁶⁶<https://www.amazon.com/Cloud-Architecture-Patterns-Bill-Wilder/dp/1449319777>

³⁶⁷<https://www.amazon.com/Things-Every-Cloud-Engineer-Should/dp/1492076732>

Notes

(free space)

Notes

(free space)

Week 20: DevOps



Time needed: 10 hours

Deployments

1. [Production deployment guides](#)³⁶⁸
Production deployment guides
2. [Production Readiness Checklist](#)³⁶⁹
Production Readiness Checklist

Infrastructure

1. [5 Common Server Setups For Your Web Application](#)³⁷⁰
5 Common Server Setups For Your Web Application | DigitalOcean DigitalOcean home DigitalOcean Homepage

³⁶⁸<https://gruntwork.io/guides>

³⁶⁹<https://gruntwork.io/devops-checklist/>

³⁷⁰<https://www.digitalocean.com/community/tutorials/5-common-server-setups-for-your-web-application>

MVP

1. [Your ultimate guide to Minimum Viable Product \(+great examples\)](#)³⁷¹
Your ultimate guide to Minimum Viable Product (+great examples) | Fast Monkeys – Official Blog
2. [How To Create a Minimum Viable Product](#)³⁷²
How To Create a Minimum Viable Product
3. [Spikes, POCs, Prototypes and the MVP](#)³⁷³
Spikes, POCs, Prototypes and the MVP | by Leigh Garland | STUDIO ZERO | Medium

SRE

1. [School of SRE](#)³⁷⁴
SchoolOfSRE

Tools

1. [htop explained](#)³⁷⁵
htop explained | peteris.rocks
2. [Linux Performance Analysis in 60,000 Milliseconds](#)³⁷⁶
Linux Performance Analysis in 60,000 Milliseconds | by Netflix Technology Blog | Netflix TechBlog

³⁷¹<https://blog.fastmonkeys.com/2014/06/18/minimum-viable-product-your-ultimate-guide-to-mvp-great-examples/>

³⁷²<https://code.tutsplus.com/articles/how-to-create-a-minimum-viable-product--cms-22245>

³⁷³<https://medium.com/studio-zero/spikes-pocs-prototypes-and-the-mvp-5cdffa1b7367>

³⁷⁴<https://linkedin.github.io/school-of-sre/>

³⁷⁵<https://peteris.rocks/blog/htop/>

³⁷⁶<https://netflixtechblog.com/linux-performance-analysis-in-60-000-milliseconds-acc010403c55>

3. [The Ultimate DevOps Tool Chest³⁷⁷](#)
The Ultimate DevOps Tool Chest | Digital.ai
4. [A tcpdump Tutorial with Examples — 50 Ways to Isolate Traffic³⁷⁸](#)
A tcpdump Tutorial with Examples — 50 Ways to Isolate Traffic | Daniel Miessler search mail mail mail

Mix

1. [Practical DevOps Learning Path — from where should i start ?³⁷⁹](#)
Practical DevOps Learning Path — from where should i start ? | by Abdennoor Toumi | Jan, 2021 | Medium
2. [DORA research program³⁸⁰](#)
DORA research program
3. [CALMS Framework | Atlassian³⁸¹](#)
CALMS Framework | Atlassian

Books

1. [The Cynefin Mini-Book³⁸²](#)
The Cynefin Mini-Book
2. [Foundations of RESTful Architecture³⁸³](#)
Foundations of RESTful Architecture - Dzone Refcardz

³⁷⁷<https://xebialabs.com/the-ultimate-devops-tool-chest/>

³⁷⁸<https://danielmiessler.com/study/tcpdump/>

³⁷⁹<https://abdennoor.medium.com/practical-devops-learning-path-from-where-should-i-start-9d536a5a7250>

³⁸⁰<https://www.devops-research.com/research.html>

³⁸¹<https://www.atlassian.com/devops/frameworks/calms-framework>

³⁸²<https://www.infoq.com/minibooks/cynefin-mini-book/>

³⁸³<https://dzone.com/refcardz/rest-foundations-restful>

3. □ [The InfoQ eMag: Tech Ethics](#)³⁸⁴
The InfoQ eMag: Tech Ethics
4. □ [Continuous Integration Patterns and Anti-Patterns](#)³⁸⁵
Continuous Integration - Dzone Refcardz
5. □ [Continuous Delivery Patterns and Anti-Patterns in the Software Lifecycle](#)³⁸⁶
Continuous Delivery - Dzone Refcardz

³⁸⁴<https://www.infoq.com/minibooks/emag-tech-ethics/>

³⁸⁵<https://dzone.com/refcardz/continuous-integration>

³⁸⁶<https://dzone.com/refcardz/continuous-delivery-patterns>

Notes

(free space)

Notes

(free space)

Week 21: Security



Time needed: 19 hours

SSL/TLS

1. [Everything you need to know about HTTP security headers](#)³⁸⁷
Appcanary - Everything you need to know about HTTP security headers
2. [TLS/SSL Explained: TLS/SSL Terminology and Basics](#)³⁸⁸
TLS/SSL Explained: TLS/SSL Terminology and Basics - DZone Security

Mix

1. [I'm harvesting credit card numbers and passwords from your site. Here's how.](#)³⁸⁹
I'm harvesting credit card numbers and passwords from your site. Here's how. | by David Gilbertson | HackerNoon.com | Medium

³⁸⁷<https://blog.appcanary.com/2017/http-security-headers.html>

³⁸⁸<https://dzone.com/articles/tlssl-terminology-and-basics>

³⁸⁹<https://medium.com/hackernoon/im-harvesting-credit-card-numbers-and-passwords-from-your-site-here-s-how-9a8cb347c5b5>

2. [Part 2: How to stop me harvesting credit card numbers and passwords from your site³⁹⁰](#)

Part 2: How to stop me harvesting credit card numbers and passwords from your site | by David Gilbertson | HackerNoon.com | Medium

Books

1. <https://www.amazon.com/Certified-Ethical-Hacker-Study-Guide/dp/1119533198>
CEH v10 Certified Ethical Hacker Study Guide: 9781119533191:
Computer Science Books @ Amazon.com

³⁹⁰<https://medium.com/hackernoon/part-2-how-to-stop-me-harvesting-credit-card-numbers-and-passwords-from-your-site-844f739659b9>

Notes

(free space)

Notes

(free space)

Week 22: Architecture



Time needed: 16 hours

Distributed

1. [Patterns of Distributed Systems](#)³⁹¹
Patterns of Distributed Systems

Documentation

1. [Agile software architecture documentation](#)³⁹²
Agile software architecture documentation - Coding the Architecture

Enterprise

1. [Catalog of Patterns of Enterprise Application Architecture](#)³⁹³
Catalog of Patterns of Enterprise Application Architecture

³⁹¹<https://martinfowler.com/articles/patterns-of-distributed-systems/>

³⁹²http://www.codingthearchitecture.com/2016/05/31/agile_software_architecture_documentation.html

³⁹³<https://martinfowler.com/eaCatalog/index.html>

2. [Software Architecture Monday - Enterprise Architecture Lessons³⁹⁴](#)
Software Architecture Monday | Developer to Architect | Mark Richards

Event-Driven

1. [Software Architecture Monday - Event-Driven Architecture Lessons³⁹⁵](#)
Software Architecture Monday | Developer to Architect | Mark Richards

Hexagonal

1. [DDD, Hexagonal, Onion, Clean, CQRS, ... How I put it all together³⁹⁶](#)
DDD, Hexagonal, Onion, Clean, CQRS, ... How I put it all together – @hgraca

Microservices

1. [Microservices³⁹⁷](#)
Microservices
2. [Seven Microservices Anti-patterns³⁹⁸](#)
Seven Microservices Anti-patterns

³⁹⁴<https://www.developertoarchitect.com/lessons-enterprise.html>

³⁹⁵<https://www.developertoarchitect.com/lessons-eda.html>

³⁹⁶<https://herbertograca.com/2017/11/16/explicit-architecture-01-ddd-hexagonal-onion-clean-cqrs-how-i-put-it-all-together/>

³⁹⁷<https://martinfowler.com/articles/microservices.html>

³⁹⁸<https://www.infoq.com/articles/seven-uservices-antipatterns/>

3. [Software Architecture Monday - Microservices Lessons](#)³⁹⁹
Software Architecture Monday | Developer to Architect |
Mark Richards

The Twelve Factors App

1. [I. Codebase](#)⁴⁰⁰
The Twelve-Factor App
2. [II. Dependencies](#)⁴⁰¹
The Twelve-Factor App
3. [III. Config](#)⁴⁰²
The Twelve-Factor App
4. [IV. Backing services](#)⁴⁰³
The Twelve-Factor App
5. [V. Build, release, run](#)⁴⁰⁴
The Twelve-Factor App
6. [VI. Processes](#)⁴⁰⁵
The Twelve-Factor App
7. [VII. Port binding](#)⁴⁰⁶
The Twelve-Factor App
8. [VIII. Concurrency](#)⁴⁰⁷
The Twelve-Factor App
9. [IX. Disposability](#)⁴⁰⁸
The Twelve-Factor App

³⁹⁹<https://www.developertoarchitect.com/lessons-microservices.html>

⁴⁰⁰<https://www.12factor.net/codebase>

⁴⁰¹<https://www.12factor.net/dependencies>

⁴⁰²<https://www.12factor.net/config>

⁴⁰³<https://www.12factor.net/backing-services>

⁴⁰⁴<https://www.12factor.net/build-release-run>

⁴⁰⁵<https://www.12factor.net/processes>

⁴⁰⁶<https://www.12factor.net/port-binding>

⁴⁰⁷<https://www.12factor.net/concurrency>

⁴⁰⁸<https://www.12factor.net/disposability>

10. □ [X. Dev/prod parity](#)⁴⁰⁹
The Twelve-Factor App
 11. □ [XI. Logs](#)⁴¹⁰
The Twelve-Factor App
 12. □ [XII. Admin processes](#)⁴¹¹
The Twelve-Factor App
 13. □ [12 Fractured Apps](#)⁴¹²
12 Fractured Apps. Over the years I've witnessed more and...
| by Kelsey Hightower | Medium
-

Books

1. □ [Clean Architecture](#)⁴¹³
Clean Architecture: A Craftsman's Guide to Software Structure and Design (Robert C. Martin Series): Martin, Robert:
9780134494166: Amazon.com: Books

⁴⁰⁹<https://www.12factor.net/dev-prod-parity>

⁴¹⁰<https://www.12factor.net/logs>

⁴¹¹<https://www.12factor.net/admin-processes>

⁴¹²<https://medium.com/@kelseyhightower/12-fractured-apps-1080c73d481c>

⁴¹³<https://www.amazon.com/Clean-Architecture-Craftsmans-Software-Structure/dp/0134494164>

