

מה Claude Code יודע לעשות ועוד לא ניסיתם

כ-40 יכולות של Claude Code ב-54 דקות: מהגדרות שעושים פעם אחת, דרך קיצורים
יומיומיים ונקודות הרחבה, ועד שימושים שאינם קוד

Gui Ferreira · NDC Conferences · 28.10.2026

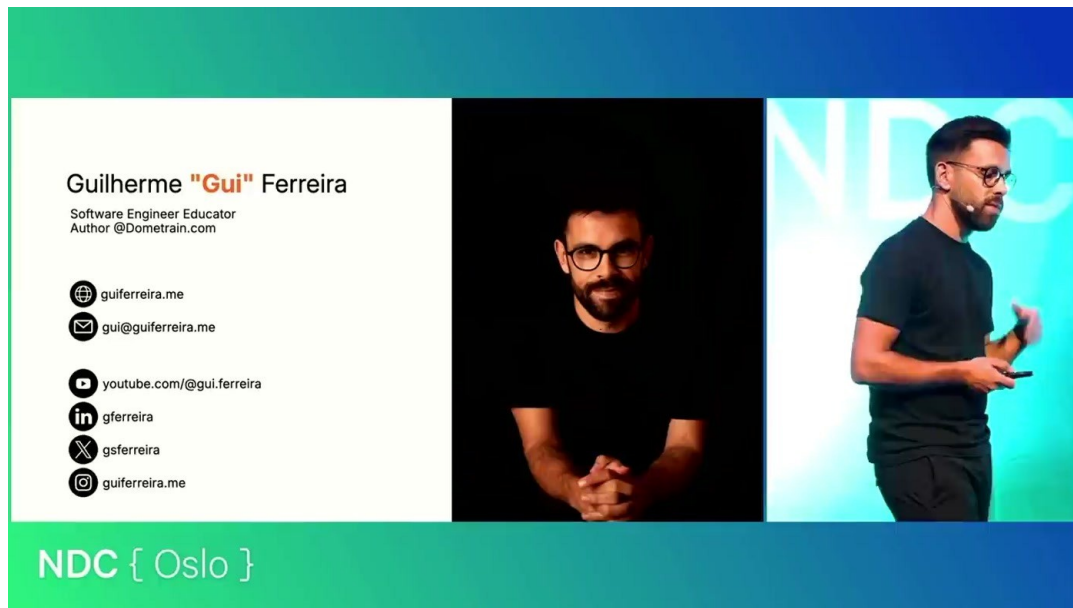
משך: כ-54 דקות · YouTube

סיכום אוטומטי — AI Briefing (aibriefing.dev)

על המסמך הזה

זו הרצאה מ-Guilherme "Gui" Ferreira (בשקופית: NDC Oslo 2026). הוא מהנדס תוכנה ומדריך שמעביר סדנאות Claude Code, ומציג את עצמו כמי שאינו קשור ל-Anthropic. הוא לא מלמד את הבסיס. ההרצאה בנויה כ-rapid-fire של כ-40 יכולות שהוא עצמו משתמש בהן ביומיום, ומיועדת למי שכבר הריץ Claude Code לפחות פעם אחת.

הטענה המרכזית שלו: רוב המפתחים פותחים את Claude, מקבלים מיד קפיצה גדולה בפרודוקטיביות ונעצרים שם, אחרי שלמדו בערך 10–20 אחוז מהכלי. הערך האמיתי נמצא בפרטים הקטנים: CLAUDE.md, ניהול context, יזום, hooks במקום הוראות טקסטואליות, skills כחבילות יכולת, ושימוש ב-Claude Code גם למשימות שאינן קוד. היעד שהוא מציב לקהל צנוע: לצאת מהחדר עם שניים-שלושה דברים לנסות ביום שני.



