Clean Code

Chapter1: Clean Code

author: Hamed Damirchi
hameddamirchi32@gmail.com
github.com/hamed98
linkedin.com/in/hamed-damirchi-ba4085178/

Handbook of agile craftsmanship

- writer is founder of agile method in programming

- *Writing clean code is what you must do in order to call yourself a professional.*
  There is no reasonable excuse for doing anything less than your best.

Foreword

- details are very important in design
- the smallest bit of sloppy construction completely dispels the charm of large whole. that is what clean code is about.
- In software, 80% or more of what we do is quaintly called “maintenance”: the act of repair

5S principles for disciplie(made by jepanees)

- 1- organization (sort) →naming correctly
- 2-tidiness (systematize) → place every thing in its right place, if it is no, refactore it!
- 3-cleaning (shine) keep workplacec clean. get rid of comments that describe what to do in future or previous codes.
- 4- standardization → to follow a standard aggred by group
- 5- discipline(self-discipline) → follow the practice and reflect on others work and willing to change

- You should name a variable using the same care with which you name a first-born child
- designing is never done and is a continues task
- our value system focuses more on outward appearance than on the substance of what we deliver.

Chapter 1: Clean Code

- کد حالا حالا ها با ماست!
- ممکنه زبان ها سطح بالاتر بشن اما ما همیشه با کد سر و کار داریم.

Bad Code

- کمپانی ای که محصول حرفه ای تولید میکرد اما بعد از مدتی ریلیز های جدید طول کشید و باگ های جدیدی در محصول پیدا شد و در نهایت کمپانی شکست خورد. علت آن؟ کارمند آن جا بعد از ۲۰ سال گفت که علتش عجله برای ارائه در مارکت و کد های بی نظم بود. ارائه ی فیچر های بیشتر این بی نظمی را هم بیشتر میکرد
- بسیار شبیه گیر کردن در باتلاق است.
- علت ها: زمان کم، گیر دادن مدیر، بد قول نشدن
- همیشه میگیم بعدا تمیزش میکنم!
- قانون LeBlanc's: بعدا یعنی هرگز

هزینه بی نظمی

- هر تغییردر کد مستلزم فهمیدن گره هایی است که قبلا ایجاد کرده ایم و گره های جدید ایجاد میشود.
- با گذشت زمان بی نظمی بیشتر میشه و دیگه راهی نمیمونه
- بهره وری تیم رفته رفته کمتر میشه و به صفر نزدیک میشه
- مدیریت نیرو جدید میگیره
- نیرو های جدید سیستم رو نمیشناسن و نمیدونن تغییری که میدن در جهت هدف سیستمه یا نه
- همه زیر استرس و فشار هستند که بهره وری رو بیشتر کنن
- ولی خبر ندارن که همه دارن بی نظمی رو بیشتر میکنن
- در نهایت بهره وری به صفر میل میکنه

