第3回 エンジニア向け / 2026.07.29 WED PM

開発ワークフローと AI活用

株式会社 法研システムズ 御中 エンジニア向け

第3回 開発ワークフロー改善とAI併用のチームルール

2026年7月29日（水）14:00 - 17:00 ／ 対面

講師 **安田 光喜**（Givery）

講師紹介 / INSTRUCTOR

安田 光喜 Mitsuki Yasuda

Givery メイン講師 ／ 生成AIエバンジェリスト

- 公立はこだて未来大学 修士課程修了（AI・デザイン領域）
- デザイン会社・デジタル教育会社の設立と運営を経験
- 生成AI事業を設立・運営、企業研修の設計と登壇を担当
- 非エンジニア向けの入門研修を数多く担当

今日はエンジニア向けです。手を動かす演習は対話型のGeminiを軸に据えます。IDEのエージェントは後半に画面でお見せしますが、まずは対話でどこまでできるかを体で掴んでください。

本日3時間のゴール

開発の各工程でAIをどこに効かせるか、どこは人間が握るかを、自分の手で切り分けます。

- [x] 開発の7工程を棚卸しし、効果×リスクで**AI活用マップ**を1枚作る
- [x] マップ上位3件に『1ヶ月で試す』印を付け、明日からの一歩を決める
- [x] AI併用コーディングの**チームルール雛形**の叩き台を作る
- [x] マスキング・レビュー観点・テスト生成・バグ調査をGeminiで一通り体験する

今日はブラウザの **Gemini（対話型）が主役**です。IDE上のエージェント（Claude Code / Copilot / Cursor）はS04で画面をお見せする**世界観の共有**にとどめます。

今日の流れ

区分	セッション	形式	持ち帰るもの
導入	オリエンテーション	講義	本日の進め方の理解
講義	リスクマネジメントとガバナンス	講義＋デモ	任せる/握るの線引き
実践	Geminiを用いた開発タスク体験	ハンズオン	マスキング・レビュー・テスト・バグ調査の型
実践	開発ワークフロー改善と振り返り	ハンズオン	AI活用マップ＋チームルール雛形

計 [180min]。前半は座学とデモ、後半は手を動かします。発表やグループワークはありません。各自で進め、詰まったら手を挙げてください。

今回の前提 3点

- **手を動かす演習はブラウザのGeminiで行います。**特別なセットアップは不要、gemini.google.com を開くだけです
- **IDE上のエージェントはS04で講師画面のデモとして紹介します。**手元操作はありません。世界観を共有するのが目的です
- **機密情報・個人情報は原則入れません。**やむを得ない場合のみ、演習1で扱うマスキングをかけてから渡します

使うのはブラウザの **Gemini**（gemini.google.com）とクラス共有シート（Googleスプレッドシート）です。IDEエージェントのインストールは今日は不要です。

全員参加を見える化 クラス共有シート

URLを開く
講師が当日チャットで配布

→

自分の行を探す
名前の行に記入

→

毎演習ごとに記録
結果＋進み具合

正解を競うものではありません。**空欄を自分の言葉で埋めること**が、今日の『参加』です。講師はこのシートを見て、つまずいている方にすぐ声をかけます。

- 全員で1枚のGoogleスプレッドシートを使います
- 各演習のあと、結果を一言と進み具合を自分の行に記入します
- 進み具合は `やってみた` / `つまづき中` / `できた` の3つから選ぶだけ

SESSION 02 / 講義・デモ [40min]

リスクマネジメントとガバナンス

AIを開発に入れるほど、著作権・脆弱性・注入・情報管理のリスクが表に出ます。何を任せ、何を人間が握るか。線引きの根拠を40分で押さえます。

2026年 エンジニアのAI活用実態

90%
AIコーディング採用（DORA 2025）

75%
Google新規コードがAI由来

84%
開発者がAIツール使用

- AIコーディングツールの採用は**90%**（Google DORA 2025年レポート）
- Googleでは新規コードの**約75%**がAI由来（2024年の約25%から1年で上昇）
- 開発者の**84%**がAIツールを使用中／使用予定（Stack Overflow 2025年調査）

出典 Google DORA 2025 State of AI-assisted Software Development　Stack Overflow Developer Survey 2025