Notes

(free space)

Notes

(free space)

Week 23: Architecture



Time needed: 19 hours

Mix

1. [Software Architecture Monday - General Architecture Lessons](#)⁴¹⁴
Software Architecture Monday | Developer to Architect | Mark Richards
-

Videos

1. Simon Brown: Software Architecture vs. Code [Video @ GOTO 2014](#)⁴¹⁵ | [Slides](#)⁴¹⁶
2. [The Frustrated Architect](#)⁴¹⁷
The Frustrated Architect

⁴¹⁴<https://www.developertoarchitect.com/lessons-general.html>

⁴¹⁵<https://www.youtube.com/watch?v=GAFZcYIO5S0>

⁴¹⁶http://gotocon.com/dl/goto-amsterdam-2014/slides/SimonBrown_SoftwareArchitectureVsCode.pdf

⁴¹⁷<https://www.infoq.com/presentations/The-Frustrated-Architect/>

3. □ [GOTO 2017 • The Many Meanings of Event-Driven Architecture • Martin Fowler](#)⁴¹⁸
GOTO 2017 • The Many Meanings of Event-Driven Architecture • Martin Fowler - YouTube

Books

1. □ [Software Architecture for Developers](#)⁴¹⁹
Software Architecture for Developers
2. □ [Building Evolutionary Architectures](#)⁴²⁰
3. □ [97 Things Every Software Architect Should Know](#)⁴²¹

⁴¹⁸<https://www.youtube.com/watch?v=STKCRUSyP0>

⁴¹⁹<https://softwarearchitecturefordevelopers.com/>

⁴²⁰<https://www.amazon.com/Building-Evolutionary-Architectures-Support-Constant/dp/1491986360>

⁴²¹<https://www.amazon.com/Things-Every-Software-Architect-Should/dp/059652269X>

Notes

(free space)

Notes

(free space)

Week 24: Jobs



Time needed: 12 hours

Freelancing

1. [□ 20 Reasons To Say “No” to Freelancing⁴²²](#)
20 Reasons To Say “No” to Freelancing - Hongkiat Facebook
Twitter Instagram Pinterest LinkedIn Google+ Youtube Red-
dit Dribbble Behance Github CodePen Whatsapp Email

Job Offer

1. [□ How To Prepare For A Salary Negotiation: A Check List⁴²³](#)
How To Prepare For A Salary Negotiation: A Check List -
Adobe 99U Adobe-full-color Adobe-white Adobe-black logo-
white Adobe-full Adobe Behance arrow-down arrow-down
2 arrow-right arrow-right 2 Line close-tablet-03 close-tablet-
05 comment dropdown-close dropdown-open facebook insta-
gram linkedin logo rss search share twitter

⁴²²<https://www.hongkiat.com/blog/reasons-not-to-freelance/>

⁴²³<https://99u.adobe.com/articles/61016/how-to-prepare-for-a-salary-negotiation-a-check-list>

2. [How To Turn Down A Job Offer](#)⁴²⁴
How To Turn Down A Job Offer

Ladder

1. [Sharing Our Engineering Ladder](#)⁴²⁵
Sharing Our Engineering Ladder — RTR Dress Code
2. [The Career Ladder Isn't In The Office](#)⁴²⁶
The Career Ladder Isn't In The Office | by Sean Johnson | HackerNoon.com | Medium

Preparation

1. [Reverse interview](#)⁴²⁷
GitHub - viraptor/reverse-interview: Questions to ask the company during your interview
2. [34 Crucial Tips For Your Next Job Interview](#)⁴²⁸
3. [7 Free Career Aptitude Tests You Can Take Online Today](#)⁴²⁹
7 Free Career Aptitude Tests You Can Take Online Today Logo - Full (Color)

Questions

1. [Interview questions](#)⁴³⁰

⁴²⁴<https://www.forbes.com/sites/jacquelynsmith/2013/08/13/how-to-turn-down-a-job-offer-2>

⁴²⁵<http://dresscode.renttherunway.com/blog/ladder>

⁴²⁶<https://medium.com/hackernoon/the-career-ladder-isnt-in-the-office-43cfe5e3b066>

⁴²⁷<https://github.com/viraptor/reverse-interview>

⁴²⁸<https://www.lifehack.org/articles/work/34-crucial-tips-for-your-next-job-interview.html>

⁴²⁹<https://blog.hubspot.com/marketing/career-aptitude-tests>

⁴³⁰<https://github.com/odino/interviews>

GitHub - odino/interviews: Random questions to ask during interviews.

2. [30 Smart Answers To Tough Interview Questions](#)⁴³¹
 30 Smart Answers To Tough Interview Questions - Business Insider Business Insider logo Close icon Loading Menu icon Search icon Business Insider logo Account icon Account icon Business Life News Reviews Search icon Insider logo Close icon Business Life News All Account icon World globe Facebook Icon Twitter icon LinkedIn icon YouTube icon Instagram icon Business Insider logo Close icon Chevron icon Chevron icon Facebook Icon Email icon Link icon Twitter icon LinkedIn icon Fliboard icon More icon Close icon Loading Close icon
3. [The 20 Toughest Job Interview Questions Heard At Apple, Google, Amazon And Others](#)⁴³²
 Toughest Job Interview Questions - Business Insider Business Insider logo Close icon Loading Menu icon Search icon Business Insider logo Account icon Account icon Business Life News Reviews Search icon Insider logo Close icon Business Life News All Account icon World globe Facebook Icon Twitter icon LinkedIn icon YouTube icon Instagram icon Business Insider logo Close icon Chevron icon Chevron icon Facebook Icon Email icon Link icon Twitter icon LinkedIn icon Fliboard icon More icon Close icon Loading Close icon

Quit

1. [Is It Better To Quit Or Get Fired?](#)⁴³³
 Is It Better To Quit Or Get Fired?

⁴³¹<https://www.businessinsider.com/30-smart-answers-to-tough-interview-questions-2013-8>

⁴³²<https://www.businessinsider.com/toughest-job-interview-questions-2013-7>

⁴³³<https://www.forbes.com/sites/deborahljacobs/2013/07/31/is-it-better-to-quit-or-get-fired>

2. [Programmers: Before you turn 40, get a plan B⁴³⁴](#)
Programmers: Before you turn 40, get a plan B | Improving Software
3. [14 Signs It's Time To Leave Your Job⁴³⁵](#)
14 Signs It's Time To Leave Your Job
4. [How to Tell If You're In a Dead End Job \(and What You Can Do About It\)⁴³⁶](#)
How to Tell If You're In a Dead End Job (and What You Can Do About It)

Remote

1. [Quick, work remote! A guide on how to set up your remote working strategy⁴³⁷](#)
Quick, work remote! A guide on how to set up your remote working strategy · Intense Minimalism

Resume

1. [20 Critical Skills to Add to Resume \(For All Types of Jobs\)⁴³⁸](#)
2. [How To Botox Your Resume To Land A Job⁴³⁹](#)
How To Botox Your Resume To Land A Job
3. [The 5-Step Editing Process for a Perfect Resume⁴⁴⁰](#)
The 5-Step Editing Process for a Perfect Resume

⁴³⁴<https://improvingsoftware.com/2009/05/19/programmers-before-you-turn-40-get-a-plan-b/>

⁴³⁵<https://www.forbes.com/sites/jacquelynsmith/2013/09/04/14-signs-its-time-to-leave-your-job>

⁴³⁶<https://lifehacker.com/how-to-tell-if-youre-in-a-dead-end-job-and-what-you-can-910478489>

⁴³⁷<https://intenseminimalism.com/2020/quick-work-remote/>

⁴³⁸<https://www.lifehack.org/836615/resume-skills>

⁴³⁹<https://www.forbes.com/sites/nextavenue/2013/08/28/how-to-botox-your-resume-to-land-a-job>

⁴⁴⁰<https://mashable.com/2014/03/15/editing-resume/>

4. □ [When Should You Lie on Your Resume?](#)⁴⁴¹
When Should You Lie on Your Resume?
 5. □ [How To Craft The Perfect Web Developer Résumé](#)⁴⁴²
How To Craft The Perfect Web Developer Résumé — Smashing Magazine Clear Search Back to top
-

Books

1. □ [97 Things Every Programmer Should Know](#)⁴⁴³

⁴⁴¹<https://lifehacker.com/when-should-you-lie-on-your-resume-955825518>

⁴⁴²<https://www.smashingmagazine.com/2018/06/web-developer-resume/>

⁴⁴³<https://www.amazon.com/Things-Every-Programmer-Should-Know/dp/0596809484>

Notes

(free space)

Notes

(free space)

Level: Advanced

Level Definition

The individual now looks at the bigger picture. Their focus falls onto understanding the essentials of the framework and often experience frustration when documentation is oversimplified. Proficiency is defined by the self-improvement skills which each person in the stage has. Not only does the proficient practitioner learn from his own mistakes, he observes others as well, anything could be a vital source of information.

- <https://www.360pmo.com/the-five-dreyfus-model-stages/>



Duration: 8 weeks (~2 months)
Average per week: ~14 hours

Schedule

- Week 24: Build Up Dictionary
- Week 25: OS
- Week 26: Productivity

- Week 27: Standards & Best Practices
- Week 28: Standards & Best Practices
- Week 29: Management
- Week 30: Management
- Week 31: Management

Week 24: Build Up Dictionary

C

Chaos Engineering

Chaos Engineering is a disciplined approach to identify failures in advance by experimenting on a system to understand how it will respond under certain conditions.

Conflict Resolutions

Conflict resolution skills are the methods and processes involved in facilitating the peaceful ending of a conflict.

Conway's Law

Any piece of software reflects the organizational structure that produced it. Organizations which design systems are constrained to produce designs which are copies of the communication structures of these organizations.

Critical Thinking

Critical thinking is the ability to engage in reflective and independent thinking that allows making the best decisions possible.

D

Delegation

Delegation is an important management skill, it is the shifting of authority/responsibility for a particular task or decision from one person to another.

F

Four Key Metrics

From the 2014 State of DevOps report have been identified the four key metrics for software delivery performance: lead time, deployment frequency, mean time to restore (MTTR), and change fail percentage.

Functional Reactive Programming

FRP is a combination of functional and reactive paradigms by integrating time flow and compositional events into functional programming.

L

Leadership

Leadership is the art of motivating a group of people to act toward achieving a common goal, usually by directing colleagues with a strategy to meet the company's needs.

P

Postel's Law

Be conservative in what you send, be liberal in what you accept.