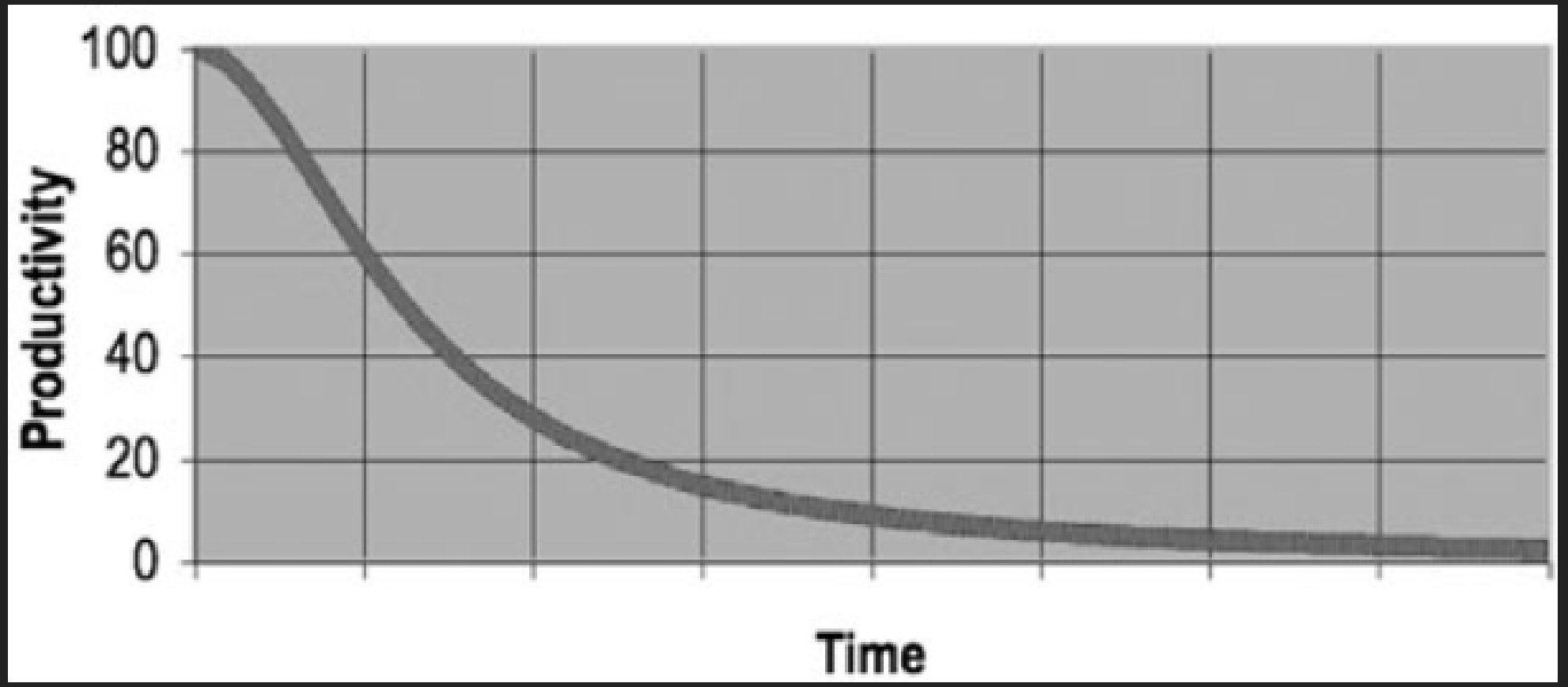


Figure 1-1

Productivity vs. time

هزینه بی نظمی:ریدیزاین عظیم

- برنامه نویس ها درخواست ریدیزاین میدن چون همه چی قفل شده
- مدیریت در نهایت مجبور میشه موافقت کنه
- یه تیم خبره جدید میان که از نو بنویسن و بقیه رو نگهداری سیستم قبلی کار میکنن
- دو تیم شروع به کار میکننو تیم خبره باید برنامه رو طوری بنویسه که کاری که برنامه قبل میکرد رو انجام بده
- تا زمانی که برنامه جدید تکمیل نشده جایگزین نمیشه
- ممکنه مدت خیلی زیادی طول بکشه
- بعد این که اعضای تیم جدید رفتن، اگر این هم بد نوشته شده باشه بازم همین داستان

هزینه بی نظمی: نگرش

- گاها مجبور میشی کاری که باید یه ساعت وقت ببره یه هفته واسش وقت بزاری
- گاها مجبور میشی کاری که باید با یه خط تغییر درست شه صد ها خط رو تغییر بدی
- خیلی وقتا بهونه میاریم، وقت کم بود، فیچر جدید اضافه شد، منیجر نتونست خوب مدیریت کنه و...
- اما مشکل واقعی از خودمونه، خودمون غیر حرفه ای عمل کردیم
- جراحی که مریض بهش میگه بابا نمیخواد دستاتو بشوری، جراح از ریسک نشستن دستش خبر داره ولی مریض نمیدونه!
- بنابراین اگر ازت خواستن کد کثیف بنویسی ولی زود تحویل بدی نباید قبول کنی چون اونا ریسک این کارو نمیدونن

نکته مهم

هیچ وقت به این فکر نکن که بخوای با بی نظم کد نوشتن تو زمان صرفه جویی کنی و سریع به ددلاینا برسی این کار باعث میشه زمان خیلی بیشتری هدر بدی واسه ادامه پروژه و اتفاقا به ددلاینا نرسی

تنها راه رسیدن به ددلاین ها و سریع بودن تو پروژه اینه که همیشه تمیز ترین حالت ممکن کد بنویسی!

