B5 Cover Page

(182x257mm or 516x728pt)

アイのムチ

— レビューで折れないメンタルづくりとレビュアー側の心得 —

［著］ACCESS 技術書典同好会

技術書典 12（2022 年冬）新刊
2022 年 1 月 22 日　 ver 1.0

■免責

本書は情報の提供のみを目的としています。
本書の内容を実行・適用・運用したことで何が起きようとも、それは実行・適用・運用した人自身の責任であり、著者や関係者はいかなる責任も負いません。

■商標

本書に登場するシステム名や製品名は、関係各社の商標または登録商標です。
また本書では、™、Ⓡ、Ⓒ などのマークは省略しています。

まえがき

「こんにちは、アメです！ 」

「みなさん、普段レビューしてますか？ 」

「見る側の方も、見てもらう側の方も、両方の方もいますよね。」

「マリアです。プロジェクトマネージャーをしています。」

「私のチームでは、Git で開発とコードレビューをします。」

「それも長くやってると…色々あるよね、アメさん？ 」

「そうなんです。」

「レビューって、人によって気にする点や出すときの心理が違いますよね。」

「そんなレビューに楽しく前向きに取り組むため、私たちのチームで取り組んだことを紹介します。」

登場人物

エマちゃん

- エンジニア 1 年生
- 映画とお笑いと甘いものを食べるのが大好き、睡眠は 1 日 9 時間

アメさん

- ゆるふわだけどサバサバなエンジニア（※要相談）
- こだわりは少ないほうで、部屋は物が増えないタイプ

ムチさん

- 使うものにはとことんこだわりたい寡黙なエンジニア
- 意外とスポーツマン。スキューバダイビングに行く

マリアさん

- 5 年前にエンジニアから転向したプロジェクトマネージャー
- スプレッドシートより Excel。Slack のバッヂはすぐ消したい派

目次

第 1 章
よくないレビューの例

本章では、よくないレビューの例をいくつか挙げます。ただし、本書で取り上げるよくないレビューとは、人間関係を悪くしたり、不必要な悩む時間を生んだり、レビューに出す人のメンタル状態やモチベーションを下げるようなレビューのことを指します。

1.1 審査員形式

エマちゃんは、チームに配属されて 1 ヶ月の新人さんです。勉強期間も終わって徐々に商用アプリ開発を任されるようになり、一生懸命やってプルリクエストを出しました。

「実装終わったのでレビューお願いします！ 」

「アメさん、見てあげれる？ 」

「ちょっと今忙しくて…。ムチさんに見てもらってください。」

「わかりました！ 」

「（アメさんがよかったなぁ…。ムチさんちょっと怖いんだよな…）」

こういうシーン、みなさんのプロジェクトでも心あたりあるかもしれません。なぜ、エマちゃん

はアメさんにはお願いしやすくて、ムチさんにはお願いしにくいのでしょうか。

おや、そう言ってる間にムチさんからレビューコメントが返ってきたようです。

▼リスト 1.10: @ Muchi commented

```
NGです。レビュアーやテスターに伝わる説明を書く努力をしてください。
冗長コードも多すぎます。既存の全体設計を事前に見なかったのでしょうか。
ここはif-elseの判定が仕様と逆ですが、わざと破壊しようとしていますか？
```

～翌日～

「（なんか体が重いなぁ。今日は午前休しよう…）」

「あれっ、エマちゃん今日どうしたの？」

「午前休って連絡きてましたよ。」

「あ、ホントだ。体が重いって言ってるのが気になるね。心当たりは？」

「昨日、ムチさんのレビューを受けて、遅くまで修正していたようです。」

「なるほど。何かあったのかな。レビュー記録を見てみよう。」

「うーん、厳しい指摘が多いみたいだなぁ。スケジュールの心配もあるし、ムチさんに話を聞いてみよう。」

エマちゃんが午前休を取ってしまったのは、甘えやメンタルが弱いからでしょうか。そう思われては気の毒ですよね。エマちゃんは自分なりによいものを作ろうと一生懸命取り組んでいたはずです。