Probabilistic Programming

Probabilistic Programming is a tool for statistical modelling, it borrows programming concepts and applies them to the problems of designing and using statistical models.

T

The Peter Principle

In a hierarchy, every employee tends to rise to his level of incompetence.

Threat Modeling

Threat modelling is the practice of identifying and understanding threats and mitigations to protect confidential data or intellectual property.

Three Rs of Enterprise Security

The Three Rs of Enterprise Security: Rotate credentials, Repave every server and application from a known good state and Repair vulnerable operating systems and application.

Time Management

Time management a the process for organizing and planning your time between specific activities by working smarter and not harder so that more can be done in less time.

Week 25: OS



Time needed: 5 hours

Virtualization

1. [What is Virtualization?](#)⁴⁴⁴
What is Virtualization?
2. [What is a Hypervisor? Types of Hypervisors Explained \(1 & 2\)](#)⁴⁴⁵
What is a Hypervisor? Types of Hypervisors Explained (1 & 2)

Container

1. [A Beginner-Friendly Introduction to Containers, VMs and Docker](#)⁴⁴⁶
A Beginner-Friendly Introduction to Containers, VMs and Docker
2. [What is a Container? | App Containerization | Docker](#)⁴⁴⁷
What is a Container? | App Containerization | Docker

⁴⁴⁴<https://www.ibm.com/cloud/learn/virtualization-a-complete-guide>

⁴⁴⁵<https://phoenixnap.com/kb/what-is-hypervisor-type-1-2>

⁴⁴⁶<https://www.freecodecamp.org/news/a-beginner-friendly-introduction-to-containers-vms-and-docker-79a9e3e119b/>

⁴⁴⁷<https://www.docker.com/resources/what-container>

Shell

1. [Learn Shell - Free Interactive Shell Tutorial](#)⁴⁴⁸
Learn Shell - Free Interactive Shell Tutorial

Mixed

1. [An Overview of Operating Systems and Explanation of the Kernel | by Madhavan Nagarajan | Level Up Coding](#)⁴⁴⁹
An Overview of Operating Systems and Explanation of the Kernel | by Madhavan Nagarajan | Level Up Coding
2. [Operating System 1 | Introduction to Operating System, User/Kernel Protection Boundary, and Different Types for Operating Systems | by Adam Edelweiss | SereneField | Medium](#)⁴⁵⁰
Operating System 1 | Introduction to Operating System, User/Kernel Protection Boundary, and Different Types for Operating Systems | by Adam Edelweiss | SereneField | Medium
3. [Operating System 2 | Building the Development/Testing Environment | by Adam Edelweiss | SereneField | Medium](#)⁴⁵¹
Operating System 2 | Building the Development/Testing Environment | by Adam Edelweiss | SereneField | Medium
4. [Operating System 3 | Process and Process Management, Process Control Block, Process Lifecycle, Process Scheduling, and Process Communication | by Adam Edelweiss | SereneField | Medium](#)⁴⁵²
Operating System 3 | Process and Process Management, Process Control Block, Process Lifecycle, Process Scheduling, and

⁴⁴⁸<https://www.learnshell.org/>

⁴⁴⁹<https://levelup.gitconnected.com/operating-system-and-kernel-ef76f4d0bd8e>

⁴⁵⁰<https://medium.com/adamedelwiess/operating-system-1-introduction-to-operating-system-user-kernel-protection-boundary-a408ebe9f463>

⁴⁵¹<https://medium.com/adamedelwiess/operating-system-2-building-the-development-testing-environment-61ba859a9dfe>

⁴⁵²<https://medium.com/adamedelwiess/operating-system-3-process-and-process-management-process-control-block-process-lifecycle-66bbf73ee3f6>

Process Communication | by Adam Edelweiss | SereneField | Medium

5. [Operating System 4 | Socket Programming Experiment | by Adam Edelweiss | SereneField | Medium⁴⁵³](#)
 Operating System 4 | Socket Programming Experiment | by Adam Edelweiss | SereneField | Medium
6. [Operating System 5 | Thread Part 1, Mutual Exclusion and Condition Variable | by Adam Edelweiss | SereneField | Medium⁴⁵⁴](#)
 Operating System 5 | Thread Part 1, Mutual Exclusion and Condition Variable | by Adam Edelweiss | SereneField | Medium
7. [Operating System 6 | Thread Part 2, Reader-Writer Problem, Spurious Wakeups, and Deadlocks | by Adam Edelweiss | SereneField | Medium⁴⁵⁵](#)
 Operating System 6 | Thread Part 2, Reader-Writer Problem, Spurious Wakeups, and Deadlocks | by Adam Edelweiss | SereneField | Medium
8. [Operating System 7 | Thread Part 3, Kernel/User-Level Threads, and Multithreading Patterns | by Adam Edelweiss | SereneField | Medium⁴⁵⁶](#)
 Operating System 7 | Thread Part 3, Kernel/User-Level Threads, and Multithreading Patterns | by Adam Edelweiss | SereneField | Medium
9. [Operating System 8 | A Tutorial for the PThread Programming | by Adam Edelweiss | SereneField | Medium⁴⁵⁷](#)
 Operating System 8 | A Tutorial for the PThread Programming | by Adam Edelweiss | SereneField | Medium

⁴⁵³<https://medium.com/adamedelwiess/operating-system-4-socket-programming-experiment-7e78be945ef5>

⁴⁵⁴<https://medium.com/adamedelwiess/operating-system-4-thread-part-1-abafa6c6bd61>

⁴⁵⁵<https://medium.com/adamedelwiess/operating-system-6-thread-part-2-reader-writer-problem-spurious-wakeups-and-deadlocks-6e28ab161002>

⁴⁵⁶<https://medium.com/adamedelwiess/operating-system-7-thread-part-3-kernel-user-level-threads-and-multithreading-patterns-6b405555d5ac>

⁴⁵⁷<https://medium.com/adamedelwiess/operating-system-8-a-tutorial-for-the-pthread-programming-bf8004e36e70>

10. [Operating System 9 | Socket Programming Experiment 2: Enable IPv4 and IPv6](#) | by Adam Edelweiss | SereneField | Feb, 2021 | Medium⁴⁵⁸
Operating System 9 | Socket Programming Experiment 2: Enable IPv4 and IPv6 | by Adam Edelweiss | SereneField | Feb, 2021 | Medium
11. [Operating System 10 | File I/O Experiment](#) | by Adam Edelweiss | SereneField | Feb, 2021 | Medium⁴⁵⁹
Operating System 10 | File I/O Experiment | by Adam Edelweiss | SereneField | Feb, 2021 | Medium
12. [Operating System 11 | Function Pointer, Callback, Malloc, and Macro Programming](#) | by Adam Edelweiss | SereneField | Feb, 2021 | Medium⁴⁶⁰
Operating System 11 | Function Pointer, Callback, Malloc, and Macro Programming | by Adam Edelweiss | SereneField | Feb, 2021 | Medium
13. [Operating System 12 | Interactions between Kernel-Level Threads and User-Level Threads with SunOS and Linux Examples](#) | by Adam Edelweiss | SereneField | Feb, 2021 | Medium⁴⁶¹
Operating System 12 | Interactions between Kernel-Level Threads and User-Level Threads with SunOS and Linux Examples | by Adam Edelweiss | SereneField | Feb, 2021 | Medium
14. [Operating System 13 | Signals and Interrupts](#) | by Adam Edelweiss | SereneField | Mar, 2021 | Medium⁴⁶²
Operating System 13 | Signals and Interrupts | by Adam Edelweiss | SereneField | Mar, 2021 | Medium

⁴⁵⁸<https://medium.com/adamedelwiess/operating-system-9-socket-programming-experiment-2-enable-ipv4-and-ipv6-c2f034511cd4>

⁴⁵⁹<https://medium.com/adamedelwiess/operating-system-9-file-i-o-experiment-d4f1191bd358>

⁴⁶⁰<https://medium.com/adamedelwiess/operating-system-11-function-pointer-callback-malloc-and-macro-programming-6fb263debc3>

⁴⁶¹<https://medium.com/adamedelwiess/operating-system-12-interactions-between-kernel-level-threads-and-user-level-threads-with-sunos-879b1154df19>

⁴⁶²<https://medium.com/adamedelwiess/operating-system-13-signals-and-interrupts-f0943d8fe287>

15. [Operating System 14 | Event-Driven Model with the Flash Server Example, Experiment Design, Metrics, and Results | by Adam Edelweiss | SereneField | Mar, 2021 | Medium⁴⁶³](#)
Operating System 14 | Event-Driven Model with the Flash Server Example, Experiment Design, Metrics, and Results | by Adam Edelweiss | SereneField | Mar, 2021 | Medium
16. [Operating System 15 | Midterm Review | by Adam Edelweiss | SereneField | Mar, 2021 | Medium⁴⁶⁴](#)
Operating System 15 | Midterm Review | by Adam Edelweiss | SereneField | Mar, 2021 | Medium
17. [Operating System 16 | OS Scheduling, First-Come&First-Serve, Shortest-Job First, Preemption, Priority, Timeslicing, Runqueue, Linux Scheduler, and Scheduling on Multi-CPU Systems | by Adam Edelweiss | SereneField | Mar, 2021 | Medium⁴⁶⁵](#)
Operating System 16 | OS Scheduling, First-Come&First-Serve, Shortest-Job First, Preemption, Priority, Timeslicing, Runqueue, Linux Scheduler, and Scheduling on Multi-CPU Systems | by Adam Edelweiss | SereneField | Mar, 2021 | Medium
18. [Operating System 17 | Memory Management, Pages and Page Tables, Segmentation, Memory Allocation, Copy-On-Write, and Checkpointing | by Adam Edelweiss | SereneField | Mar, 2021 | Medium⁴⁶⁶](#)
Operating System 17 | Memory Management, Pages and Page Tables, Segmentation, Memory Allocation, Copy-On-Write, and Checkpointing | by Adam Edelweiss | SereneField | Mar, 2021 | Medium
19. [Operating System 18 | Inter-Process Communication, Pipe, Message Queue, Socket, Shared-Memory IPC and Its Syn-](#)

⁴⁶³<https://medium.com/adamedelwiess/operating-system-14-event-driven-model-with-the-flash-server-example-experiment-design-metrics-f8b6213907f5>

⁴⁶⁴<https://medium.com/adamedelwiess/operating-system-15-midterm-review-a18d77561476>

⁴⁶⁵<https://medium.com/adamedelwiess/operating-system-16-os-scheduling-first-come-first-serve-shortest-job-first-preemption-ef1b97937b3>

⁴⁶⁶<https://medium.com/adamedelwiess/operating-system-17-memory-management-pages-and-page-tables-segmentation-memory-allocation-6b819b9de555>