שקופית הפתיחה: Guilherme "Gui" Ferreira, Software Engineer Educator, עם קישורים לאתר, ל-Youtube ולרשתות החברתיות שלו

איך לקרוא את המסמך

- 1. **תקציר מנהלים:** הטענות המרכזיות, כל אחת עם חותמת זמן, וגם מה לא נאמר בהרצאה.
- 2. **ארבעה דליים:** מבנה ההרצאה ואיפה מריצים את Claude.
- 3. **Setup: CLAUDE.md,** קבצי settings, deny ו-status line.
- 4. **Daily workflow:** טבלת קיצורים ופקודות, ניהול context, remote control, voice ועבודה במקביל.
- 5. **The Platform Layer:** hooks, skills, ה-skills המובנים, mcp.json ו-headless mode.
- 6. **Beyond Code:** סיכום רבעוני, ארגון תיקיית ההורדות, Obsidian ותוכנית ריצה.
- 7–8. **פעולות ומילון:** מה לעשות מחר בבוקר, שאלות פתוחות ומילון מונחים.

הערות מקור המסמך הופק אוטומטית מהתמלול ומהשקופיות של ההקלטה. תמלול אוטומטי (ASR) עלול לשבש שמות ומונחים, למשל "cloud" במקום Claude או "ptw/" במקום /btw/, ובמקומות כאלה המסמך מציין את אי-הוודאות. הציטוטים מועתקים מהתמלול מילה במילה ובבדקו מול חותמת הזמן שלהם. לא נוסף שום מידע ממקורות חיצוניים.

1. תקציר מנהלים

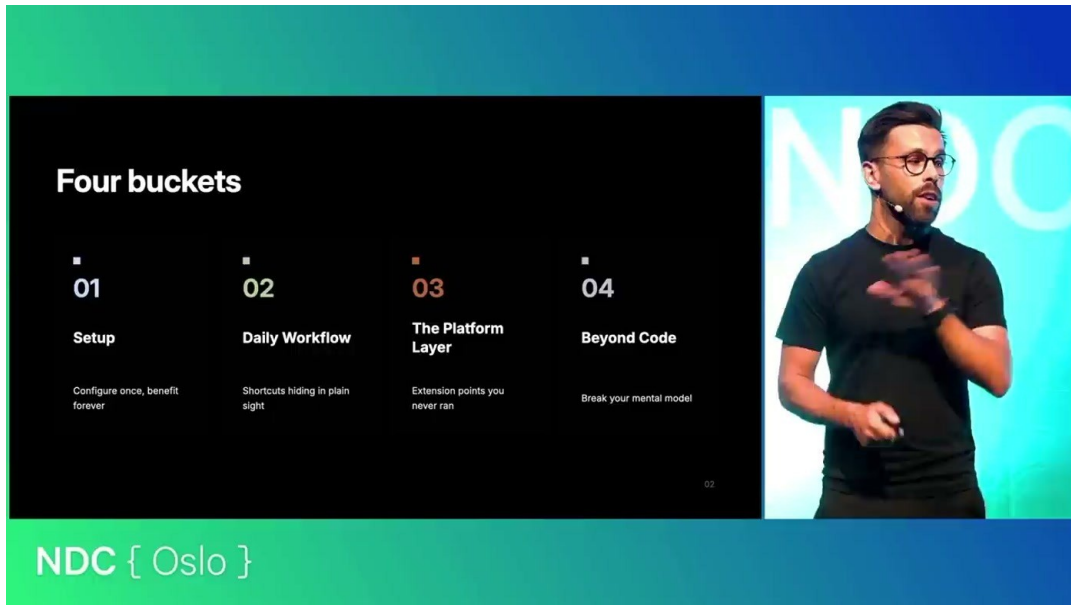
Ferreira מחלק את ההרצאה לארבעה "דליים": הגדרות חד-פעמיות, זרימת העבודה היומית, שכבת ההרחבה ושימושים מחוץ לקוד. רוב הזמן מוקדש לדלי השני. החוט המקשר בין כולם הוא ניהול context ועלות: כל טוקן שנכנס ל-context בלי צורך פוגע באיכות של האיטרציות הבאות ומייקר אותן.

