معماری تمیز

Clean Architecture

راهنمای استادی در طراحی و ساختار نرم‌افزار

ترجمه و تالیف

مهرداد عباسی - سید مرتضی قائم‌مقامی

سرشناسه: مارتین، رابرت سی. Martin, Robert C.

عنوان و مترجمین: معماری تمیز: راهنمای استادی در طراحی و ساختار نرم‌افزار / [تالیف: رابرت سی مارتین]؛ ترجمه و تالیف: مهرداد عباسی، سیدمرتضی قائم‌مقامی.

» **مشخصات نشر:** اصفهان، موسسه تولید علم فرزانگان برنا، ۱۳۹۸.

» **شابک:** ۹۷۸-۶۰۰-۸۲۰۲-۸۰-۶ » **مشخصات ظاهری:** ۴۰۶ص، وزیری

» **وضعیت فهرست‌نویسی:** فیپا

» **یادداشت عنوان اصلی**

» Clean architecture : a craftsman's guide to software structure and design,2018.

» **عنوان دیگر:** راهنمای استادی در طراحی و ساختار نرم‌افزار

» **موضوع:** معماری نرم‌افزار » **موضوع:** Software architecture

» **موضوع:** نرم‌افزار -- تولید » **موضوع:** Computer software -- Development

» **موضوع:** Computer programming--History.

» **شناسه افزوده:** عباسی، مهرداد، ۱۳۶۵، مترجم

» **شناسه افزوده:** قائم‌مقامی، سید مرتضی، ۱۳۶۶-، مترجم

» **رده‌بندی کنگره:** QA ۷۶/۷۶ » **رده‌بندی دیویی:** ۰۰۴/۰۲

» **شماره کتابشناسی ملی:** ۵۸۹۳۹۰۲

عنوان: معماری تمیز: راهنمای استادی در طراحی و ساختار نرم‌افزار

ترجمه و تالیف: مهرداد عباسی - سید مرتضی قائم مقامی

صفحه‌آرایی: زهرا مسائلی

طراح جلد: منیژه عباسی

تیراژ: ۵۰۰ تیراژ

نشانی ناشر: اصفهان، خیابان بزرگمهر، خیابان هشت بهشت غربی، مجتمع فرهنگی، طبقه سوم، واحد ۳۰۱

تلفن انتشارات: ۴۰۲۶۶ ۰۹۱۳

تلفن فروش: ۰۹۱۳۳۳۷۴۶۷

چاپ اول: ۱۳۹۸

موسسه نشر تولید علم فرزانگان برنا

Nashrtolidelm@gmail.com

قیمت: ۱۴۰۰۰۰ تومان

سخن مترجمین

کتاب پیش روی شما ترجمه کتاب Clean Architecture اثر Robert C. Martin (معروف به عموباب) می‌باشد. این کتاب شامل چند دهه تجربه برنامه‌نویسی و چگونگی رفتار در پروژه‌های مختلف برنامه‌نویسی است و خواندن آن برای ورود به دنیای حرفه‌ای برنامه‌نویس‌ها توصیه می‌شود. فصل‌های کتاب شامل موضوعاتی مانند معماری، برنامه‌نویسی ساخت‌یافته، برنامه‌نویسی شی‌گرا، برنامه‌نویسی تابعی، اصل SRP، اصل OCP، اصل LSP، اصل DIP، قوانین کسب و کار و سرویس‌ها است.

خوانندگان عزیز توجه داشته باشند که در راستای فهم و درک بهتر مطالب، برخی مثال‌ها به گونه‌ای ترجمه شده است، تا کاربران فارسی زبان راحت‌تر بتوانند مطلب را درک کنند.

با توجه به دقت فراوانی که در جهت ارائه ترجمه مناسب و قابل فهم برای شما خوانندگان عزیز صورت گرفته است، ممکن است اشکالاتی در بعضی کوچک نیز وجود داشته باشد. لذا از تمام عزیزان خواهشمندیم که در صورت داشتن هر گونه پیشنهاد یا انتقاد جهت اصلاح یا بهتر شدن این کتاب، از طریق سایت **zerobook.ir** ما را از نظرات خود مطلع فرمایید.

در پایان خداوند را سپاسگزاریم که توفیق به انجام رساندن این کار را به ما اعطا کرد و به امید آنکه شما خوانندگان گرامی در سایه الطاف خداوند متعال همواره سلامت و موفق باشید.