[chronization](#) | by Adam Edelweiss | SereneField | Mar, 2021 | [Medium](#)⁴⁶⁷

Operating System 18 | Inter-Process Communication, Pipe, Message Queue, Socket, Shared-Memory IPC and Its Synchronization | by Adam Edelweiss | SereneField | Mar, 2021 | [Medium](#)

20. [Operating System 19 | Libcurl Library, Sys V IPC APIs, and Linux Signals](#) | by Adam Edelweiss | SereneField | Mar, 2021 | [Medium](#)⁴⁶⁸

Operating System 19 | Libcurl Library, Sys V IPC APIs, and Linux Signals | by Adam Edelweiss | SereneField | Mar, 2021 | [Medium](#)

⁴⁶⁷<https://medium.com/adamedelwiess/operating-system-18-inter-process-communication-pipe-message-queue-socket-shared-memory-ipc-ceab3729f0e2>

⁴⁶⁸<https://medium.com/adamedelwiess/operating-system-19-libcurl-library-sys-v-ipc-apis-and-linux-signals-6ff4a8c62e56>

Notes

(free space)

Notes

(free space)

Week 26: Productivity



Time needed: 18 hours

Cognitive Biases

1. [Cognitive Biases in Programming](#)⁴⁶⁹
Cognitive Biases in Programming. As developers, we're familiar with the... | by Yash Ranadive | HackerNoon.com | Medium

Critical Thinking

1. [How to Improve Critical Thinking](#)⁴⁷⁰
How to Improve Critical Thinking | Scott H Young

Decision Making

1. [A Checklist for Making Faster, Better Decisions](#)⁴⁷¹
A Checklist for Making Faster, Better Decisions Navigation
Menu Account Menu Search Menu Close menu Search

⁴⁶⁹<https://medium.com/hackernoon/cognitive-biases-in-programming-5e937707c27b>

⁴⁷⁰<https://www.scotthyoung.com/blog/2019/03/07/improve-critical-thinking/>

⁴⁷¹<https://hbr.org/2016/03/a-checklist-for-making-faster-better-decisions>

2. [The Decision Matrix: How to Prioritize What Matters](#)⁴⁷²
The Decision Matrix: How to Prioritize What Matters
3. [7 mental models you should know for smarter decision making](#)⁴⁷³
The Science Behind Smarter Decision Making: 7 Mental Models To Know

Destructive Approaches

1. [Jobs contribute to workaholism, insomnia, divorce, death](#)⁴⁷⁴
Jobs contribute to workaholism, insomnia, divorce, death - Business Insider Business Insider logo Close icon Loading Menu icon Search icon Business Insider logo Account icon Account icon Business Life News Reviews Search icon Insider logo Close icon Business Life News All Account icon World globe Facebook Icon Twitter icon LinkedIn icon YouTube icon Instagram icon Business Insider logo Close icon Chevron icon Chevron icon Facebook Icon Email icon Link icon Twitter icon LinkedIn icon Fliboard icon More icon Close icon Valuable Not valuable Loading Chevron icon
2. [KPIs, Velocity, and Other Destructive Metrics](#)⁴⁷⁵
KPIs, Velocity, and Other Destructive Metrics | Allen Holub

Stress

1. [30 Free Or Cheap Ways To Reduce Stress And To Refresh Yourself](#)⁴⁷⁶

⁴⁷²<https://fs.blog/2018/09/decision-matrix/>

⁴⁷³<https://thenextweb.com/lifehacks/2016/08/01/989517/>

⁴⁷⁴<https://www.businessinsider.com/disturbing-facts-about-your-job-2011-2>

⁴⁷⁵<https://holub.com/kpis-velocity-and-other-destructive-metrics/>

⁴⁷⁶<https://www.lifehack.org/articles/money/30-free-cheap-ways-reduce-stress-and-refresh-yourself.html>

Time Management

1. [Time Management](#)⁴⁷⁷
Time Management Skills and Training from MindTools.com

Mix

1. [Strategy vs. Tactics: What's the Difference and Why Does it Matter?](#)⁴⁷⁸
Strategy vs. Tactics: Why the Difference Matters
2. [16 Tips for Getting 90 Percent of Your Work Done Before Lunch](#)⁴⁷⁹
16 Tips for Getting 90 Percent of Your Work Done in the Morning | Inc.com logo navigation logo Combined Shape Group 5 Group 3 Fill 1 Group 3 Group 3 Group 5 Group 3 Fill 1 Group 3 Group 3 logo logo navigation logo Combined Shape Shape
3. [26 Time Management Hacks I Wish I'd Known at 20](#)⁴⁸⁰
26 Time Management Hacks I Wish I'd Known at 20
4. [44 ways to be more productive](#)⁴⁸¹
44 ways to be more productive
5. [13 Tech CEOs And Founders Reveal Their Favorite Productivity Hacks To Help You Get More Done](#)⁴⁸²
Tech CEOs Favorite Productivity Hacks - Business Insider Business Insider logo Close icon Loading Menu icon Search icon Business Insider logo Account icon Account icon Business Life News Reviews Search icon Insider logo Close icon

⁴⁷⁷https://www.mindtools.com/pages/main/newMN_HTE.htm

⁴⁷⁸<https://fs.blog/2018/08/strategy-vs-tactics/>

⁴⁷⁹<https://www.inc.com/neil-patel/16-tips-for-getting-90-of-your-work-done-in-the-morning.html>

⁴⁸⁰<https://www.slideshare.net/egarbugli/26-time-management-hacks-i-wish-id-known-at-20>

⁴⁸¹<https://www.stl-training.co.uk/sharing/16-44-ways-be-more-productive.html>

⁴⁸²<https://www.businessinsider.com/tech-ceos-favorite-productivity-hacks-2013-8>

Business Life News All Account icon World globe Facebook
 Icon Twitter icon LinkedIn icon YouTube icon Instagram icon
 Business Insider logo Close icon Chevron icon Chevron icon
 Facebook Icon Email icon Link icon Twitter icon LinkedIn
 icon Fliboard icon More icon Close icon Loading Chevron
 icon Close icon

6. [Pretend Your Time is Worth \\$1,000/Hour and You'll Become 100x More Productive](#)⁴⁸³
 Pretend Your Time is Worth \$1,000/Hour and You'll Become 100x More Productive | by Anthony Moore | The Startup | Medium
7. [Things I Learnt The Hard Way \(in 30 Years of Software Development\)](#)⁴⁸⁴
 Julio BIASON .Net 4.1 | Things I Learnt The Hard Way (in 30 Years of Software Development)

Videos

1. [GOTO 2017 • Forget Velocity, Let's Talk Acceleration • Jessica Kerr](#)⁴⁸⁵
 GOTO 2017 • Forget Velocity, Let's Talk Acceleration • Jessica Kerr - YouTube

Books

1. [97 Things Every Project Manager Should Know](#)⁴⁸⁶

⁴⁸³<https://medium.com/swlh/pretend-your-time-is-worth-1-000-hour-and-youll-become-100x-more-productive-f04628bb3e6d>

⁴⁸⁴<https://blog.juliobiason.me/thoughts/things-i-learnt-the-hard-way/>

⁴⁸⁵https://www.youtube.com/watch?v=Lbcyyu8XB_Y

⁴⁸⁶<https://www.amazon.com/Things-Every-Project-Manager-Should/dp/0596804164>

2. □ **Making Work Visible**⁴⁸⁷

⁴⁸⁷<https://www.amazon.com/Making-Work-Visible-Exposing-Optimize/dp/1942788150>

Notes

(free space)

Notes

(free space)

Week 27: Standards & Best Practices



Time needed: 22 hours

Architecture

1. [Terraform best practices](#)⁴⁸⁸
Welcome - Terraform Best Practices
2. [Your guide to Kubernetes best practices](#)⁴⁸⁹
A guide to our top Kubernetes posts | Google Cloud Blog

Data

1. [Data Management patterns](#)⁴⁹⁰
Data Management patterns - Cloud Design Patterns | Microsoft Docs

⁴⁸⁸<https://www.terraform-best-practices.com/>

⁴⁸⁹<https://cloud.google.com/blog/products/containers-kubernetes/your-guide-kubernetes-best-practices>

⁴⁹⁰<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/data-management>

Design

1. [Design and Implementation patterns](#)⁴⁹¹
Design and Implementation patterns - Cloud Design Patterns | Microsoft Docs

HA

1. [Availability patterns](#)⁴⁹²
Reliability patterns - Cloud Design Patterns | Microsoft Docs
2. [Resiliency patterns](#)⁴⁹³
Reliability patterns - Cloud Design Patterns | Microsoft Docs

Observability

1. [Management and Monitoring patterns](#)⁴⁹⁴
Management and Monitoring patterns - Cloud Design Patterns | Microsoft Docs

Scalability

1. [Performance and Scalability patterns](#)⁴⁹⁵
Reliability patterns - Cloud Design Patterns | Microsoft Docs

⁴⁹¹<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/design-implementation>

⁴⁹²<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/availability>

⁴⁹³<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/resiliency>

⁴⁹⁴<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/management-monitoring>

⁴⁹⁵<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/performance-scalability>

Security

1. [Security patterns](#)⁴⁹⁶
Reliability patterns - Cloud Design Patterns | Microsoft Docs
2. [Secure Programming HOWTO](#)⁴⁹⁷
3. [WHAT DO WE REALLY NEED TO ENCRYPT. CHEAT-SHEET](#)⁴⁹⁸
What Do We Really Need to Encrypt. Cheatsheet
4. [Security architecture anti-patterns](#)⁴⁹⁹

⁴⁹⁶<https://docs.microsoft.com/en-us/azure/architecture/patterns/category/security>

⁴⁹⁷<https://dwheeler.com/secure-programs/Secure-Programs-HOWTO.html>

⁴⁹⁸<https://www.cossacklabs.com/blog/what-we-need-to-encrypt-cheatsheet.html>

⁴⁹⁹<https://www.ncsc.gov.uk/whitepaper/security-architecture-anti-patterns>

Notes

(free space)

Notes

(free space)

Week 28: Standards & Best Practices



Time needed: 12 hours

Books

1. [How to Monitoring the SRE Golden Signals](#)⁵⁰⁰
How to Monitoring the SRE Golden Signals (E-Book)
2. [Framework for Improving Critical Infrastructure Cybersecurity Version 1.1](#)⁵⁰¹
3. [Handbook for Computer Security Incident Response Teams \(CSIRTs\)](#)⁵⁰²
4. [INTERPOL Global Guidelines for Digital Forensics Laboratories](#)⁵⁰³
5. [ISO 31000:2018 Risk management — Guidelines](#)⁵⁰⁴
ISO - ISO 31000:2018 - Risk management — Guidelines
6. [ISO/IEC 27000:2018 Information technology — Security techniques](#)⁵⁰⁵