- **CLAUDE.md ארוך מדי:** לפי מחקר של ETH Zurich שהוא מצטט, CLAUDE.md ארוך ומפורט מוריד ביצועים בכ-3% ומעלה עלויות בכ-20%. המלצתו: לנקות לפי שלוש שאלות, או להריץ /doctor שעושה זאת אוטומטית.
- **חננו גישה לסודות ב-settings.json:** בלי permissions.deny, Claude שמנסה לפתור באג עלול לקרוא קבצי secrets ומשתני סביבה, והמפתחות יישלחו ל-Anthropic.
- **ערכו את התוכנית ישירות במקום להתווכח עליה:** Ctrl+G פותח את ה-plan כקובץ markdown בעורך. אחרי השמירה Claude ממשיך מהגרסה הערוכה.
- **Rewind עדיף על refactor:** Esc כפול או /rewind מחזירים אחורה, כך שכיוון שננטש לא נשאר ב-context ולא מזהם אותו.
- **הריצו context על session ריק:** כך רואים כמה context תופסים skills ו-MCPs שאינכם צריכים. אם מדובר בכ-8% בהתחלה, כדאי לנקות.
- **Compaction יזום:** הוא מבצע /compact בעצמו בסוף כל שלב בפיצ'ר. את סף ה-40% ("dump zone") הוא מציג כדעה רווחת ולא כממצא מוכח.
- **Hooks במקום הוראות ב-CLAUDE.md:** PostToolUse hook שמריץ טסטים ו-linter פועל בכל פעם, לא עולה טוקנים, והפידבק שלו חוזר לשיחה.
- **Skill הוא חבילה ולא רק prompt:** בתיקיה אפשר לשים סקריפטים ותיעוד, לפני שכותבים skill משלכם כדאי לנסות את /simplify, /batch ו-/loop המובנים.
- **Claude Code כעוזר אישי:** ארגון 1,626 פריטים בתיקיית ההורדות, vault של Obsidian, ושני skills שמושכים ריצות מ-Strava ובונים תוכנית אימונים שבועית.

מה לא נאמר

- **המחקר לא מצוטט במדויק:** Ferreira מייחס ל-ETH Zurich את המספרים 3%-ו-20% בלי שם מאמר, בלי גודל מדגם ובלי לפרט על אילו משימות נמדדו.
- **סף ה-40%:** "dump zone" מוצג במפורש כדעה ללא מחקר שתומך בו. אין נתון משלו.
- **אין מספרי עלות:** הוא ממליץ לעקוב אחרי עלות ל-session, אבל לא מציג אף נתון עלות אמיתי מהעבודה שלו.
- **סיכוני remote control:** אין התייחסות לאבטחה או להרשאות כשמאשרים פעולות מהטלפון.
- **מגבלות deny:** לא נאמר מה permissions.deny לא מכסה, ולא כיצד לבדוק שהוא אכן אוכף את מה שהוגדר.
- **זמינות ויציבות:** לא צוין אילו פקודות (/goal, /focus, claude agents) ניסיוניות או תלויות גרסה. גם לא היו הדגמות חיות, רק צילומי מסך.