マリアさんはムチさんを呼んで、コメントの真意を聞いてみました。

「ムチさん、エマさんの実装だけど、何か大きな問題がある？」

「いえ、そこまでは。」

「レビューの指摘が結構重たいみたいだけど。」

「あれは、少し勉強不足を感じたので、自分で考えて成長してほしいと期待しました。」

ムチさんの言い分もよくわかります。甘いだけではやっていけないのがプロの世界ですからね。

ですが、エマちゃんはやる気を削がれ、望まない理由で有休まで消費してしまいました。これはエマちゃんと会社の双方にとって、非常にもったいないことですよね。

では、このレビューはどうすればよかったのでしょうか。

■審査員形式のレビューとは

- 複数のレビュアーが辛口審査員のように振る舞う
- 発表者のコードを攻撃的に批判する

今回は、批評者はムチさんだけですが、エマちゃんが努力不足だったりよく考えずにやっている、さらに悪意があって間違えたなどと批判するかのようなコメントをしています。

これは発表者のメンタルを壊し、チームには鬱憤とやる気の低下が残るのみなのです。

「（とはいえ、ムチさんは元々そういうスタイルの人で、悪気はないから注意するのもなぁ…。そうだ、アメさんにも見てもらおう）」

「アメさん、このレビュー指摘だけど、アメさんからもフォローしてくれない？ 」

「はーい。」

「（ふむふむ、なるほど。エマちゃんはこれに悩んでたのか…。私ならこう書くかな）」

▼リスト 1.2: @Ame commented

ご対応ありがとうございます。実装箇所は合ってますが、いくつかお願いしたいことをMust/Wantで伝えますね。

Must: if-elseを仕様通りに直す
Want: 類似実装の共通化、誤りを未然に防ぐための単体テスト作成

説明文はフォーマットを使うとよいと思いますよ。
わからないところは過去のPRを参考にしてみてください。

「おはようございます〜。今朝はすみませんでした〜。」

「おはよう。大丈夫？ ムチさんの指摘がちょっと怖かった？ 」

「あっ、そうじゃないんです。でも、直しててなんか辛くなっちゃいました…。」

「そういうときは、抱え込まず相談してね。私も最初、ムチさんにあんな感じでしごかれたのよね。」

「えっ、そうなんですか！ 意外です。」

「私のほうが後から入ったのもあるけど、あの人、新人だから厳しいんじゃなくて、誰に対してもそうなのよね。」

「でも、ムチさんはきっとこう伝えたかったと思うよ。」

それからアメさんは、「私ならこう書く」と考えたコメント通りエマちゃんに伝えました。それを聞いて、エマちゃんはやる気を取り戻しました。

アメさんのコメントは審査ではなく、実装者に寄り添うような提案をしていますね。良かった部分は認め、改善点は必須と任意を分けてヒントを出すに留め、実装者自身が考えて学ぶ余地も残しています。

エマちゃんは 1 年目で 1 つ 1 つの達成を自信に変えている段階なので、それを後退させるフィードバックはチーム力の後退にも繋がります。

気持ちのよいフィードバックとは、必ずしも全員一緒ではないのですが、コミュニケーションを

重ねる中で、適切なバランスを見つけていきましょう。

1.2 試験形式

「実装終わったけど今日はアメさん不在だし、期限が近いからムチさんに出さなきゃ…。今度こそ何も言われなければいいな！ 」

▼リスト 1.10: @ Muchi commented

```
NGです。このAPIが例外を吐いたときの考慮がされていません。
typoがいくつかあるし、インデントも不揃いです。全体的に不合格。
```

「ぐすん…不合格なんて…。コンビニスイーツで元気になろう…。」

ある日の朝会の後、マリアさんはエマちゃんを呼び止め、こう問いかけました。

「エマちゃんも入って 1 ヶ月経ったね。何か困ってることある？ 」

「私はどうすれば 1 人前になれるでしょうか…。」

「というと？ 」

「こないだムチさんにレビューで不合格と言われちゃって、早く合格して皆さんのお役に立ちたいんです。」

「う〜ん、不合格って言い方は厳しいなぁ…。またムチさんに真意を聞いてみようかな。」

ところがムチさんは席にいなかったので、マリアさんはアメさんのところに行き、経緯を説明しました。

「ということがあってね、アメさんはどう思う？ 」

「エマちゃんは既に私たちのメンバーですし、私は不合格などと評価したりしないですよ。」

「エマちゃんは私からフォローしますね。」

「いつもごめんね、アメさん。」

■試験形式のレビューとは

・レビューの最後に「合格」「不合格」みたいな用語が飛び交う
・タイプミスや改善点を見つけて指摘することを、失敗の通達と捉えている

これは知識が浅い人やニューフェイスがいると起こりがちですが、エマちゃん、ムチさん、アメさん、マリアさんの 4 人は、立場こそ違えど「ソフトウェアを完成させ、ユーザーを満足させる」という目的は一緒なはずです。

その中で試験のようなレビューをすると、発表者はユーザーではなくレビュワーの満足度を高くすることに注力したり、試験に合格することが目的と錯覚します。そして試験に落ちるとやる気を損ない、協調的であるべきメンバーの和が、いつの間にか乱れてしまうのです。

「エマちゃん、それおいしい？ 」

「あ、これですか！ この前落ち込んでるときコンビニで見かけた新作のしらたまあんぱんです！ なんかホッとする味です！ 」

「いいね。私も食べてみようかな。ところで、この前レビューしてた期限の近いあれ、どうなった？ 」

「時間がなかったのでムチさんに続きを引き取ってもらいました…。」

「えーん、思い出したら悲しくて、もう 1 個食べたくなっちゃいました〜。買ってきます〜。」

「待って待って！ あれね、実は私も途中まで見てコメントしかけてたの。これ見て。」

▼リスト 1.10: @ Muchi commented

```
1．このAPIが例外を起こしたときの考慮
2．typoやインデントずれ
これらを直していただければ、残りはLGTMです。
```