هنر کد تمیز نوشتن

- کد تمیز نوشتن سخته!
- کد نوشتن مثل نقاشی کردن میمونه، همه میتونیم نقاشی خوب رو از بد تشخیص بدیم ولی این به این معنی نیست که میتونیم نقاشی خوبی بکشیم
- برنامه نویسی که code-sense داره میتونه کد کثیف رو تبدیل به کد تمیز کنه. (؟) وگر نه فقط میتونه تشخیص بده کد کثیف رو

کد تمیز چه ویژگی هایی داره؟

از زبان مخترع c++ :
راحت بشه خوند و ساده باشه و پیچیده نباشه
راحت بشه توش باگ ها رو پیدا کرد
وابستگی اجزا مختلف به حداقل برسه
error handling توسط یه استراتژی از قبل مشخص شده اجرا بشه
بیشترین پرفورمنس ممکن رو داشته باشه
جوری نباشه که تمایل به بی نظمی رو بیشتر کنه(هنگام تغییر)

- یک پنجره شکسته تو یه ساختمون باعث میشه افراد به بقیه پنجره ها هم اهمیت ندن و بقیه هم بشکنه(اثر کد کثیف)

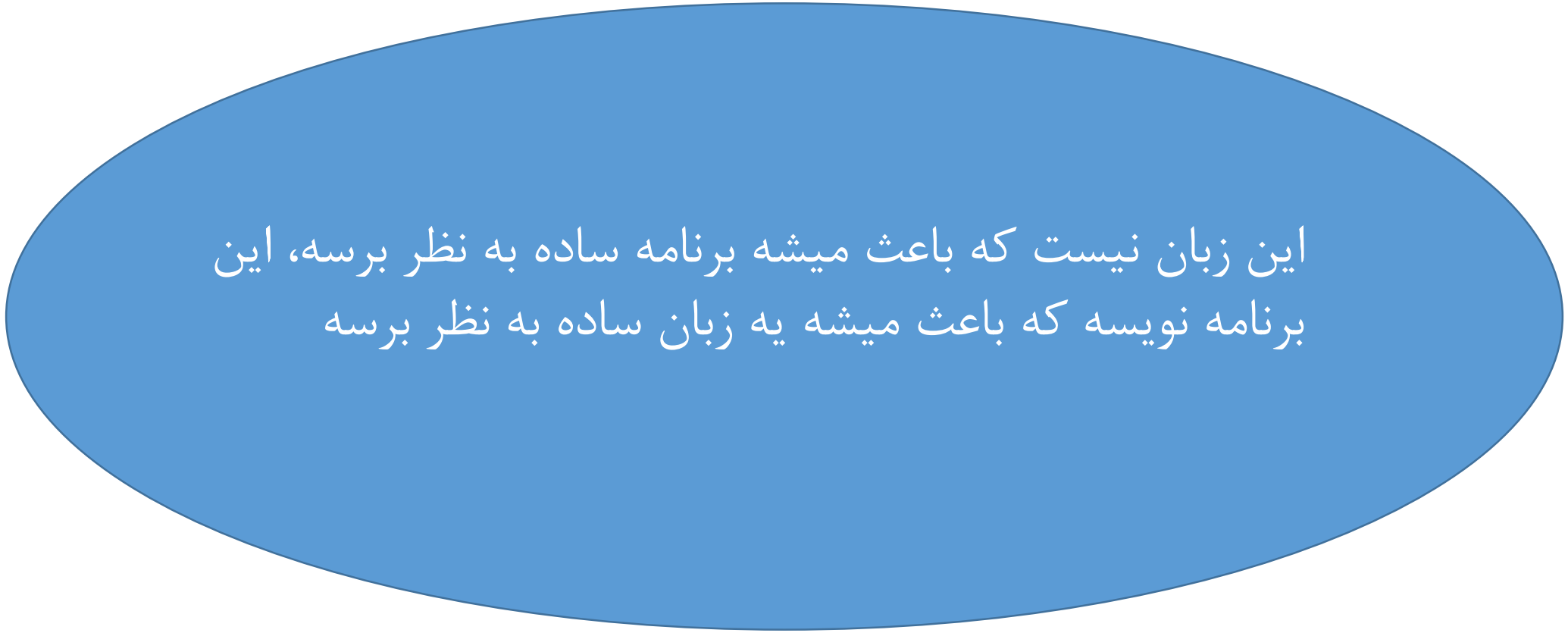
کد تمیز چه ویژگی هایی داره؟

از زبان گریدی بوش، نویسنده کتاب شی گرایی.
ساده است و هدف توش مشخصه
مشخصه که برنامه نویس چه هدفی داشته، دقیقا داره چی کار میکنه(دقیقا مثل یه کتابی که خوب نوشته شده باشه)
از انتزاع به نحو احسن استفاده کنه

بقیه ویژگی های کد تمیز که افراد مطرح ذکر کردن:

- توسط یه برنامه نویس دیگه به راحتی خونده میشه و به راحتی میتونه تغییر و بهبود بده کد رو
- یونیت تست داره (TTD)
- نام گذاری صحیح و با معنا
- برای انجام یک کار تو برنامه ، فقط یک راه وجود داشته باشه (دو تکه کد جدا یه کار واحد رو انجام ندن)
- اجزای مختلف حداقل وابستگی رو داره
- مینیمال باشه (ترجیح تو کد کمتره)
- کدی که تست نداره هر چه قدر هم خوانا باشه به هیچ وجه تمیز نیست!
- کدیه که وقتی میخونیش میفهمی برنامه نویس به کد اهمیت داده (care)
- هیچ کد تکراری نداره