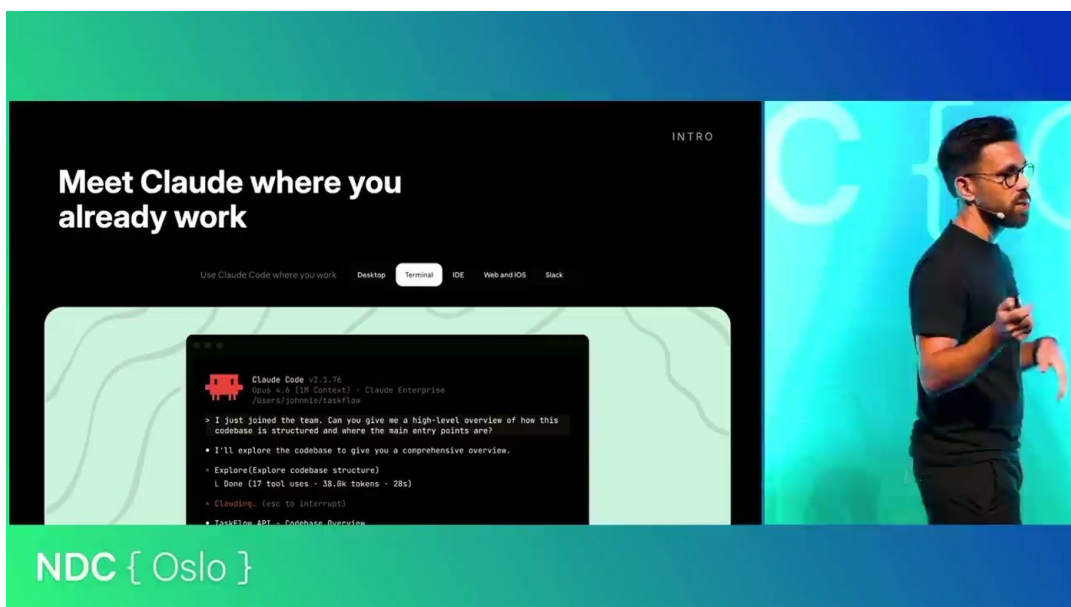
2. ארבעה דליים ואיפה מריצים את Claude



שקופית "Four buckets": 01 Setup (Configure once, benefit forever), 02 Daily Workflow (Shortcuts hiding in plain sight), 03 The Platform Layer (Extension points you never ran), 04 Beyond Code (Break your mental model)

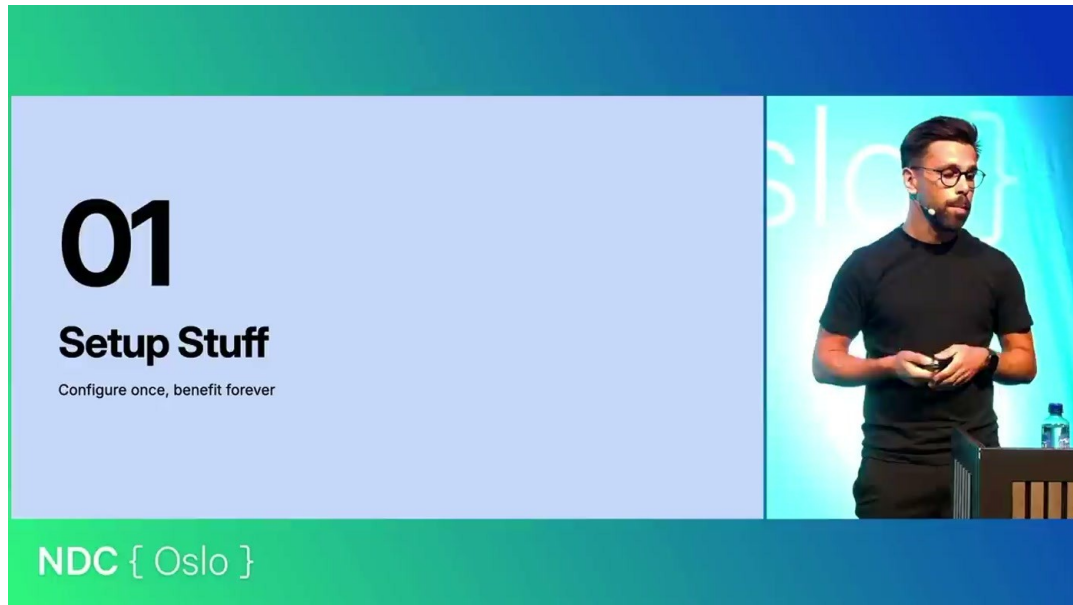
הפתיחה ממקמת את הבעיה: ביום שישי שלפני ההרצאה ספר Ferreira "104 slash commands" זמינות כברירת מחדל, והקצב גבוה מכדי לעקוב אחריו. הוא מאבחן את המשתמש הטיפוסי כמי שקיבל קפיצה מהירה בפרודוקטיביות "and you just learn 10 to 20 percent of it".