「（LGTM…そうだったんだ。よかった…。間違えたところは、次は気をつけよう）」

それから、エマちゃんのしらたまあんぱんブームも終わり、気になってた体重もそれほど増えずに済んだのでした。

アメさんがエマちゃんに見せたコメントは、評価などの不必要な言及を避け、そのとき必要な指摘だけを伝えるものでした。そして、やはり良かった部分は認めています。

何かを書いて校正を頼むのは試験ではなく、同じ目的を持つ仲間同士の協力作業なんです。

1.3 意識高い系

それから 1 ヶ月経ち、エマちゃんは困ったことは何でもアメさんに相談し、順調にスキルアップしていきました。

そして、エマちゃんは少し大きな実装を任されました。この間勉強したばかりのアーキテクチャーやデザインパターンを駆使し、わからないところはアメさんに質問しながら、なんとか仕上げました。

「アメさん、色々とありがとうございました！」

「いえいえ。早くムチさんからも Approve をもらえるといいね！」

「（今回はアメさんの言う通りに作ったし、ムチさんも納得してくれるはず…！）」

エマちゃんはルンルン気分で PR を作り、退社しました。水曜日だし、ウェンズデーを活用して前から見たかった映画を見たり、ケーキバイキングに行ったりして、オフをエンジョイしました。

翌日、出社するとムチさんの指摘が来ていました。

▼ リスト 1.10: @ Muchi commented

```
この判定はビジネスロジックなので、サービスクラスでやる仕事じゃないです。もっとドメインを意識して書き直して下さい。
doTask()に2つ以上の責務があり意味不明です。
```

「(ぐす…アメさんに色々聞いて頑張ったのに書き直しだなんて…)」

「エマちゃん、Approve もらえた？」

「アメさーん！（泣）」

「ど、どうしたの！？ 」

アメさんは泣くエマちゃんを必死になだめました。その日の午後、アメさんがマリアさんに呼ばれました。

「エマちゃんと 1on1 をしたんですが、ちょっと自信を失ってるようだね。」

「はい、私がレビューして OK だったんですが、ムチさんにとっては NG だったようで…。」

「レビューのチェックポイントは 2 人とも同じなんだよね？ なんで分かれたの？ 」

「私はそこまで関心の分離を徹底しようと思ってなかったんですが、ムチさんは徹底したかったようです。」

「それに、エマちゃんは何も考えてないわけじゃなく、エマちゃんなりの考えでこういう記述してるんだなって感じて、尊重してあげたかったんです。」

「だから、今は OK を出して、高度な実装は追々学んでくれるといいな〜と思ったのが、私側の感覚ですかね。」

「なるほどね。アメさんから見て、ムチさんのやり方はどう思う？ 」

「厳しめに見てくれるのはありがたいんですが、頑張って書いたのに突き放すような言い方はよくないかなって思いますね。」

■意識高い系のレビューとは

- 突き放す感じの指摘で、具体的な修正指示が含まれていない
- タイプミスや改善点を見つけて指摘することを、失敗の通達と捉えている

どんな実装にも、レビュイーにとってはベストの実装だったり、レビュイーなりの考えがあるこ

とが多いので、否定だけはよくありません。具体的な修正指示や例を出してあげる方が親切です。

　また、答えをそのまま教えればよいとも限りません。あくまで「例」を示し、レビュイーが自分で考え、答えはネットや本で見つけるほうが、スキルアップやモチベーションに繋がることがあります。

（ここで 4 コマ漫画を挿入したい。ストーリーは、マリアさんがみんなをランチに誘ったらムチさんが「大丈夫です」と断る。「大丈夫ですってなにー！？」ってツッコミが入る）

1.4 持久戦

さて、レビューを受けるのはエマちゃんだけではありません。

マリアさんとアメさんは、今後のプロジェクト計画についてプロジェクト責任者やステークホルダーたちと対面でのレビューを受けました。しかし、その会議は予定終了時刻を 2 時間もオーバーした上、いくつかの議論は行われませんでした。

「遅くまでお疲れ様。」

「あの方々とやると、いつも予定時間を超えますよね〜。」

「ホントにね、やんなってくる。」

「それに最初は段取り良かったけど、途中から同じ話の繰り返しでした。」

「終わりのほうは、向こうも疲れてたしね。」

「しかも結論がいくつか出てないと思うので、別の日に再レビュー設定することになりそうです。」

「最初の 1 時間で切り上げてもそんなに変わんなかったよね。」

「決めた予定だから最後までやりたかったんでしょうね。」

「今頃向こうは、やった！ って満足感だけ残ってるんじゃない？ 」

実はこのレビュー、向こうも満足感ではなく、マリアさんと同じようなことを感じ、愚痴っていたのでした。

つまり、誰か特定の人の振る舞いが良くなかったわけではなく、お互いにこのまま続けると非効率とわかっていながら、引っ込みがつかなかったのです。

■持久戦のレビューとは

- やたらと長く、予定を超過する
- 最初はやる気があっても、時間が経つと疲れる
- 発表者もレビュアーも、時間が経つにつれ不注意になるが、切り上げられない
- 体力を吸われ、レビューの精度も低くなる
- 充実感を伴って終わり、本質的な問題に気づかない