⁵⁰⁰<https://www.slideshare.net/OpsStack/how-to-monitoring-the-sre-golden-signals-ebook>

⁵⁰¹<https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf>

⁵⁰²https://resources.sei.cmu.edu/asset_files/Handbook/2003_002_001_14102.pdf

⁵⁰³https://www.interpol.int/content/download/13501/file/INTERPOL_DFL_GlobalGuidelinesDigitalForensicsLaboratory.pdf

⁵⁰⁴<https://www.iso.org/standard/65694.html>

⁵⁰⁵<https://www.iso.org/standard/73906.html>

ISO - ISO/IEC 27000:2018 - Information technology — Security techniques — Information security management systems — Overview and vocabulary

Notes

(free space)

Notes

(free space)

Week 29: Management



Time needed: 13 hours

Approaches

1. [Radical Candor — The Surprising Secret to Being a Good Boss⁵⁰⁶](#)
Radical Candor — The Surprising Secret to Being a Good Boss
| First Round Review
2. [The Future of Management Is Teal⁵⁰⁷](#)
The future of management is teal

Leadership

1. [30 Outdated Leadership Practices Holding Your Company Back⁵⁰⁸](#)
30 Outdated Leadership Practices Holding Your Company
Back

⁵⁰⁶<https://firstround.com/review/radical-candor-the-surprising-secret-to-being-a-good-boss/>

⁵⁰⁷<https://www.strategy-business.com/article/00344>

⁵⁰⁸<https://www.forbes.com/sites/mikemyatt/2013/07/28/30-outdated-leadership-practices-holding-your-company-back>

Product/Project Management

1. [15 Ways to Screw Up an IT Project](#)⁵⁰⁹
15 Ways to Screw Up an IT Project | CIO
2. [15 Project Management Quotes That Will Help You Stay Motivated](#)⁵¹⁰
3. [20 Product Prioritization Techniques: A Map and Guided Tour](#)⁵¹¹
20 Product Prioritization Techniques: A Map and Guided Tour
4. [Why Companies Need Full-Time Product Managers \(And What They Do All Day\)](#)⁵¹²
Why Companies Need Full-Time Product Managers (And What They Do All Day) — Smashing Magazine Clear Search
Back to top
5. [Classic Mistakes Enumerated](#)⁵¹³
Classic Mistakes Enumerated

Public Speaking

1. [20 tips for better conference speaking](#)⁵¹⁴
20 tips for better conference speaking ~ Authentic Boredom
2. [The Secret Activity Behind A Successful Speaker](#)⁵¹⁵
The Secret Activity Behind A Successful Speaker

⁵⁰⁹<https://www.cio.com/article/2384088/15-ways-to-screw-up-an-it-project.html>

⁵¹⁰<https://www.lifehack.org/articles/work/15-project-management-quotes-that-will-help-you-stay-motivated.html>

⁵¹¹<https://foldingburritos.com/product-prioritization-techniques/>

⁵¹²<https://www.smashingmagazine.com/2014/09/why-companies-need-full-time-product-managers/>

⁵¹³<https://web.archive.org/web/20170707001328/http://www.stevemcconnell.com/rdenum.htm>

⁵¹⁴http://cameronmoll.com/archives/2009/02/20_tips_better_conference_speaking/

⁵¹⁵<https://www.forbes.com/sites/nickmorgan/2013/08/22/the-secret-activity-behind-a-successful-speaker>

Roles

1. [The Five Flavors of Being a CTO](#)⁵¹⁶
The Five Flavors of Being a CTO
2. [The Role of the CTO: Four Models for Success](#)⁵¹⁷

Quit

1. [17 REASONS NOT TO BE A MANAGER](#)⁵¹⁸
17 Reasons NOT To Be A Manager – charity.wtf
2. [Career Break Or Sabbatical? How To Decide What Is Right For You](#)⁵¹⁹
Career Break Or Sabbatical? How To Decide What Is Right For You | Careershifters
3. [Career alternatives for a burnt-out developer?](#)⁵²⁰
Career alternatives for a burnt-out developer? - programmer burntout | Ask MetaFilter caret-down clock comment email facebook feed go-to-bottom go-to-top heart log-out moon pencil search-white twitter cog list user mefi-shirt bracketed-plus down-arrow html-bracket-left html-bracket-right slash two-lines bold close hyperlink icon_19502 icon_248 icon_299 italic media1 media2 media4 media5 media7 media8 music-note hide show
4. [Surviving being senior \(tech\) management.](#)⁵²¹
Surviving being senior (tech) management. | by kellan | Medium

⁵¹⁶<https://www.linkedin.com/pulse/five-flavors-being-cto-matt-tucker/>

⁵¹⁷http://www.brixtonspa.com/Career/The_Role_of_the_CTO_4Models.pdf

⁵¹⁸<https://charity.wtf/2019/09/08/reasons-not-to-be-a-manager/>

⁵¹⁹<https://www.careershifters.org/expert-advice/career-break-or-sabbatical-how-to-decide-what-is-right-for-you>

⁵²⁰<https://ask.metafilter.com/124950/Career-alternatives-for-a-burntout-developer>

⁵²¹<https://medium.com/@kellan/surviving-being-senior-tech-management-aa6654efd027>

Mix

1. [Your first 90 days as CTO or VP Engineering.](#)⁵²²
Your first 90 days as CTO or VP Engineering.
 2. [The Joel Test For Programmers \(The Simple Programmer Test\)](#)⁵²³
The Joel Test Updated For Programmers
 3. [Learnings from 80 startup CTOs](#)⁵²⁴
Learnings from 80 startup CTOs. I participated in a startup CTO meet-up... | by Javier Escribano | Medium
-

Books

1. [The Manager's Path](#)⁵²⁵

⁵²²<https://lethain.com/first-ninety-days-cto-vpe/>

⁵²³<https://simpleprogrammer.com/joel-test-programmers-simple-programmer-test/>

⁵²⁴<https://medium.com/@fesja/learnings-from-80-startup-ctos-88ddb5f9c024>

⁵²⁵<https://www.amazon.com/Managers-Path-Leaders-Navigating-Growth/dp/1491973897>

Notes

(free space)

Notes

(free space)

Week 30: Management



Time needed: 18 hours

Books

1. [Accelerate](#)⁵²⁶
2. [97 Things Every Engineering Manager Should Know](#)⁵²⁷
Amazon.com: 97 Things Every Engineering Manager Should Know: Collective Wisdom from the Experts (9781492050902): Fournier, Camille: Books

⁵²⁶<https://www.amazon.com/Accelerate-Building-Performing-Technology-Organizations/dp/1942788339>

⁵²⁷<https://www.amazon.com/Things-Every-Engineering-Manager-Should/dp/1492050903>

Notes

(free space)

Notes

(free space)

Week 31: Management



Time needed: 15 hours

Videos

1. □ Andrea Provaglio: Rethinking Leadership [Video @ GOTO 2017](#)⁵²⁸ | [Slides](#)⁵²⁹
2. □ Michael Lopp: The New Manager Death Spiral [Video @ #LeadDevNewYork 2018](#)⁵³⁰ | [Slides](#)⁵³¹
3. □ Patrick Kua: The Constant Life of a Tech Lead [Video @ The Lead Developer UK 2017](#)⁵³² | [Slides](#)⁵³³
4. □ Roy Osherove: Ten Mistakes Team Leaders Make [Video @ Skills Matter 2011](#)⁵³⁴ | [Slides](#)⁵³⁵
5. □ Dan North: Patterns of Effective Teams [Video @ GOTO 2017](#)⁵³⁶ | [Slides](#)⁵³⁷

⁵²⁸<https://www.youtube.com/watch?v=A04Pu5LlzHw>

⁵²⁹https://files.gotocon.com/uploads/slides/conference_7/273/original/GOTO%20Berlin%20-%20Rethinking%20Leadership-2.pdf

⁵³⁰<https://www.youtube.com/watch?v=pAbU3WJ-NBw>

⁵³¹<https://speakerdeck.com/calibrate/9-new-manager-death-spiral>

⁵³²https://www.youtube.com/watch?v=9jd_vpLk50

⁵³³<https://www.slideshare.net/patkua/constant-life-of-a-tech-lead>

⁵³⁴<https://www.youtube.com/watch?v=qhjXc6niO3k>

⁵³⁵<https://www.slideshare.net/royosherove/ten-mistakes-software-team-leaders-make-by-roy-osherove-5whyscom>

⁵³⁶<https://www.youtube.com/watch?v=lvs7VEsQzKY>

⁵³⁷https://files.gotocon.com/uploads/slides/conference_3/62/original/Patterns_of_Effective_Teams%20PDF.pdf

Books

1. □ [Managing Humans](#)⁵³⁸

⁵³⁸<https://www.amazon.com/Managing-Humans-Humorous-Software-Engineering/dp/1484221575>

Notes

(free space)

Notes

(free space)

Appendices

Appendix A: Extra Learning Paths

1. [□ Startup Playbook](#)⁵³⁹
Startup Playbook
2. [□ Awesome lists](#)⁵⁴⁰
GitHub - sindresorhus/awesome: ☒ Awesome lists about all kinds of interesting topics
3. [□ Blockchain Learning Path](#)⁵⁴¹
GitHub - protofire/blockchain-learning-path: A suggested learning path for blockchain development
4. [□ Developer Roadmap](#)⁵⁴²
GitHub - luuctrunc1234/dev-roadmap: the learning path and resource collections to become software developer
5. [□ Kubernetes Learning Path v2.0](#)⁵⁴³
Kubernetes Learning Path | Microsoft Azure
6. [□ Starway to Orione: the Orione Team Learning Path](#)⁵⁴⁴
GitHub - xpeppers/starway-to-orione: The Orione Team Learning Path
7. [□ The of Secret Knowledge](#)⁵⁴⁵
GitHub - trimstray/the-book-of-secret-knowledge: A collection of inspiring lists, manuals, cheatsheets, blogs, hacks, one-liners, cli/web tools and more.
8. [□ Virgilio](#)⁵⁴⁶
GitHub - virgili0/Virgilio: Your new Mentor for Data Science

⁵³⁹<https://playbook.samaltman.com/>

⁵⁴⁰<https://github.com/sindresorhus/awesome>

⁵⁴¹<https://github.com/protofire/blockchain-learning-path>

⁵⁴²<https://github.com/luuctrunc1234/dev-roadmap>

⁵⁴³<https://azure.microsoft.com/en-us/resources/kubernetes-learning-path/>

⁵⁴⁴<https://github.com/xpeppers/starway-to-orione>

⁵⁴⁵<https://github.com/trimstray/the-book-of-secret-knowledge>

⁵⁴⁶<https://github.com/virgili0/Virgilio>

E-Learning.