مهرداد عباسی

MehrdadLinux@gmail.com

سید مرتضی قائم مقامی

mortezaghaem@gmail.com

فهرست مطالب

فهرست مطالب

فهرست مطالب

فهرست مطالب

فهرست مطالب

عنوان | شماره صفحه

فهرست مطالب

فهرست مطالب

فهرست مطالب

فهرست مطالب

عنوان | شماره صفحه

پیشگفتار

وقتی در مورد معماری صحبت می‌کنیم، در مورد چه چیزی صحبت می‌کنیم؟

مانند هر توصیفی، توصیف نرم‌افزار از دیدگاه معماری به همان اندازه که گویا است، مبهم هم است. در معماری می‌توانید بیش از حد توان خودتان قول انجام کاری را بدهید که در نهایت قادر به تحویل آن کار باشید و در شرایط دیگر می‌توانید بیش از آنچه قول داده‌اید را تحویل دهید. زیبایی معماری در ساختار آن است و ساختار چیزی است که پارادایم‌ها[1] و بحث‌های مربوط به توسعه نرم‌افزار مانند کامپوننت‌ها، کلاس‌ها، توابع، ماژول‌ها، لایه‌ها و خدمات، میکرو یا ماکرو را به خود اختصاص داده است. اما ساختار بدشکل بسیاری از سیستم‌های نرم‌افزاری معمولاً برعکس باور یا درک ما هستند، مانند طرح‌های سازمانی بسیاری از اتحادیه جماهیر شوروی، بلوک‌های چوبی بازی جنگا[2] که رسیدنشان به ابر غیر محتمل است، بخش‌های قدیمی از یک سیستم نرم‌افزاری که معماری قابل فهم ندارد[3] و فراموش شده‌اند. واضح نیست که ساختار نرم‌افزار به همان روش ساختار ساختمان، از برداشت ذهنی ما پیروی می‌کند.

ساختمان‌ها دارای یک ساختار فیزیکی هستند، خواه ساختمان پایه و اساس آن‌ها در سنگ باشد یا در بتون، خواه ساختمان‌ها دارای یک طبقه بلند باشند (چند طبقه و بلند) یا فضای سرتاسری و جادار داشته باشند (یک طبقه و دارای فضای بزرگ و وسیع)، خواه ساختمان‌ها بزرگ باشند یا کوچک، چه مجلل باشند یا بی‌زرق و برق. ساختار ساختمان‌ها انتخاب‌های کم و محدودی هستند، اما همه این ساختارها مطابق با قوانین فیزیکی گرانش و قوانین فیزیکی موادشان هستند. از سوی دیگر، در مورد نرم‌افزار، زمان کمی برای مطابقت با قوانین طراحی اصلی، صرف می‌شود البته به جز در مواردی که نرم‌افزار دارای اهمیت باشد. نرم‌افزار از چه چیزی ساخته شده است؟ برخلاف ساختمان‌هایی که ممکن است از آجر، بتن، چوب، فولاد و شیشه ساخته شوند، نرم‌افزار

---

1. Paradigms

۲. بازی جنگا Jenga از ۵۴ بلوک چوبی ساخته شده است که طول هر بلوک باید دقیقاً سه برابر عرض و ضخامتش هم باید یک پنجم طولش باشد. برای شروع بازی جنگا Jenga باید بلوک‌های چوبی رو به صورت افقی و عمودی روی هم بچینید تا یک برج ۱۸ طبقه چوبی ساخته بشود. بعد از اینکه برج چوبی تکمیل شد، بازیکنی که برج را چیده باید بازی را شروع کند. روش بازی به این صورت است که هر بازیکن به نوبت باید یک بلوک چوبی را از هر جایی غیر از دو ردیف اول (بالای برج) بردارد. هر بازیکنی که برج را خراب کند، باخته است! «مترجم»

۳. big ball of mud یک سیستم نرم‌افزاری است که معماری قابل فهم ندارد. اگر چه از دیدگاه مهندسی نرم‌افزار ناخوشایند است، اما چنین سیستم‌هایی به دلیل فشارهای تجاری، بازده توسعه‌دهنده و آنتروپی کد، در عمل کاربرد دارند. آن‌ها یک نوع ضد الگوی طراحی هستند. «مترجم»

از نرم‌افزار ساخته شده است. ساختارهای نرم‌افزاری بزرگ از کامپونت‌های نرم‌افزاری کوچک‌تر ساخته می‌شوند که به نوبه خود همچنان از کامپونت‌های نرم‌افزاری کوچک‌تری ساخته شده‌اند و همین‌طور ادامه می‌یابد. کدنویسی می‌تواند شبیه یک سلسله مراتب وابسته به یکدیگر باشد.

زمانی که در مورد معماری نرم‌افزار صحبت می‌کنیم، نرم‌افزار ذاتاً بازگشتی و فراکتال است، طرح نرم‌افزار در کد ریخته می‌شود. همه چیز مربوط به جزئیات است. سطوح در هم تنیده جزئیات، در معماری ساختمان نیز وجود دارد، اما صحبت کردن در مورد واحد سنجش فیزیکی، در نرم‌افزار بی‌معنی است. نرم‌افزار دارای ساختارهای بسیاری است - حتی انواع بسیاری از ساختارها را دارد- با این حال تنوع ساختاری نرم‌افزاری، بیشتر از ساختارهای فیزیکی موجود در ساختمان‌ها، می‌باشد. شما حتی می‌توانید به راحتی استدلال کنید که در معماری نرم‌افزار نسبت به معماری ساختمان، فرآیندهای طراحی و تمرکز بیشتری وجود دارد، بدین معنا که غیرمنطقی نیست وابستگی به معماری در معماری نرم‌افزار، بیشتر از وابستگی به معماری در معماری ساختمان باشد.