とはいえ、対面での議論を途中で切り上げるのは、思い切りの要ることです。スケジュール調整も必要なので容易ではありません。

最初に「この場ではこれだけ話す！」「議題が逸れたらチャットツールやチケットなど、非同期な方法で続きを議論する」など、あらかじめ合意しておくことが重要でしょう。

1.5 行き当たりばったり

エマちゃんたちの会社は、360 度評価が採用されています。今日はムチさんがメンバーからの評価フィードバックを受ける日です。

「ムチさんのプロジェクト貢献度、コードの質の良さ、ドキュメントの更新頻度は高い点数だったよ。」

「はい。」

「良い点は、ベースの知識の豊富や、解決力、実装の早さなどだって。私もその通りだと思う。」

「はい。」

「でもね、改善点として、円滑なコミュニケーションがややしづらい、他の人との交流が少なく話しかけづらいって書かれてたのが気になるね。」

「なるほど。」

「改善点について、何か思うことは？ 」

「特に…ないです。」

「（ムチさん、毎回こうなんだよな〜。本心が読めない…）」

席に戻ったムチさんは、さっさと帰り支度を始めました。

実はムチさん、帰りに書店でインテリア雑誌を立ち読みし、良い本なら買うのが日課です。今日も観葉植物を替えたくて新刊を読んでいますが、どうにも落ち着きません。360 度評価をきっかけに、過去数ヶ月のことが気になり始めたようです。

「（今まで通りやってるはずなのに、新人が配属されてからなんかペースが狂うなぁ…）」

「（アメさんが人によって態度やレビューの許可基準を変えるから、僕がうまくフォローしてるつもりなんだけど）」

「（このままレビュー品質が担保されなくなる前に、PM に警告しとこうかな）」

翌日、ムチさんは Slack でマリアさんに「プロジェクトに潜在リスクがあります」と一言送りました。何のことか見当つかなかったマリアさんは急いで戻り、ムチさんと直接話すことにしました。

「…じゃ、リスクはバグとかのことじゃないんだね。」

「ええ。アメさんをレビュアーとして教育すべきだと思います。」

「（ムチさんが、気になることを自分から話してくれたのは初めてだなぁ）」

「（でも、エマちゃんのフォローとか頑張ってくれてるアメさんに、何か足りない点があるんだろうか…）」

「アメさんのどのあたりが足りてないの？ 」

「レビューが行き当たりばったりなところかと。」

「行き当たりばったり…か。私にはよくわかんないけど事実としてムチさんはそう感じてるってことだね？ 」

「はい。」

「（もしかして、この前の OK と NG が 2 人の間で分かれた話が何か関係してるのかなぁ）」

「わかった、アメさんにも意見聞いてみるね。」

「よろしくお願いします。」

マリアさんは、後で行き当たりばったりとは何かを何度も考え直しました。一般的には、次のような状態を指します。

■行き当たりばったりのレビューとは

・複数のレビュアーがおり、人それぞれレビューポイントや評価基準が大きく違う
・同じレビュアーでも一貫性がない。例えば、グローバル変数をある日は禁止し、別の日は許可する
・前と同じ問題が見つかっても、前と同じ指摘が行われない。言うのを躊躇う。同じ指摘は 2 度と行われない

これらは人数が増えれば起こりがちです。ここで間違えないでほしいのが、「評価基準が"大きく"違う」と書いたように、多少は違ってよいのです。人間の目によるチェック漏れは 0 にできないし、レビュアーのスキルや経験に応じてレビューポイントが違うのも当然です。例えば可読性の低さやブラックボックス化の懸念など、不慣れな人のほうが気づきやすいこともありますね。