את Claude אפשר להריץ ב-terminal, כהרחבה ב-IDE או כאפליקציית desktop. הוא עצמו עובד ב-terminal, כי זה מהיר יותר וכי "usually the experimental features come there first". לדבריו רוב מה שהוא מציג עובד בכל הסביבות.



"Meet Claude where you already work": בחירת סביבת ההרצה (terminal, IDE, desktop) לצד צילום של מסך הפתיחה של Claude Code ב-terminal

3. Setup — מגדירים פעם אחת, נהנים לתמיד



שקף מעבר לפרק 01: "Setup Stuff — Configure once, benefit forever"

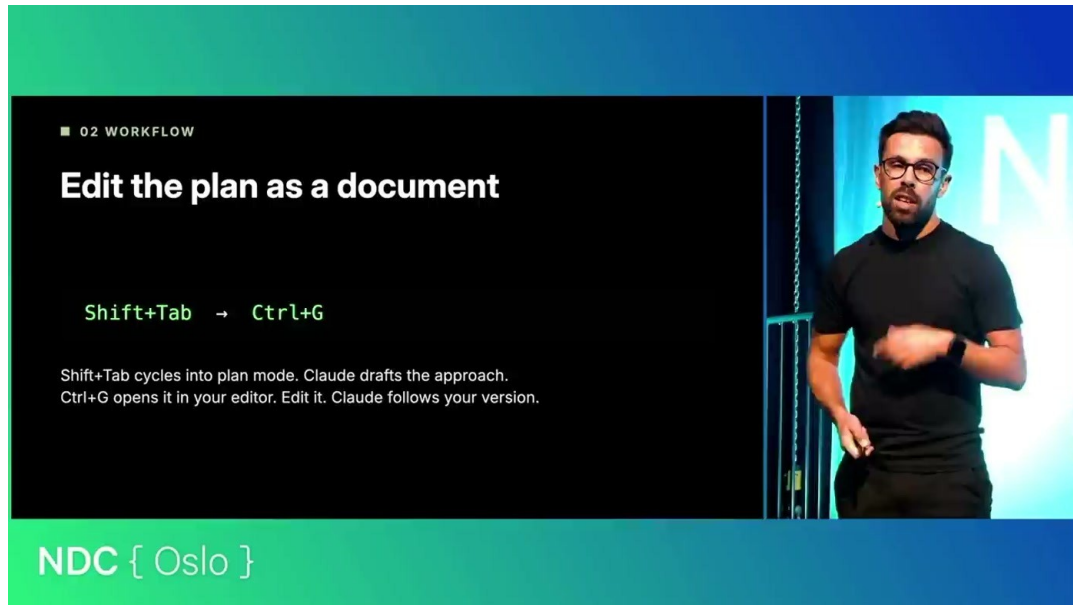
הצעד הראשון למי שחוזר מכלי אחר הוא `import` עם שם הכלי. הפקודה מייבאת MCP servers, קבצי תצורה ו-`sub-agents` מ-Cursor או Codex. הצורך בה נובע מכך שאין סטנדרט משותף לכלים, והבולט שבהבדלים הוא `CLAUDE.md` נקרא `AGENTS.md` בכלים אחרים.

לגבי `CLAUDE.md`, Ferreira מודה שבשנה שעברה היה ממליץ לחדוש לקובץ כל מה שאפשר, ושזו הייתה טעות. הבעיה הראשונה היא כפילות: אותו מידע מופיע ב-`README` וב-`CLAUDE.md`, וה-`agent` מתבלבל. הפתרון הוא `import` עם `@` ונתיב לקובץ, למשל כללי צוות או `schema`. הבעיה השנייה היא אורך הקובץ:

— Gui Ferreira eth Zurich proved that the performance when you have a long and descriptive `cloud.md` tends to drop in 3% and your costs will increase in 20%