便利さと信頼のギャップ

84%
AIツールを使用

29%
出力を信頼

Stack Overflow 2025年調査では、使用率は**84%**まで上がった一方、出力を信頼する開発者は**29%**にとどまります。信頼は前年の**40%→29%**へ低下しました。**便利だが信用しすぎない**、これが現場の実感です。

- 使う人は増えたが、そのまま信じる人はむしろ減っている
- 『ほぼ正しいが微妙にずれる』出力への不信が背景
- だからこそ**人間の確認とレビューを工程に組み込む**のが前提になる

出典 Stack Overflow Developer Survey 2025: AI

リスク1 著作権・ライセンス

論点	内容
Doe v. GitHub（Copilot訴訟）	2025年に和解成立。学習データとコード生成の権利関係は依然グレー
著作権法30条の4（日本）	情報解析目的の学習は原則適法。ただし『享受目的』や権利者の利益を不当に害する利用は対象外
Gemini Code Assist のIP補償	Googleが著作権侵害クレームに対する補償を提供。契約条件の範囲内での利用が前提

生成コードがどのライセンスに由来するかは追跡しにくいため、社内でOSSライセンスの確認手順を持つのが現実的です。

出典 Doe v. GitHub 訴訟の動向 著作権法第30条の4（e-Gov） Gemini Code Assist の生成AI補償

リスク2 脆弱性

調査（年次）	内容
Veracode 2025	AI生成コードの**約45%**に何らかのセキュリティ脆弱性が混入
Apiiro 2025	AI併用でコード量が増え、脆弱性検出が**4倍**・深刻な露出が**10倍**に
Slopsquatting（2025〜）	AIが**約20%**の頻度で存在しないパッケージ名を提案。攻撃者がその名で悪意パッケージを登録

生成コードは動くことと安全であることが別物です。依存パッケージの実在確認とスキャンを工程に入れます。

出典 Veracode 2025 GenAI Code Security Report　Apiiro: AI coding assistants and security risk　Slopsquatting（パッケージハルシネーション）解説

リスク3 プロンプトインジェクション

- **プロンプトインジェクション**＝外部データに紛れた指示でAIを乗っ取る攻撃
- **CVE-2025-54794 / CVE-2025-54795**はClaude Codeで報告された脆弱性（2025年の事例）。パストラバーサルとコマンド実行につながりうるものとして修正済み
- **Comment and Control**＝コードコメントやドキュメントに隠した指示でエージェントを操る手口
- 外部から読み込む資材（Issue・PR・Web）を信頼しきらない設計が2026年前半でも継続論点

エージェントに外部データを読ませるほど、その中に紛れた指示に従ってしまう危険が増えます。**読ませる対象を絞る・実行前に人間が確認する**が基本です。

出典 Cymulate: prompt injection の実態　SecurityWeek: Claude Code の脆弱性報道

AIに任せる / 人間が握る 線引き

AIに任せる

- 観点出し・たたき台の列挙
- 初稿・ドラフトの生成
- 定型的な変換・整形
- 調べものの下ごしらえ

人間が握る

- 承認と本番反映の判断
- アーキテクチャ・設計の意思決定
- 法務・セキュリティの最終確認
- 責任を伴う対外コミュニケーション

AIは**広げる**のが得意、人間は**決める**のが役目。任せる範囲を明文化しておくと、チームで判断がぶれません。

コードレビュー方針の現実解

組織・時期	方針
Linux Kernel（2025年11月）	AI補助は**Assisted-by:** の明記を必須化。ただし**Signed-off-by:**（責任表明）はAI名義を禁止
Anthropic Code Review（2026年3月）	**CLAUDE.md＋REVIEW.md**で観点を明文化。AIレビュー後も**人間の承認**を必須とする運用

AIレビューは観点の抜け漏れを減らす一次フィルタ。最終承認は人間が担う二段構えが定着しつつあります。

出典 Linux Kernel documentation: AI assistance　Anthropic: Claude Code のコードレビュー運用

AIガバナンスの公的枠組み

枠組み	要点
AI事業者ガイドライン 第1.2版（2026年3月31日）	総務省・経産省。AIエージェント／フィジカルAIのリスク整理、リスクベースアプローチの具体化
日本のAI法（2025年9月1日 全面施行）	理念法で罰則なし。内閣にAI戦略本部を設置、基本計画策定・実態調査・指導等
EU AI Act（2026年8月2日〜）	ガバナンス構造とGPAI執行が適用。Digital Omnibus暫定合意（2026年5月7日）で高リスク（Annex III）義務は2027年12月2日へ延期