ですが、あまりにも行き当たりばったりな場合、発表者が気づいて理不尽さを感じたり、鬱憤、やる気の低下に繋がりかねません。

また、チェック漏れによる品質低下リスクも高まります。行きすぎないよう、Try と改善が必要です。

1.6 独裁の場

「アメさん、ちょっといいかな。」

「はい、なんでしょう。」

「ムチさんから、アメさんもムチさんと同じように厳しくレビューしてほしいと言われてるんだけど。」

「ほぉ、そうなんですか。」

「アメさんはどう思う？ 」

「確かに、レビュー基準は合わせたいですよね。」

「どんな風に合わせる？ 」

「例えば、Google 社の公開する Code Review Developer Guide をみんなで読み合わせたりとか。」

「なるほど、じゃ、アメさんもムチさんに賛成なんだね。」

「あ、でもムチさんと同じようにには簡単には乗れないですね。」

「誰かを絶対正しいみたいにすると、独裁的なレビューになってよくないと思うんです。」

「独裁的なレビュー？ 」

■独裁的なレビューとは

- 誰か 1 人が権力を握り、レビューを支配している
- 権力者に承認されることがレビューの目的となる

誰かが権力者になると、権力者なしで意思決定できないチームになったり、開発者のキャリアアップを妨げ、チームの人数が増えても成果も上がらなくなる傾向があります。

「なるほどね、じゃ続きはアメさんとムチさんで直接相談してもらえるかな？ 」

「はい、そうします！ 」

1.7 論点がズレている

「ということで、ムチさん、よろしくお願いします！ 」

「はい。」

「まず、私たちのレビューのやり方で、認識が合っている部分を明確にしません？ 」

「私は重要じゃないチェックに時間をかけなくてよいと思ってます。それはムチさんも一緒なんじゃないかと思います。」

「重要じゃないとは具体的に？ 」

「例えば命名規則とか、プラグインバージョン、スペルミス、マジックナンバーとかです。」

「そんなの人力で見てちゃダメだね。」

■論点がずれているレビューとは

- ソフトウェアの出荷には重要でない、コーディングの細かい部分ばかり言及される
- 自動レビューで賄える箇所を、手動で頑張ってチェックする

もちろんその洗い出し自体は大事ですが、そこばかり見すぎていると重要なものを見逃したり、時間を予想以上に費やしてしまいます。

これは几帳面な性格の人がいると起こりがちですが、人が見るレビューでは、ソフトウェアの品質に満足しているかとか、メンテナンス性、複数のやり方から最適な方法の選択といった、人でなければ判断できないことに集中しましょう。

「では、ムチさんと私で認識が違ってる部分は、どのへんが気になりますか？ 」

「アメさんは新人に優しすぎたり、以前 NG だったものも OK で返すことがあるよね？ 」

「う〜ん…確かに状況に応じてスピードを上げたり、反対に性能や品質を優先したりしてますね。」

「では、その状況判断の根拠は何？ 」

「えーと…正直、直感ですね。」

「それが良くないよね。レビューは責任持ってやってくれなきゃ。」

「すみません、わかりました。今後はムチさんの指示通りにレビューします。」

「お願いします。」

「（またなんかペース狂う…）」

「アメさん、ムチさん、大変だよ、市場トラブルが起きた！ 」

1.8 一方通行

マリアさんはプロジェクトメンバー全員を集め、今大規模な障害で全ユーザーがアプリを使えなくなっていること、至急修正が必要なことを伝えました。

エマちゃん、アメさん、ムチさんのチームは、大掛かりの修正をタスク分けしてスピーディーにこなしましたが、先ほどのレビューの話をしたばかりで時間がなく、レビュ〜はムチさんが全て引き受けました。

▼リスト 1.10: @ Muchi commented

```
@Ame
このユーティリティクラスを作った意図が伝わりません。
```

▼リスト 1.10: @ Ame commented

```
今後他のモジュールが呼ぶ可能性があるので作りました！
```

▼リスト 1.10: @ Muchi commented

```
それなら必要に応じてリファクタリングすれば良いと思います。
こんなイケてないことするくらいなら、今はコンパクトな参照関係にすべきです。
```

▼リスト 1.10: @ Ame commented

```
わかりました、修正します！
```

「（なんか、アメさんがムチさんにしごかれる流れになってるけど…チーム大丈夫なのかな）」

▼リスト 1.10: @ Muchi commented

```
@Ema
UIの変更が含まれてませんが、スケジュールはUI含め本日までです。レビュー都合もあるので
ボヤボヤせず早めに出してください。
```

「（ひぃん、人の心配してる場合じゃなかったぁ…）」

「（でも、ムチさんのコードって誰がレビューしてるんだろ…）」

エマちゃんはメインブランチのマージログを確認しました。すると、ムチさんはレビューなしで Main ブランチに入れていました。

「アメさん！ ムチさんのコードって誰もレビューしなくていいんですか？ 」

「う〜ん、したほうがいいよね…でも今そんな余裕ないし、ムチさん一番先輩だからまぁ大丈夫かなって。」

「なるほど、そういうものなんですね。」

「あっ、ごめんねエマちゃん！ そういうものではなく、見過ごしてただけでこれは本当はダメなパターンなの。」

「えっ？ 」

「一方通行のアンチパターンってよく呼んでるけど、余裕がなくてすっかり忘れてたわ。思い出させてくれてありがとう！ 」

■一方通行のレビューとは

- レビュアーのコードを誰もレビューしない
- 上下関係があり、若手が先輩のコードをレビューしない
- 一番上の人が自分のコードをアンレビューでマージする

これは年功序列や経験差がハッキリしたチームで起こりがちです。実は経験が浅い人こそ、他の人に無い観点で問題に気づくことがあるのです。実際、みなさんのチームでも若手が思わぬ改善点を見つけくれることはよくありますよね。

ベテランの書くコードにも意外な問題が潜んでいるものです。厳しめに言うと、長くいる人は成長してスキルが優れているというより、単に仕事に慣れただけということもよくあります。