9. [hacker-laws](#)⁵⁴⁷
GitHub - dwmkerr/hacker-laws: ☒☒ Laws, Theories, Principles and Patterns that developers will find useful. #hacker-laws
10. [ShowPath.tech](#)⁵⁴⁸
GitHub - PJijin/Show-Path: ☒☒ Learning Path for Programmers <https://roadmap.now.sh>
11. [Frontend Development](#)⁵⁴⁹
GitHub - dypsilon/frontend-dev-bookmarks: Manually curated collection of resources for frontend web developers.

⁵⁴⁷<https://github.com/dwmkerr/hacker-laws>

⁵⁴⁸<https://github.com/PJijin/Show-Path>

⁵⁴⁹<https://github.com/dypsilon/frontend-dev-bookmarks>

Appendix B: Exercises

Week 2

1. Let's Echo⁵⁵⁰
2. Looping and Skipping⁵⁵¹
3. A Personalized Echo⁵⁵²
4. The World of Numbers⁵⁵³
5. Comparing Numbers⁵⁵⁴
6. Getting started with conditionals⁵⁵⁵
7. More on Conditionals⁵⁵⁶
8. Cut #1⁵⁵⁷
9. Cut #2⁵⁵⁸
10. Cut #3⁵⁵⁹
11. Cut #4⁵⁶⁰
12. Cut #5⁵⁶¹
13. Cut #6⁵⁶²
14. Cut #7⁵⁶³

⁵⁵⁰<https://www.hackerrank.com/challenges/bash-tutorials-lets-echo/problem>

⁵⁵¹<https://www.hackerrank.com/challenges/bash-tutorials---looping-and-skipping/problem>

⁵⁵²<https://www.hackerrank.com/challenges/bash-tutorials---a-personalized-echo/problem>

⁵⁵³<https://www.hackerrank.com/challenges/bash-tutorials---the-world-of-numbers/problem>

⁵⁵⁴<https://www.hackerrank.com/challenges/bash-tutorials---comparing-numbers/problem>

⁵⁵⁵<https://www.hackerrank.com/challenges/bash-tutorials---getting-started-with-conditionals/problem>

⁵⁵⁶<https://www.hackerrank.com/challenges/bash-tutorials---more-on-conditionals/problem>

⁵⁵⁷<https://www.hackerrank.com/challenges/text-processing-cut-1/problem>

⁵⁵⁸<https://www.hackerrank.com/challenges/text-processing-cut-2/problem>

⁵⁵⁹<https://www.hackerrank.com/challenges/text-processing-cut-3/problem>

⁵⁶⁰<https://www.hackerrank.com/challenges/text-processing-cut-4/problem>

⁵⁶¹<https://www.hackerrank.com/challenges/text-processing-cut-5/problem>

⁵⁶²<https://www.hackerrank.com/challenges/text-processing-cut-6/problem>

⁵⁶³<https://www.hackerrank.com/challenges/text-processing-cut-7/problem>

15. [Cut #8](#)⁵⁶⁴
16. [Cut #9](#)⁵⁶⁵

Week 4

1. [The Hurdle Race](#)⁵⁶⁶
2. [A Very Big Sum](#)⁵⁶⁷
3. [Designer PDF Viewer](#)⁵⁶⁸
4. [Viral Advertising](#)⁵⁶⁹
5. [Solve Me First](#)⁵⁷⁰
6. [Correctness and the Loop Invariant](#)⁵⁷¹
7. [Breaking the Records](#)⁵⁷²
8. [Intro to Tutorial Challenges](#)⁵⁷³
9. [Plus Minus](#)⁵⁷⁴
10. [Staircase](#)⁵⁷⁵
11. [CamelCase](#)⁵⁷⁶
12. [Cats and a Mouse](#)⁵⁷⁷
13. [Bill Division](#)⁵⁷⁸
14. [Utopian Tree](#)⁵⁷⁹
15. [Divisible Sum Pairs](#)⁵⁸⁰

⁵⁶⁴<https://www.hackerrank.com/challenges/text-processing-cut-8/problem>

⁵⁶⁵<https://www.hackerrank.com/challenges/text-processing-cut-9/problem>

⁵⁶⁶<https://www.hackerrank.com/challenges/the-hurdle-race>

⁵⁶⁷<https://www.hackerrank.com/challenges/a-very-big-sum>

⁵⁶⁸<https://www.hackerrank.com/challenges/designer-pdf-viewer>

⁵⁶⁹<https://www.hackerrank.com/challenges/strange-advertising>

⁵⁷⁰<https://www.hackerrank.com/challenges/solve-me-first>

⁵⁷¹<https://www.hackerrank.com/challenges/correctness-invariant>

⁵⁷²<https://www.hackerrank.com/challenges/breaking-best-and-worst-records>

⁵⁷³<https://www.hackerrank.com/challenges/tutorial-intro>

⁵⁷⁴<https://www.hackerrank.com/challenges/plus-minus>

⁵⁷⁵<https://www.hackerrank.com/challenges/staircase>

⁵⁷⁶<https://www.hackerrank.com/challenges/camelcase>

⁵⁷⁷<https://www.hackerrank.com/challenges/cats-and-a-mouse>

⁵⁷⁸<https://www.hackerrank.com/challenges/bon-appetit>

⁵⁷⁹<https://www.hackerrank.com/challenges/utopian-tree>

⁵⁸⁰<https://www.hackerrank.com/challenges/divisible-sum-pairs>

16. [Service Lane](#)⁵⁸¹
17. [Alternating Characters](#)⁵⁸²
18. [Insertion Sort - Part 2](#)⁵⁸³
19. [Sequence Equation](#)⁵⁸⁴
20. [Counting Sort 2](#)⁵⁸⁵
21. [Maximizing XOR](#)⁵⁸⁶
22. [Birthday Cake Candles](#)⁵⁸⁷
23. [The Love-Letter Mystery](#)⁵⁸⁸
24. [Find Digits](#)⁵⁸⁹
25. [Lonely Integer](#)⁵⁹⁰
26. [Grading Students](#)⁵⁹¹
27. [Beautiful Days at the Movies](#)⁵⁹²
28. [Jumping on the Clouds: Revisited](#)⁵⁹³
29. [Marc's Cakewalk](#)⁵⁹⁴
30. [Flipping bits](#)⁵⁹⁵

Week 5

1. [Fizz Buzz Kata](#)⁵⁹⁶

Fizz Buzz Kata

⁵⁸¹<https://www.hackerrank.com/challenges/service-lane>

⁵⁸²<https://www.hackerrank.com/challenges/alternating-characters>

⁵⁸³<https://www.hackerrank.com/challenges/insertionsort2>

⁵⁸⁴<https://www.hackerrank.com/challenges/permutation-equation>

⁵⁸⁵<https://www.hackerrank.com/challenges/countingsort2>

⁵⁸⁶<https://www.hackerrank.com/challenges/maximizing-xor>

⁵⁸⁷<https://www.hackerrank.com/challenges/birthday-cake-candles>

⁵⁸⁸<https://www.hackerrank.com/challenges/the-love-letter-mystery>

⁵⁸⁹<https://www.hackerrank.com/challenges/find-digits>

⁵⁹⁰<https://www.hackerrank.com/challenges/lonely-integer>

⁵⁹¹<https://www.hackerrank.com/challenges/grading>

⁵⁹²<https://www.hackerrank.com/challenges/beautiful-days-at-the-movies>

⁵⁹³<https://www.hackerrank.com/challenges/jumping-on-the-clouds-revisited>

⁵⁹⁴<https://www.hackerrank.com/challenges/marcs-cakewalk>

⁵⁹⁵<https://www.hackerrank.com/challenges/flipping-bits>

⁵⁹⁶<https://kata-log.rocks/fizz-buzz-kata>

2. [Roman Numerals Kata](#)⁵⁹⁷
Roman Numerals Kata
3. [String Calculator Kata](#)⁵⁹⁸
String Calculator Kata
4. [Task List Kata](#)⁵⁹⁹
Task List Kata

Week 6

1. [Tell Don't Ask Kata](#)⁶⁰⁰
Tell Don't Ask Kata
2. [Game of Life Kata](#)⁶⁰¹
Game of Life Kata
3. [Banking Kata](#)⁶⁰²
Banking Kata
4. [Gossiping Bus Drivers Kata](#)⁶⁰³
Gossiping Bus Drivers Kata

Week 11

1. [Race Car Katas - Leaderboard](#)⁶⁰⁴
Race Car Katas - Leaderboard
2. [Mars Rover Kata](#)⁶⁰⁵
Mars Rover Kata

⁵⁹⁷<https://kata-log.rocks/roman-numerals-kata>

⁵⁹⁸<https://kata-log.rocks/string-calculator-kata>

⁵⁹⁹<https://kata-log.rocks/task-list-kata>

⁶⁰⁰<https://kata-log.rocks/tell-dont-ask-kata>

⁶⁰¹<https://kata-log.rocks/game-of-life-kata>

⁶⁰²<https://kata-log.rocks/banking-kata>

⁶⁰³<https://kata-log.rocks/gossiping-bus-drivers-kata>

⁶⁰⁴<https://kata-log.rocks/race-car-katas-leaderboard>

⁶⁰⁵<https://kata-log.rocks/mars-rover-kata>

Week 12

1. Japan Population⁶⁰⁶
2. Population Density Difference⁶⁰⁷
3. Revising Aggregations - Averages⁶⁰⁸
4. Weather Observation Station 16⁶⁰⁹
5. Employee Names⁶¹⁰
6. Japanese Cities' Names⁶¹¹
7. Select By ID⁶¹²
8. Select All⁶¹³
9. Japanese Cities' Attributes⁶¹⁴
10. Revising Aggregations - The Sum Function⁶¹⁵
11. Revising Aggregations - The Count Function⁶¹⁶
12. Revising the Select Query II⁶¹⁷
13. Stand out from the crowd⁶¹⁸
14. Weather Observation Station 14⁶¹⁹
15. Employee Salaries⁶²⁰
16. Average Population⁶²¹
17. Weather Observation Station 1⁶²²
18. Weather Observation Station 13⁶²³

⁶⁰⁶<https://www.hackerrank.com/challenges/japan-population>

⁶⁰⁷<https://www.hackerrank.com/challenges/population-density-difference>

⁶⁰⁸<https://www.hackerrank.com/challenges/revising-aggregations-the-average-function>