- هر بخش کد دقیقا یک کار واحد رو انجام بده. اگر یه تابع دو کار انجام میده باید تبدیل بشه به دو تابع جدا از هم.
- استفاده صحیح از abstraction
- وقتی به کد تمیز نگاه میکنی جوریه که انگار اون زبون واسه اون کار ساخته شده!



سخن نویسنده

- هیچ روش "درست" توی کد تمیز نوشتن نیست، مثل ورزش های رزمی که درست و غلطی نداره
- این مطالب حاصل ده ها سال تجربه است
- قول میده که با خوندن این کتاب و عمل به توصیه هاش، بتونید حرفه ای تر بشید!

- ادیتور emacs که دهه ۸۰-۹۰ بود، همه چیز رو ثبت میکرد، هر کلیدی که فشرده میشد حتی!
- چیزی که ضبط کرد: بیشتر زمان واسه اسکرول کردن و navigate کردن بین ماژول های مختلفه.

Bob enters the module.
He scrolls down to the function needing change.
He pauses, considering his options.
Oh, he’s scrolling up to the top of the module to check the initialization of a variable.
Now he scrolls back down and begins to type.
Ooops, he’s erasing what he typed!
He types it again.
He erases it again!
He types half of something else but then erases that!
He scrolls down to another function that calls the function he’s changing to see how it is
called.
He scrolls back up and types the same code he just erased.
He pauses.
He erases that code again!
He pops up another window and looks at a subclass. Is that function overridden?

…

- بنابراین بیشتر وقت ما به خوندن کدای قبلیمون میگذره. چون باید بدونیم قبلا چی نوشتیم که الان بخوایم یه چیزی بهش اضافه کنیم
- یکی از هدفامون اینه که این خوندن کد رو آسون تر کنیم

boy scout rule

- **leave the campground cleaner than you found it**
- اگر این قاعده رعایت شه همیشه کد تمیز میمونه
- یعنی هر وقت یه بی نظمی دیدیم سریع درستش کنیم