そういった互いの立場を理解して、足りないところを埋め合うほうが、チーム内の雰囲気も品質も良くなるのです。

「だからエマちゃん、もし手が空いてたら私やムチさんのコードをレビューしてくれない？ 」

「えっ、私なんかがいいんですか？ 」

「エマちゃんしか気づけないこともたくさんあると思うの、だからお願い。ムチさんには私から説明しておくわ。」

「わかりました、やります！ 」

そして初めてレビュアーとなったエマちゃんでした。

「なぜエマさんがレビューをしているんですか？ 誰の指示ですか？ 」

「私です、エマさんの成長と一方通行レビューにならないようにお願いしてます！ 」

「（また独断で…。新人にレビューのやり方教える前に自分が学んで欲しいんだけど…）」

「（ま、新人に細かい実装のこと教える手間が減るのならいいか）」

エマちゃんは最初ほとんど Approve でしたが、やがて小さな部分に目が届くようになりました。それに、アメさんとも技術的な踏み込んだ話ができるようになりました。ムチさんはまだ怖いようですが…。

一方、アメさんとムチさんの間も変化しつつあります。ムチさんはアメさんに厳しくし始め、アメさんは半ば諦めたようにムチさんのやり方に従い始めました。アメさんには一体どんな考えがあるのでしょうか。次章に続きます。

1.9 良いレビューとは

レビューのアンチパターン集はここまでです。では、良いレビューとは一体何を満たすレビューでしょうか。筆者は、以下の要素を持つものと考えます。

1 つめは、建設的なフィードバックがあり、健全で活発な議論を生むことです。コメントを書く人は、自分の考えを明確にしつつ、それが正しいものだとは主張しません。状況に応じていろんな代替案や回避策が存在することを認め、修正の強要ではなく、どう思う？ などと問いかけます。もしレビュイーから反対意見が出た場合は、インターネットや先人の知恵に寄り道して、2 人の間でいい着地点を探しましょう。それが難しそうなら、第 3 者に客観的な意見を求めるのがよいでしょう。

2 つめは、レビュアーがレビュイーに共感していることです。いかなる方法でも、レビュイーが実装にかけた時間や労力を称賛し、強い口調ではなく親切で控えめな口調で接することが大切です。たくさんコメントした場合は、プルリクエスト以外の場所、例えば対面やオンライン会議、チャットやメールなどで積極的に連絡を取り、誤解が生じるのを防ぎます。そうすればきっとつらい思いをする人は減り、レビューの雰囲気や会話の方向性がとてもポジティブなものとなるでしょう。

3 つめは、全ての人が対等であることです。職種や職歴、スキルレベル、入社時期、サラリーなどに影響されず、全員が同じ基準、同じ目線で、かつお互い足りないものを埋める場所としてレビューを活用します。アンチパターンに挙げたような、スキルテストや評価の場として利用するのは適切ではありません。

第 2 章
レビューで折れないメンタルづくり

2.1 心理的安全性を高める

エマちゃんたちのチームは、紆余曲折ありながらもなんとかトラブルを収束させました。

「みんなお疲れ様。遅くまで対応してくれてありがとう。」

「今後こういうことを未然に防ぐため、ふりかえりを開かないとね。」

「ふりかえり…？ 」

「エマちゃん、やったことなかった？ 」

「そういえば、最初は週 1 回やってたけど、忙しくなってから忘れてたね。」

「ふりかえりは、アジャイル開発では必須と言われるチーム改善のための会議で、KPT や YWT などの方法で定期的に行うものよ。」

「どんな手法にも共通しているのは、チーム全員がよりよいやり方のアイデアを見つけ、提案することだよ。」

「私のような新人でも発言していいんですか？ 」

「もちろんよ。レビューと同じでプロジェクトの問題にも、入ったばかりの人の方が気づきやすいし。」

「あと、そういう人の不安を取り除く会でもあるの。」

「だから、感じたことはそのまま言ってね。」

「はい、わかりました！ 」

「でも、ふりかえり中に特定の人を非難するとか、盛り上がって関係ない話に行くのは NG だよ。」

「そうですね。多くの問題は人ではなく、チームのルールやプロセスにありますし、決めた時間どおりに進行するのも大事ですよね。」

そして、さっそくふりかえり KPT 方式で行われました。

エマちゃんは、忙しいのが過ぎ去ってよく眠れそうとか、技術的に成長したなどを Keep で挙げつつ、レビューに時間がかかってしまうことを Problem に書きました。

Try を書く時間では、レビュアーとレビュイーが直接話して疑問点を解決する、ペアプログラミングをするなどのアイデアが挙がりましたが、エマちゃんはどうもスッキリしません。ふりかえりが終わってから、アメさんに相談しました。