⁶⁰⁹<https://www.hackerrank.com/challenges/weather-observation-station-16>

⁶¹⁰<https://www.hackerrank.com/challenges/name-of-employees>

⁶¹¹<https://www.hackerrank.com/challenges/japanese-cities-name>

⁶¹²<https://www.hackerrank.com/challenges/select-by-id>

⁶¹³<https://www.hackerrank.com/challenges/select-all-sql>

⁶¹⁴<https://www.hackerrank.com/challenges/japanese-cities-attributes>

⁶¹⁵<https://www.hackerrank.com/challenges/revising-aggregations-sum>

⁶¹⁶<https://www.hackerrank.com/challenges/revising-aggregations-the-count-function>

⁶¹⁷<https://www.hackerrank.com/challenges/revising-the-select-query-2>

⁶¹⁸<https://www.hackerrank.com/challenges/weather-observation-station-14>

⁶¹⁹<https://www.hackerrank.com/challenges/salary-of-employees>

⁶²⁰<https://www.hackerrank.com/challenges/average-population>

⁶²¹<https://www.hackerrank.com/challenges/weather-observation-station-1>

⁶²²<https://www.hackerrank.com/challenges/weather-observation-station-13>

⁶²³<https://www.hackerrank.com/challenges/weather-observation-station-10>

19. Weather Observation Station 10⁶²⁴
20. African Cities⁶²⁵

Week 15

1. Update List⁶²⁶
2. Reverse a List⁶²⁷
3. Filter Array⁶²⁸
4. List Length⁶²⁹
5. Solve Me First FP⁶³⁰
6. Filter Positions in a List⁶³¹
7. Sum of Odd Elements⁶³²

⁶²⁴<https://www.hackerrank.com/challenges/african-cities>

⁶²⁵<https://www.hackerrank.com/challenges/weather-observation-station-2>

⁶²⁶<https://www.hackerrank.com/challenges/fp-update-list/problem>

⁶²⁷<https://www.hackerrank.com/challenges/fp-reverse-a-list/problem>

⁶²⁸<https://www.hackerrank.com/challenges/fp-filter-array/problem>

⁶²⁹<https://www.hackerrank.com/challenges/fp-list-length/problem>

⁶³⁰<https://www.hackerrank.com/challenges/fp-solve-me-first/problem>

⁶³¹<https://www.hackerrank.com/challenges/fp-filter-positions-in-a-list/problem>

⁶³²<https://www.hackerrank.com/challenges/fp-sum-of-odd-elements>

Appendix C: More IT Books

1. □ [Freely available programming books](#)⁶³³
GitHub - EbookFoundation/free-programming-books: Freely available programming books
2. □ [Sitepoint Library](#)⁶³⁴
Library - SitePoint Premium

⁶³³<https://github.com/EbookFoundation/free-programming-books>

⁶³⁴<https://www.sitepoint.com/premium/library/>

Appendix D: IT Trends

1. [Master Technology Trends, Digital Trends & Digital Business](#)⁶³⁵
Master Technology Trends, Digital Trends & Digital Business
2. [Technology Radar](#)⁶³⁶
Technology Radar | An opinionated guide to technology frontiers | ThoughtWorks
3. [Stack Overflow Annual Developer Survey](#)⁶³⁷
Stack Overflow Developer Survey 2020
4. [Exploit Database - Exploits for Penetration Testers, Researchers, and Ethical Hackers](#)⁶³⁸
5. [The 2020 State of DevOps Report is here!](#)⁶³⁹
2020 State of DevOps Report | presented by Puppet, & CircleCi
6. [The Global CTO Survey 2020 Report](#)⁶⁴⁰
The Global CTO Survey 2020 Report
7. [State Of Remote Work](#)⁶⁴¹
State of Remote Work 2019 | Buffer

⁶³⁵<https://www.gartner.com/en/information-technology/insights/trends-predictions>

⁶³⁶<https://www.thoughtworks.com/radar>

⁶³⁷<https://insights.stackoverflow.com/survey/2020>

⁶³⁸<https://www.exploit-db.com/>

⁶³⁹<https://puppet.com/resources/report/2020-state-of-devops-report>

⁶⁴⁰<https://www.stxnext.com/resources/cto-survey-2020>

⁶⁴¹<https://buffer.com/state-of-remote-work-2019>

Appendix E: Tech People

Alberto Brandolini

EventStorming wizard I like to solve problems and to write software that does that. I'll flood you with sticky notes and call it #EventStorming. I run @avanscoperta

- <http://albertobrandolini.wikidot.com/>
- <http://www.linkedin.com/in/brando>
- <https://twitter.com/ziobrando>

Alistair Cockburn

Co-author of the Agile Manifesto, created Heart of Agile. Creatively showing a humane way forward since the 1990s.

- <https://alistair.cockburn.us/>
- <https://www.linkedin.com/in/alistaircockburn/>
- <https://twitter.com/tothelistair>

Allen Holub

I help you build software better & build better software.

- <https://holub.com/>
- <https://www.linkedin.com/in/allenholub/>
- <https://twitter.com/allenholub>

Andrea Provaglio

Agile Executive Coach, Agile Enterprise Coach, Keynote Speaker, Mentor

- <http://andreaprovaglio.com/>
- <http://linkedin.com/in/provaglio>
- <https://twitter.com/andreaprovaglio>

Andrew Hunt

Andy Hunt is a programmer, consultant, author and publisher. Latest books: <http://conglommora.com> (scifi/adventure)

- <https://toolshed.com/>
- <https://twitter.com/pragmaticandy>

Arie van Bennekum

Co-author #AgileManifesto | Thought leader @Wemanity | Spreading the #agile mindset globally | Keynoter | Paradigm shift expert | Tweets on Agile, travel & fun

- <https://arievandenbennekum.com/>
- <https://www.linkedin.com/in/ariev3/>
- <https://twitter.com/arievandenbennekum>

Camille Fournier

CTO, Speaker, Entrepreneur Distributed systems, dysfunctional programming, and all that management gobbledegook. Putting the rouge in rogue. Author, “The Manager’s Path.” she/her

- <https://www.camilletalk.com/>
- <https://www.linkedin.com/in/camille-fournier-9011812>
- <https://twitter.com/skamille>

Charity Majors

CTO, honeycomb cofounder/CTO @honeycombio, co-wrote Database Reliability Engineering, loves whiskey, rainbows, and Friday deploys. I test in production and so do you. ☒

- <https://charity.wtf/>
- <https://www.linkedin.com/in/charity-majors/>
- <https://twitter.com/mipsytipsy>

Chris Messina

Hashtag inventor · @ProductHunt Community Member of the Year 2020 · Product Designer · Technologist

- <https://chrismessina.me/>
- <https://www.linkedin.com/in/factoryjoe/>
- <https://twitter.com/chrismessina>

Claudio Perrone

I help companies experiment so their business grows. Creator of PopcornFlow for Continuous Innovation & Change. Most companies face inertia. My framework enables ultra-rapid experimentation so that people can introduce change, innovate, and thrive under uncertainty.

- <https://www.linkedin.com/in/claudioperrone/>
- <https://twitter.com/agilesensei>

Dan North

Principal at Dan North & Associates Optimizer of organizations, teams and software, programmer, Agile coach, technologist, troublemaker. Christian, husband, occasional blogger.

- <https://dannorth.net/>
- <https://www.linkedin.com/in/dannorth/>
- <https://twitter.com/tastapod>

Dave Farley

Independent Software Developer and Consultant, Founder and Director of Continuous Delivery Ltd. Co-author of Continuous Delivery book Independent software developer and consultant <http://continuous-delivery.co.uk>

- <http://www.davefarley.net/>
- <https://www.linkedin.com/in/dave-farley-a67927/>
- <https://twitter.com/davefarley77>

Dave Kerr

Expert Associate Partner, Digital at McKinsey & Company ☒☒☒☒ climber, coder, speaker, writer. A wave phenomenon in the chronosynclastic infundibulum. He/Him.

- <https://dwmkerr.com/>
- <https://linkedin.com/in/dwmkerr>
- <https://twitter.com/dwmkerr>

Eric Evans

Software Designer, Domain Language, Inc. Domain Linguist

- <https://www.domainlanguage.com/>
- <https://www.linkedin.com/in/ericevansddd>
- <https://twitter.com/ericevans0>

Erich Gamma

Technical Fellow at Microsoft Software developer and skier. VS Code Dev Lead in Zurich.

- <https://www.linkedin.com/in/erichgamma/>
- <https://twitter.com/erichgamma>

Francesco Cirillo

Author of the Pomodoro Technique * Productivity Expert * Senior Agile Coach * Software Designer

- <https://francescocirillo.com/>
- <https://www.linkedin.com/in/cirillof/>
- <https://twitter.com/cirillof>

Frederic Laloux

(experimenting life without a job identity)

- <https://www.reinventingorganizations.com/>
- <https://www.linkedin.com/in/frederic-laloux-108174/>
- https://twitter.com/fred_laloux

Gene Kim

Author, Researcher, Speaker, Director, DevOps Enthusiast WSJ bestselling author: Unicorn Project! DevOps researcher/enthusiast. Coauthor: Phoenix Project, Accelerate. Host of The Idealcast. Tripwire founder. Clojure.

- <https://itrevolution.com/>
- <https://www.linkedin.com/in/realgenekim/>
- <https://twitter.com/realgenekim>

Greg Young

Greg Young coined the term “CQRS” (Command Query Responsibility Segregation) and it was instantly picked up by the community who have elaborated upon it ever since.

- <https://twitter.com/gregyoung>

Herberto Graça

Principal Developer at CM.com Obsessive enterprise web developer, blogger, speaker, traveler, chopper lover, evangelist of DDD, Lean, Agile and Explicit Architecture

- <https://herbertograca.com/>
- <https://www.linkedin.com/in/hgraca/>
- <https://twitter.com/hgraca>

James Lewis

Technical Director at Thoughtworks James Lewis, Consultant at ThoughtWorks, Developer, blame me for Microservices

- <https://bovon.org/>
- <https://www.linkedin.com/in/james-lewis-microservices/>
- <https://twitter.com/boicy>

Joel Spolsky

Founder of Stack Overflow, Trello, and Fog Creek Software (now Glitch). Blogger and author. He/him NYC gay techie. Founder of Fog Creek, Trello, Stack Overflow, and Glitch. Check out HASH, an online platform for assembling models of the world <https://hash.ai>

- <https://www.joelonsoftware.com/>
- <https://www.linkedin.com/in/joelspolsky/>
- <https://twitter.com/spolsky>

Jeff Atwood

Indoor enthusiast. Co-founder of <http://stackoverflow.com> and <http://discourse.org>. Let's be kind to each other. Disclaimer: I have no idea what I'm talking about.