اما مقیاس فیزیکی چیزی است که انسان‌ها در جهان می‌فهمند و به دنبال آن هستند.

اگر چه از لحاظ دیداری جذاب است، اما مستطیل‌های موجود در یک نمودار پاورپوینت، معماری یک سیستم نرم‌افزاری نیستند. شکی نیست که آن‌ها معماری را از یک دیدگاه خاص نمایش می‌دهند، اما اشتباه گرفتن این مستطیل‌ها با تصویر کلی[1] برای معماری، از دست دادن تصویر کلی در معماری است (اگر دچار چنین اشتباهی شویم، مفهوم معماری را به خوبی درک نکردیم). معماری نرم‌افزار شبیه هیچ چیز نیست. نشان دادن با شکل‌ها، تنها یک انتخاب است، یک روش مشخص و معلوم. این کار یک انتخاب براساس یک مجموعه از انتخاب‌هاست: (انتخاب‌هایی مانند اینکه) باید شامل چه چیزی باشد، باید چه چیزی را حذف کرد، با استفاده از شکل یا رنگ روی چه چیزی باید تاکید کرد، از طریق هم شکلی یا نادیده گرفتن روی چه چیزی نباید تأکید کرد. هیچ چیزِ ذاتی یا طبیعی در مورد برتری یک دیدگاه نسبت به دیگری وجود ندارد.

هر چند که ممکن است صحبت کردن در مورد فیزیک و مقیاس فیزیکی در معماری نرم‌افزار بی‌معنی باشد، ما نگران برخی محدودیت‌های فیزیکی خاص هستیم. سرعت پردازنده و پهنای باند شبکه می‌توانند تاثیر شدیدی بر بازدهی و عملکرد نرم‌افزار داشته باشند. حافظه و رسانه ذخیره‌سازی می‌توانند

1. Big picture

زیاده‌خواهی‌های هر code base[1] را محدود کنند. نرم‌افزار می‌تواند ابزاری باشد که ما با استفاده از آن رویاهایمان را عملی می‌کنیم، اما در نهایت در جهان فیزیکی اجرا می‌شود و محدود است.

این شرارت و بی‌عاطفگی در عشق است، بانو، که آرزو بی‌نهایت است و اجرا محدود است؛ میل بی‌حد و حصر است و عمل یک برده محدود است.

– قسمتی از نمایش‌نامه تراژیک تروئیلوس و کرسیدا (انگلیسی: Troilus and Cressida) اثر ویلیام شکسپیر Act III. Scene II.

جهان فیزیکی جایی است که ما و شرکت‌هایمان و اقتصادهایمان در آن زندگی می‌کنند. این یک درجه‌بندی دیگر بر ما می‌دهد که ما می‌توانیم معماری نرم‌افزار را با استفاده از آن درک کنیم، به جای دیگر نیروها و کمیت‌های کمتر فیزیکی که ما می‌توانیم از طریق آن‌ها صحبت و استدلال کنیم.

معماری، تصمیمات مهم طراحی را نشان می‌دهد که یک سیستم را شکل می‌دهد، در این جا مهم بودن، با هزینه تغییر، اندازه‌گیری می‌شود.

– گریدی بوش

زمان، پول و تلاش (نیروی انسانی)، درکی از مقیاس به ما می‌دهند که برای تمایز قایل شدن بین بزرگ بودن و کوچک بودن نرم‌افزار و امکان تشخیص ویژگی‌های معماری از چیزهای دیگر را برای ما فراهم می‌کنند. این اندازه‌گیری همچنین به ما می‌گوید که چگونه می‌توانیم مشخص کنیم که آیا معماری خوب است یا خیر. یک معماری خوب نه تنها نیازهای کاربران، توسعه‌دهندگان و مالکان را در یک نقطه معین از زمان برآورده می‌کند، بلکه با گذشت زمان نیز نیازهای آن‌ها را برآورده می‌کند.

اگر فکر می‌کنید معماری خوب گران است، معماری بد را امتحان کنید.

---

۱. در توسعه نرم‌افزار، codebase یا (code base) به کل مجموعه‌ای از کد منبع اشاره می‌کند که برای ساخت یک سیستم نرم‌افزاری، نرم‌افزار یا component نرم‌افزاری خاص استفاده می‌شود. به طور معمول، codebase شامل تنها فایل‌های کد منبعی نوشته شده توسط انسان است. بنابراین، codebase معمولا شامل کد منبع ایجاد شده از طریق ابزار (فایل تولید شده) یا کتابخانه‌های دودویی (object files) نیست، زیرا آن‌ها می‌توانند از source code اولیه نوشته شده توسط انسان ساخته شوند. با این حال، به طور کلی شامل پوشه‌های configuration و property files می‌شود، زیرا آن‌ها اطلاعات لازم برای ساخت هستند. «مترجم»