「すみません、個人的に聞きたいんですが…。」

「なぁに？ 」

「今回、アメさんもだいぶムチさんの指摘を受けてたじゃないですか。」

「だねぇ。久しぶりでなんか懐かしい気分だった♪」

「つらくならなかったんですか？ 」

「ん〜、そういうのはなかったかなぁ。細かく見てくれたし、助かります！ みたいな気持ちで。」

「すごいです。私だとなんか怖くて…。」

「あっ、エマちゃん、最初ムチさん怖いって言ってたもんね。今もなの？ 」

「はい…あまり印象変わってないです。ムチさんと直接話したりペアプロするの、気が重いです。」

「そっかぁ。ふりかえりの Try じゃ実は解決しないのね。」

「じゃ、エマちゃんは、ムチさんにもっと優しく接してほしい？ 」

「う〜ん…そのほうが安心ですが、アメさんや他の人も厳しく言われてるんだから、私だけって難しいのかなぁって思います…。」

「それに、友達になってほしいわけでもないし、むしろちゃんと言ってくれたほうが成長できる気もして…。」

「なんだか、うまく言葉にできないんです。」

「そうねぇ…たとえば、きつく言われても気にしなくなる方法があるならどう？ 」

「あっ、鋼のメンタルってやつですか！ 」

「ちょっと違うわね（笑）」

「きつく言われたときって、責められたとか、評価が落ちたって風に捉えてない？ 」

「はい、そう思っちゃうことが多いです。」

「もし、そうならないって確信できてれば、さっきエマちゃんが自分で言ったように、成長のためとか、細かく見てくれて助かったとかって思えない？ 」

「そうかも…。」

「エマちゃんが必要なのは、心理的安全性なのかもしれないわね。」

「心理的安全性、ですか？ 」

「詳しくは、マリアさんに教えてもらって。あの人、実はこういう話好きだから。知らなかったでしょ？ 」

「はい！ マリアさんに聞いてみます！ 」

心理的安全性とは、恐怖や不安を感じずに自分の意見を安心して伝えられる状態を指します。つまり、エマちゃんは今、心理的安全性が低い状態なのです。

心理的安全性が低い状態で感じる不安は、大きく分けると 4 種類あります。

■無知だと思われる不安

誰かに質問や相談をするとき、「そんなことも知らないのか」と思われるんじゃないかと感じる状態です。もし言われなくても、査定の低評価に繋がるのではないかと、一度感じた不安は勝手に膨らんでいきますよね。

その結果、わからないことを気軽に相談や質問できなくなり、大きなミスや孤立に繋がるかもしれないのです。

■無能だと思われる不安

何かをやったとき、「この程度なのか」と思われるんじゃないかと感じる状態です。

この不安はあらゆるネガティブな判断を引き起こします。ミスをしても報告せず隠蔽する、ミスを素直に認めない、本質から離れた努力をして有能に見せようとする、などです。その結果、大きなトラブルや損失に繋がる可能性がありますし、発覚が後になればなるほど大ごとになってしまうのです。

■邪魔だと思われる不安

発言や指摘をするとき、他人の足を引っ張って嫌われるのではないかと感じる状態です。

その結果、発言や指摘が少なくなり、アイデアの提案や他人のミスのフォローなどができなくなります。それにより、チームの一部の人の考えばかりが適用され、それ以外の人にとって居心地の悪い状態になったり、チームの改善機会が失われてしまいます。

■ネガティブだと思われる不安

発言や指摘をするとき、マイナスなことは言ってはいけないんじゃないか、せっかくの良い雰囲気を壊すのではないかと感じたり、それを繰り返すことでネガティブで後ろ向きな人だと思われるのを恐れる状態です。

最終的にチームを良くできる話でも、少しでもマイナスな要素が含まれていると発言や指摘をできないので、チームの改善機会が失われたり、本来持っている問題解決能力を発揮できず終わってしまいます。

「同じ状況でも感じ方は人それぞれだけど、心理的安全性が低いと、こんなにたくさんの問題があるんだよ。」

「ひぇー！ 大きなミスやトラブルなんて起こしたくないです！」

「それはみんな同じだよ。だから、エマちゃんが今感じている不安は、早く無くさないと後が大変だよ。」

「そうですね…どうすればムチさんともうまくやっていけますか？」

「ムチさんは悪意があってそう言ってるんじゃなく、エマちゃんに期待して言ってくれてると思うけどね。」

「それはわかるんですが…どうすればその、信頼性？ が高くなるんでしょう。」

よく誤解されますが、やわらかい言葉を選んで使うとか、はっきり言うのを避ける、笑顔でいる、賛成するといった優しさのことではありません。

心理的安全性が高いとは、無知・無能・邪魔・ネガティブと思われそうな行動をとっても安全であるとメンバー全員が感じており、お互いに踏み込んだ話し合いをしたり、助けを求めたり、リスクをとったり、時には厳しいことを言ったり、建設的な反対意見を出せる状態です。また、そうなることがゴールではなく、組織として高いパフォーマンスを出すことが目的で、そのための土台と考えるのがよいです。

心理的安全性を上げるために、私たちができるのは以下のようなことです。

■心理的安全性を高めるための意識づくり

・失敗が起こることを受け入れる
・人や過去を責めず、プロセスやこれから成功する方法を考える
・うまくいかないのは学びや成長の機会だととらえる
・指摘は一定レベルのことができてるから受けられるもの、土台には好評価と承認があるととらえる
・相手に好奇心を持ち、質問や意見を積極的にする
・予想と異なることに好奇心や関心を持つ