- <https://blog.codinghorror.com/>
- <https://twitter.com/codinghorror>

Jeff Sutherland

Founder and Chairman at Scrum, Inc.

- <http://jeffsutherland.com/>
- <https://www.linkedin.com/in/jeffsutherland>
- <https://twitter.com/jeffsutherland>

Jessica Kerr

Developer and symmatheicist. @industriallogic, @greaterthancode, @arresteddevops Tweets are mine, licensed CC0. Following is not endorsement. she/her

- <https://jessitron.com/>
- <https://www.linkedin.com/in/jessicakerr/>
- <https://twitter.com/jessitron>

Jijin P

Fullstack Developer at ClixWall ☒ Maker ☒☒ Developer Bug Hunter
☒☒ React, GraphQL ☒ Exploring Blockchain, #Bitcoin #IndiaWantsCrypto
Working on @CodeKeepIO

- <https://www.linkedin.com/in/pjijin/>
- <https://twitter.com/PJijin>

Jim Highsmith

Author, Agile Project Management, Agilist, Adaptive Leadership, Enterprise Agility, Cyclist

- <http://jimhighsmith.com/>

- <https://www.linkedin.com/in/jhighsmith/>
- <https://twitter.com/jimhighsmith>

John Sonmez

Founder of Bulldog Mindset and Simple Programmer (Do not message me here, visit my site and email me from there) Want to know the best way to increase your salary or hourly rate? <http://devcareerboost.com/blog-course/>

- <https://simpleprogrammer.com/>
- <https://www.linkedin.com/in/johnsonmez/>
- <https://twitter.com/simpleprogrammrr>

Jonathon Morgan

CEO at Yonder (@therealyonder), where we're building a more authentic internet.

- <https://www.goodattheinternet.com/>
- <https://www.linkedin.com/in/jonathonmorgan/>
- <https://twitter.com/jonathonmorgan>

Jurgen Appelo

Entrepreneur, writer, and speaker in the areas of Agility, Innovation, and Leadership. I also read fiction, paint walls, and drink coffee. Successful entrepreneur, Top 100 Leadership Speaker. Newest book: Startup, Scaleup, Screwup.

- <https://jurgenappelo.com/>
- <https://www.linkedin.com/in/jurgenappelo>
- <https://twitter.com/jurgenappelo>

Kamran Ahmed

Engineering Manager at Tradeling.com Lover of all things web and OpenSource. Coding and writing stuff for humans. Instructor @eggheadio and <http://youtube.com/theroadmap>. Building <http://roadmap.sh>

- <https://www.linkedin.com/in/kaamranahmed>
- <https://twitter.com/kamranahmedse>

Ken Schwaber

Co-Founder of Scrum, Head Of Scrum.org, Founder Of The Scrum Alliance

- <https://kenschwaber.wordpress.com/>
- <https://www.linkedin.com/in/ken-schwaber-09a55/>
- <https://twitter.com/kschwaber>

Kent Beck

Programmer, coach coach, singer/guitarist, peripatetic. Learning to be me. Works at @GustoHQ.

- <https://www.kentbeck.com/>
- <https://www.linkedin.com/in/kentbeck/>
- <https://twitter.com/KentBeck>

Kevlin Henney

consultant · father · he/him · husband · itinerant · programmer · keynote speaker · technologist · trainer · writer

- <https://about.me/kevin>
- <https://www.linkedin.com/in/kevin>
- <https://twitter.com/kevinhenney>

Marcello Duarte

Lead Agile Coach at @BTGroup • Best UK Agile Coach @AgileAwards 2014 • #chess

- <https://www.linkedin.com/in/marcelloduarte/>
- https://twitter.com/_md

Mark Richards

Hands-on Software Architect, Author of Fundamentals of Software Architecture (O'Reilly), avid hiker

- <https://www.developertoarchitect.com>
- <https://www.linkedin.com/in/markrichards3/>
- <https://twitter.com/markrichardssa>

Martin Fowler

Author, speaker, and general loud mouth on Software Development. Works for ThoughtWorks. Also hikes, watches theater, and plays modern board games

- <https://www.martinfowler.com/>
- <https://twitter.com/martinfowler>

Michael T. Nygard

SVP, Enterprise Architecture at Sabre Corporation Author of Release It!, speaker

- <https://www.michaelnygard.com/>
- [linkedin.com/in/mtnygard/](https://www.linkedin.com/in/mtnygard/)
- <https://twitter.com/mtnygard>

Neil Patel

Co-founder of Neil Patel Digital. New York Times bestselling author, top 100 entrepreneur under 30 by Obama, and top 10 marketer by Forbes.

- <https://neilpatel.com/>
- <https://www.linkedin.com/in/neilkpatel/>
- <https://twitter.com/neilpatel>

Ric Messier

Consultant, Author, Trainer, Educator, Technologist, General Curiosity

- https://www.amazon.com/Ric-Messier/e/B00I1IWG82/ref=dp_-byline_cont_pop_book_1
- https://www.linkedin.com/in/ricmessier?lipi=urn%3Ali%3Apage%3Ad_-flagship3_profile_view_base_contact_details%3B7Ih%2F5A12RTckDAvTaq7X
- <https://twitter.com/kilroyinco>

Robert C. Martin

Owner, Uncle Bob Consulting LLC. Computer Software Consultant
Software Craftsman

- <http://cleancoder.com/>
- <https://www.linkedin.com/in/robert-martin-7395b0/>
- <https://twitter.com/unclebobmartin>

Ron Jeffries

Ron Jeffries is one of the three founders of the Extreme Programming software development methodology circa 1996, along with Kent Beck and Ward Cunningham. He was from 1996, an XP coach on the Chrysler Comprehensive Compensation System project, which was where XP was invented.

- <https://www.ronjeffries.com/>
- <https://twitter.com/RonJeffries>

Roy Osherove

Pipeline Driven Teams, Crypto & Blockchain, Enterprise DevOps, Continuous Delivery, Team Leadership & Testing Consultant Co-Ops & Pipeline driven organizations, eXtreme Programming, TDD practitioner, elastic leadership. I take pull responsibly for my actions.







































- <https://osherove.com/>
- <https://www.linkedin.com/in/osherove/>
- <https://twitter.com/royosherove>

Sam Altman

Samuel H. “Sam” Altman is an American entrepreneur, investor, programmer, and blogger. He is the CEO of OpenAI and the former president of Y Combinator.

- <http://blog.samaltman.com/>
- <https://twitter.com/sama>

Simon Brown

Author “Software Architecture for Developers” | #c4model & @structurizr for software architecture diagramming | Keynote speaker & trainer |                                      

- <https://simonbrown.je/>
- <https://www.linkedin.com/in/simonbrownjersey/>
- <https://twitter.com/simonbrown>

Sindre Sorhus

Full-time open-sourcerer. Wants more empathy & kindness in OSS. ✂ Swift. Makes macOS apps, CLI tools, npm packages. Made @awesome__re. Follow: @sindre_gh_repos

- <https://sindresorhus.com/>
- <https://www.linkedin.com/in/sindresorhus/>
- <https://twitter.com/sindresorhus>

Steve Denning

Author and Independent Management Consulting Professional The Age of Agile (HarperCollins, 2018) is the definitive guide to managing in a world that is volatile, complex, uncertain & ambiguous <http://a.co/hFeZGck>

- <http://www.stevedenning.com/>
- <https://www.linkedin.com/in/steve-denning-3863355/>
- <https://twitter.com/stevedenning>

Steve McConnell

Author of Code Complete, CEO at Construx Software, Contributor to CDC Ensemble Model for Covid-19 Forecasting #CovidComplete

- <https://stevemcconnell.com/>
- <https://www.linkedin.com/in/stevemcc>
- <https://twitter.com/stevemconstrux>

Tom Preston-Werner

Building @ChatterbugApp & @RedwoodJS. Formerly: GitHub co-founder/CEO. Board @Netlify, @HackClub. Angel investor, pilot. Also: Gravatar, Jekyll, SemVer, TOML.

- <https://tom.preston-werner.com/>
- <https://www.linkedin.com/in/mojombo/>
- <https://twitter.com/mojombo/>

Vasco Duarte

Scrum Master Toolbox Podcast Host, Scrum Master, Agile Coach at Oikosofy Agile, Lean and Scrum Speaker. Author of <http://bit.ly/NoEstimatesBook> Podcast host for <http://bit.ly/ScrumMasterToolboxPodcast>

- https://about.me/duarte_vasco
- <https://www.linkedin.com/in/duartevasco/>
- https://twitter.com/duarte_vasco

Vaughn Vernon

DDD and Reactive Leader; VLINGO development and support; DDD training and consulting Domain Model Whisperer. Champion simplicity & reactive. Author: Implementing #DDDdesign; #DDDdesign Distilled; Reactive #ActorModel. Founder @vlingo_io /PLAT-FORM.

- <https://www.linkedin.com/in/vaughnvernon/>
- <https://twitter.com/VaughnVernon>

Venkat Subramaniam

Founder of Agile Developer, Inc. and creator of agilelearner.com, and dev.next Conference programmer, author, speaker, founder Agile Developer, Inc., creator of <http://agilelearner.com>, co-founder of @devdotnext Conference, professor @CSatUH

- <http://www.agiledeveloper.com/>
- <https://www.linkedin.com/in/vsubramaniam>
- https://twitter.com/venkat_s

Ward Cunningham

Founder, Cunningham & Cunningham, Inc Objects, Patterns, Agile, Wiki

- <https://www.linkedin.com/in/wardcunningham/>
- <https://twitter.com/wardcunningham>

William Durand

Lifelong learner. OSS evangelist. (Traveler.) Cyclist. (Runner.) (Speaker.) Web worker @Mozilla. He/him.

- <https://williamdurand.fr/>
- <https://twitter.com/couac>

Woody Zuill

Senior Consultant, Agile Expertise and Coaching I help teams create an environment where everyone can excel in their work and life. <http://bit.ly/MobProgrammingBook> - visit <http://mobprogramming.org>

- <http://zuill.us/>
- <https://www.linkedin.com/in/woodyzuill/>
- <https://twitter.com/WoodyZuill>

Yegor Bugayenko

Director of System Programming Laboratory at Huawei Lab Director at <http://Huawei.com>; Co-Founder of <http://Zerocracy.com>; Co-Author of <http://ElegantObjects.org>; Co-Creator of <http://Zold.io>

- <https://www.yegor256.com/>
- <https://www.linkedin.com/in/yegor256>
- <https://twitter.com/yegor256>

Zach Holman

angel investor, recovering programmer. @oaklandrootssc investor and fan. ☒ tweets at @zshfc.

- <https://zachholman.com/>
- <https://twitter.com/holman>