出典 AI事業者ガイドライン 第1.2版 内閣府 AI法 欧州委員会 AI規制フレームワーク

データの取り扱い 3つの環境

データの種類	無料版のGemini	個人の有料プラン	会社アカウント(Workspace)
入力が学習に使われるか	使われることあり(設定でオフ可)	無料版と同じ(オフ可)	契約上 使われない
機密情報(社外秘・契約など)	原則 入れない	原則 入れない	原則 入れない(必要時マスキング)
個人情報(氏名・顧客データ)	入れない	入れない	入れない(必要時マスキング)
ソースコード(社内リポジトリ)	入れない	入れない	扱う場合は社内規程に従う

会社アカウント(Workspace)は入力が**学習に使われない**設計です。それでも機密・個人情報・社内コードは『**念のため入れない**』が実務の主流。**迷ったら入れない**、やむを得なければ次のマスキングが原則です。

出典 Google Workspace 生成AI プライバシー　Gemini Apps プライバシーハブ

リスク / SECURITY

どうしても使うときは マスキング

- **ダミー置換**（今回はこれ）　田中花子→顧客A、○○商事→A社、実キー→DUMMY-KEY
- **列ごと削除**　ログやCSVの氏名・電話・住所の列は丸ごと消すのが確実
- **一般化**　37歳→30代、内部IP→ネットワーク種別だけ、実URL→example.com
- **仮名化 / 匿名化**　記号に置き換える（元に戻せる／戻せない の違い）

コツは3つ。**同じ名前は必ず同じ仮名へ**（ゆらさない）。**迷う列は消す**（塗るより確実）。最後に**消し残しを目視**（ログの末尾やスタックトレースに実値が紛れがち）。演習1でこの判断を実際に手でやります。

出典　個人情報保護委員会 仮名・匿名加工情報

実例 / PRACTICE

ベストプラクティスの実態

95%
メルカリ AI利用率

70%
AI生成コード比率

64%
開発量の増加

メルカリでは社内のAI利用率**95%**、AI生成コード比率**70%**、開発量が**64%増**（いずれも2025年次）。社内『Project Double』では対象業務の工数が**約1/5**に。**もともと強いチームほど効果が大きい**のが共通点です。

- 効果が出るのはツール導入だけでなく、レビューやルール整備がそろったチーム
- AIで量が増えるほど、確認とテストの設計が効いてくる
- 『入れれば速くなる』ではなく『整備した分だけ速くなる』

出典 メルカリ AI活用の取り組み（Mercari Engineering） メルカリ Project Double 事例

向き合い方の原則 5つ

- **AIは対話相手であって検索ではない。**1回で諦めず、3回4回と聞き返す前提で使う
- **1回で終わらせない。**追加で『もっとこうして』と磨くのが正しい使い方
- **出力は必ず人間が確認する。**生成コード・依存パッケージ・数値は動作と実在を照合する
- **機密・個人情報・社内コードは入れない。**判断に迷ったら入れない
- **AIレビュー→人間レビューの2段階を守る。**AIだけで承認しない

SESSION 03 / ハンズオン [55min]

Geminiを用いた開発タスク体験

ここから手を動かします。マスキング判断・要件分解・レビュー観点・テスト生成・バグ調査を、対話型Geminiで一通り体験します。当日必須は演習1〜3、演習4・5は時間次第で任意です。

すべての演習は4ステップで進む

1 考える
何を伝えれば良い回答が返るか1〜2分予想

→

2 書く
自分の言葉で依頼文を書く

→

3 実行
Geminiに投げて回答を読む

→

4 答え合わせ
hintsの参考例と自分の依頼文を並べる

『考える』を飛ばさないのがコツ。参考プロンプトを先に見るとコピー作業になり、体験が残りません。答え合わせは講師の合図で hints フォルダを開きます。

- 参考プロンプトは `hints/stepNN_xxx.md` に入っています
- 開くのは Step4『答え合わせ』のタイミングだけ
- 各演習のあと、クラス共有シートの自分の行に気づきを1行残します

演習 / [13MIN]