「どれも私に足りてないことかもです…。」

「エマちゃん、入社してから今まで許してもらえなかった間違いってある？ 」

「言われてみると、思いつかないです。」

「そうだね。私もない。お客さんや偉い人たちとも色々あったけど全部許してもらえた（笑）」

「でも不安になるのって、これからが予測しにくいからなんだよね。」

「私たちのいるソフトウェア業界は、VUCA と言われる不安定で、不確実で、複雑で、曖昧な状態の典型だから。」

「その中で予測をして、確実な利益にするには、心理的安全性が高いチームであることが大前提なんだよ。」

「なるほど、少しずつわかってきました。」

「でも、みなさんに好奇心を持っても、新人なので質問や意見をしずらいことが多いんですが…。」

「たしかに、人は誰でも安全な方法を取りたがるものだよ。」

「黙ってて解雇された人は聞いたことないし、先輩の意見に従えば責任も感じないからね。」

「でも、成果を出すチームの多くはミスが少ないチームじゃなくて、ミスを繰り返して成長してきた、報告が多かったチームだと言われているんだよ。」

「そうなんですかぁ。」

（TODO: なんか個人体験を書き起こしたい。思い出し中）

2.2 安全な組織づくりに加わる

「アメさんとムチさんって、性格は違ってもうまくやってるようですけど、2 人とも心理的安全性が高いんですか？ 」

「私が見た限り、2 人ともまだまだだね。」

「えっ、そうなんですか。」

「ムチさんは知ってのとおり完璧主義で、的確な意見を出してくれるし、意見することも厭わない人なんだけど。」

「一言が多くて、つい人を責めちゃったりね〜。」

「あはは…。」

「あと、必要最小限の会話しかしないけど、それは自分の責任範囲が増える怖れかなって思うんだ。」

「安心してもっといろんな事柄や変化に対して考えを言って欲しいかな。」

「アメさんは反対ですよね。」

「そうだね。アメさんは謙虚な自信家で、人から多少何か言われたり失敗した程度じゃ気持ちが揺らがないから、積極的に動いたり場をまとめてくれる。」

「それに、ふわっとしてるようでちゃんとリスク管理してるんだよね。大きなミスをしないから、安心して色々お任せできるんだ。」

「でも、詰めが甘いところはあって、意見が食い違ったらあっさり自分の意見を引っ込めちゃいがちだね。」

「ムチさんと議論してるとき、大体アメさんが譲ってますよね。」

「そうなんだよ。こだわりが無いのか、議論が面倒くさいのかわからないんだけど、ベストじゃなくベターで満足しちゃう感じ。」

「もうちょっと徹底的に問題を洗い出して報告して欲しいかな。」

「ちなみに、この話ってアメさんやムチさんにも直接言うんですか？ 」

「もちろん、2 人にはそれぞれ 1on1 で良いことと改善点をオープンに伝えるよ。私が感じた課題を隠してたら、それこそ心理的安全性に問題があるってことだからね。」

「でも、なんだかんだこの 2 人だから足りないものを埋めあって、そこそこうまくいってるかもしれないね。」

「 2 人が詰め切れてない部分があったら、そこはエマちゃんの出番だからね。」

「はい！ 私もお役に立ちたいです、いや、立ちます！ 」

「いいね！ 期待してるよ！ 」

人は誰でも、無意識に思い込んだり失敗を犯すものです。本書にはモデルケースはおらず、アメさんもムチさんも完璧にレビューしてはいないし、それをマネジメントしてるマリアさんにもコントロールできないことはたくさんあります。

こうすれば絶対正解という落としどころはなく、このチームが話し合ってきたことから、1 つでも多くの気づきが読者のみなさまに発生したならば幸いです。

2.3 文章の隅々を受け止めない

- 「イケてない」とか一言多い部分は読み返さない。さらっと流す
- コメントを細かく書けと言う人と、コメントなしでも理解できる読みやすいコードであるべきという人がいる話
- レビューのコメントを自分の価値、長期的な評価と直結しない

あとがき

自書を持ち、あとがきを書くこの瞬間。実は昔から少し憧れていました。
パッパラパーの話を入れる。

♣ 著者紹介

Vitantonio Nagauzzi (@tonionagauzzi)

Smartphone App Engineer, "You decide you're happy or not."

- https://about.me/knagauchi
- 株式会社 ACCESS

♣ 既刊一覧

- 『ACCESS テックブック』(技術書典 7)

アイのムチ

レビューで折れないメンタルづくりとレビュアー側の心得

2022 年 1 月 22 日　ver 1.0 （技術書典 12）

著　者　ACCESS 技術書典同好会
イラスト　（好きな表記を教えてください to Umemoto さん）
発行者　tonionagauzzi
連絡先　vitantonionagauzzi@gmail.com
http://about.me/knagauchi
@tonionagauzzi (https://twitter.com/tonionagauzzi)

(powered by Re:VIEW Starter)