演習1 入力前のマスキング判断

- 題材＝ `demo_data/マスキング練習_生ログ.txt`
  （架空の障害調査ログ）
- Step1 このログを**そのまま貼ってよいか**を1〜2分で判断する
- Step2 残す列・消す列・置き換える値を決め、加工版を自分で作る
- Step3 加工版をGeminiに渡して読ませ、意味が通るか確認する
- Step4 `hints/step01_masking_hint.md` を開き、自分の加工と並べる

マスキング版の例（答え合わせで開く）

次の障害調査ログについて、原因の当たりを付けてください。
個人を特定できる情報と実際の認証情報は伏せてあります。
（氏名→顧客A、実キー→DUMMY-KEY、内部IP→一般化 済み）

全消しは働けず、生貼りは漏れる。その中間を手で決めるのが今日の判断です。ログ末尾やスタックトレースに実値が残りやすい点に注意してください。

講師デモ(手元は任意) / [3MIN]

デモ 渡す前に安全化する

- 加工前のログを講師画面で開く
- 氏名の列を削除、または『田中花子→顧客A』のダミー置換を実演
- 実キーを `DUMMY-KEY`、内部IPを一般化して、加工後だけをGeminiに渡す

依頼例

> このログから障害の原因仮説を挙げてください。
> 個人情報と認証情報は含まれていません。

ダミーが『顧客A』『DUMMY-KEY』なのは、**実務でも氏名や認証情報をこう加工してから渡す**ためです。加工は手作業でも十分。渡す前のひと手間を習慣にしましょう。

演習 / [10MIN]

演習2 要件分解と仕様ドラフト

- 題材＝ `demo_data/要件メモ_新規機能.md`（曖昧さを含む機能メモ）
- Step2 メモから**曖昧な点・決まっていない点**を抽出させる
- Step3 抽出結果をもとに、仕様ドラフトを**構造化**して出させる
- Step4 `hints/step02_requirement_decompose_hint.md` で観点を確認

参考プロンプト（答え合わせで開く）

> 次の要件メモを読み、曖昧な点・未確定の点を
> 箇条書きで洗い出してください。そのうえで、
> 目的／入力／出力／制約／受け入れ条件の
> 見出しで仕様ドラフトを構造化してください。

曖昧さを先に潰すのがコツ。AIに仕様を書かせる前に、決まっていない点を可視化すると手戻りが減ります。

演習 / [12MIN]

演習3 コードレビュー観点出し

- 題材＝ `demo_data/レビュー対象コード_python.py`（意図的に問題を仕込んだ関数）
- Step2 **バグ / セキュリティ / パフォーマンス / 可読性 / テスト網羅**の5観点でレビューを依頼
- Step3 指摘を読み、根拠が薄いものを自分で見極める
- Step4 `hints/step03_code_review_hint.md` で見落とし観点を確認

参考プロンプト（答え合わせで開く）

> 次のコードを、バグ／セキュリティ／パフォーマンス／可読性／テスト網羅 の5観点でレビューしてください。
> 各指摘に重要度と修正方針を1行ずつ添えてください。

AIレビュー→人間レビューの2段階が前提です。AIの指摘は一次フィルタ。**AIだけで承認しない**を守ってください。

演習 / [10MIN]

演習4 テストケース生成

- 題材＝ `demo_data/テスト対象関数_python.py`
- Step2 **正常系 / 境界値 / 異常系**の3段階でテストケースを出させる
- Step3 pytest形式のコードまで生成させ、抜けたケースを自分で足す
- Step4 `hints/step04_test_generation_hint.md` で網羅観点を確認

参考プロンプト（答え合わせで開く）

> 次の関数に対するpytestを書いてください。
> 正常系・境界値・異常系の3段階で分け、
> 各ケースが何を検証しているかコメントを付けてください。

境界値と異常系は人間が見落としがちな一方、AIが列挙を助けてくれる領域です。**時間次第で任意送り可**（早く終わった方向け）。

演習 / [10MIN]

演習5 バグ原因特定

- 題材＝ `demo_data/障害ログ_dummy.txt`（架空の障害ログ）
- Step2 ログから**原因の仮説を3件**と、それぞれの**確認手順**を出させる
- Step3 仮説の優先順位を自分で付け直す
- Step4 `hints/step05_bug_analysis_hint.md` で見落とし仮説を確認

参考プロンプト（答え合わせで開く）

> 次の障害ログから、考えられる原因の仮説を3件挙げ、
> それぞれについて確認手順を具体的に書いてください。
> 最も疑わしい順に並べてください。

AIは仮説を**広げる**のが得意。絞り込みは人間が担います。**時間次第で任意送り可**（早く終わった方向け）。

SESSION 04 / ハンズオン [55min]

開発ワークフロー改善と振り返り

開発の7工程を棚卸しし、効果×リスクでAI活用マップを作ります。続いてチームルール雛形の叩き台を作り、最後にIDEエージェントの世界観を講師画面で共有します。

演習 / [30MIN]

演習6 開発ワークフロー×AI活用マップ

- 題材＝ `templates/開発ワークフロー×AI活用マップ_テンプレート.csv`
- Step1 自分の開発の**7工程**（要件／設計／実装／レビュー／テスト／デバッグ／運用など）を棚卸し
- Step2 各工程を**効果**と**リスク**で評価し、優先度を付ける
- Step3 上位3件に**『1ヶ月で試す◯』**を立て、共有シートに記入
- Step4 `hints/step06_workflow_map_hint.md` で工程分類と優先度の付け方を確認

列	書くこと
A 工程	要件定義 / コードレビュー など
B AIで効く部分	観点出し・下書き・変換の一部
C 効果	高 / 中 / 低
D リスク	高 / 中 / 低（機密・誤りの入りやすさ）
E 1ヶ月で試す	◯ を上位3件に

完璧なマップより**『まず試す1件』が決まること**が目的です。効果が高くリスクが低い工程から着手するのが現実的です。

演習 / [8MIN]

チームルール雛形 叩き台づくり

- 題材＝ `templates/AI併用コーディング_チームルール雛形.md`
- S02の『任せる／握る』とレビュー方針をもとに、自チーム版へ書き換える
- 決めるのは**入力してよい情報の範囲・AIレビューと人間承認の手順・生成コードの由来確認**
- この8分は**叩き台まで**。磨きは持ち帰りで構いません

完成度より**着手優先**です。空欄でも、自チームの言葉で1項目埋まれば十分。持ち帰って正式版に育ててください。
発展・任意として templates/PRレビュー観点リスト_テンプレート.md の自チーム化もあります（当日は扱いません）。

講師デモ(手元操作なし) / [12MIN]

AI駆動開発ツール 世界観の共有

ツール	2026年前半の状況
Claude Code	ターミナル/IDE/デスクトップ/Slackで動く自律エージェント。モデルは**Claude Sonnet 5**（2026年6月30日、Sonnet 4.6を置換）
GitHub Copilot	2026年6月1日から全プランが**従量課金（AI Credits）**へ移行。Business/Enterprise向けに自社モデル**MAI-Code-1-Flash**がGA
Cursor	2026年6月のTeams更新で、ファーストパーティ（**Composer 2.5**等）とサードパーティAPIの使用量プールを分離

経験豊富な開発者は平均**2.3ツール併用**と報告されています。**今回の軸はGeminiの対話型**で、これらのIDEエージェントは興味喚起の紹介どまりです。手元操作はありません。

出典 AI Coding Tools Pricing（2026年6月） Flat-rate AI coding subscription era is ending

振り返り クラス共有シートに記入

- [x] 自分の開発で**最も効きそうな工程**を1つ
- [x] AI併用の**整備で悩む点**（ルール・レビュー・情報管理など）を1つ
- [x] **1ヶ月後に試す1件**と、その第一歩

3問を共有シートの自分の行（振り返り欄）に記入してください。書き終えたら、講師がシートを画面に映し、いくつかの気づきを匿名で拾って紹介します（お名前は出しません）。**空欄が残っていたら、退室前に一言だけでも書き足してください。**

全3回の総括と1ヶ月後

回	テーマ	到達点
第1回	完全入門編	業務で1件AIを使えた成功体験
第2回	基礎研修：プロンプト実践	自部署向けAI活用計画書
第3回（本日）	エンジニア向け	開発ワークフロー×AI活用マップ＋チームルール雛形

1ヶ月後にレビューを
マップ上位3件から**1件**をチームで導入してみてください。1ヶ月後に、何が効いて何がずれたかをチームで振り返ると、ルールが実態に合っていきます。

今日いちばんの目標は**『どこにAIを効かせ、どこは人間が握るか』を自分の言葉で言えるようになること**。マップに書いた1件、ぜひチームで試してみてください。3時間お疲れさまでした。