- **שלוש שאלות לכל הוראה ב-CLAUDE.md:** (1) האם צריך אותה בכל משימה? (2) אם היא ספציפית, למשל הרצת `performance tests`, האם להעביר אותה ל-`skill`? כלל לשפה או לסימט קובץ עובר ל-`claude/rules`. (3) האם ה-`agent` יכול להבין את זה לבד? מבנה תיקיות, למשל, ממילא יתיישן ויסטה מהמציאות.
- **doctor/ עושה את הניקוי:** מעבר לבדיקת התקנה, כשמריצים אותו בתוך פרויקט הוא מפעיל את שלוש השאלות על `CLAUDE.md`, מסמן כפילויות ומציע מה להעביר ל-`skills`.
- **שלוש שכבות של CLAUDE.md:** הקובץ המשותף לצוות. `CLAUDE.local.md` שנמצא ב-`gitignore`, להעדפות אישיות כמו TDD. ו-`claude/CLAUDE.md` בפרופיל המשתמש, להוראות שחלות על כל `session`, למשל שפת שיחה.
- **permissions.deny ב-settings.json:** רשימת קבצים ופקודות `bash` ש-Claude לא יקרא ולא יריץ. לדבריו, בלי זה ייתכן ש-"your keys are going over the wire through Anthropic".
- **Theme ו-status line:** theme משנה את התצוגה, אבל העיקר הוא `status line`: לוח מחוונים שמציג פרויקט, branch, מודל ואחוז ניצול של `context window`. נוצר בשיחה עם `statusline` ש-Claude הופך לסקריפט.

4. Daily workflow — קיצורים שמסתתרים לעין כל



"Edit the plan as a document": Shift+Tab נכנס ל-plan mode, Claude מנסח את הגישה, Ctrl+G פותח אותה בעורך, ו-Claude ממשיך לפי הגרסה הערוכה

זה החלק הארוך בהרצאה. הרעיון המרכזי: לא לנהל משא ומתן עם Claude על פרטים קטנים בתוכנית. Ctrl+G פותח את ה-plan בעורך, ולדבריו זה "It's like a shared Google Doc with Claude". כדי לצמצם הנחות סמויות, הוא מבקש מ-Claude לראיין אותו דרך כלי ה-AskUserQuestion: "hey Claude, interview me and don't stop until we clarify every single thing". כשכיוון מתברר כשגוי, rewind עדיף על בקשת refactor, כי הקוד שנזנח לא נשאר ב-context: "It's like taking one step back to then take two steps forward". בטבלה מרוכזים הקיצורים והפקודות שהוצגו:

פקודה / קיצור	מה זה עושה ומתי
Shift+Tab → Ctrl+G	plan mode, ואחריו עריכת התוכנית כ-markdown בעורך
"interview me"	Claude שואל שאלות דרך AskUserQuestion עד שכל הנקודות מובהרות
Esc Esc / /rewind	חזרה להודעה קודמת ותיקון ה-prompt, בלי להשאיר שאריות ב-context
/model + effort	החלפת מודל ורמת reasoning לכל ה-thread. בתמלול נשמע גם "Fable" לצד Opus, Sonnet, ו-Haiku (שם לא ודאי)
ultrathink	max effort ל-prompt אחד בלבד, ואחריו חזרה לרמה הקודמת
Ctrl+S	שומר בצד prompt (stash) שעוד לא נשלח כדי לענות ל-Claude, ומחזיר אותו אחר כך
/btw	שאלת צד בזמן שהמשימה רצה, בלי לזהם את ה-context (בתמלול: "ptw")
עלות ל-session	פילוח לפי מודל, כולל sub-agents. שם הפקודה בתמלול לא ברור ("status")
/context	להריץ על session ריק כדי לאבחן כמה context תופסים MCPs ו-skills
/compact	compaction יזום בסוף כל שלב, ולא רק אוטומטית כשה-context מתמלא
/memory	בדיקת הזיכרונות שנשמרו וקידום העדפות שימושיות ל-CLAUDE.md
!	מריץ פקודת bash ישירות בלי reasoning. הפלט נכנס ל-context
/diff	חלונית שמציגה את הקבצים והשורות ש-Claude שינה
/voice	הכתבה קולית: לוחצים על רווח ומדברים

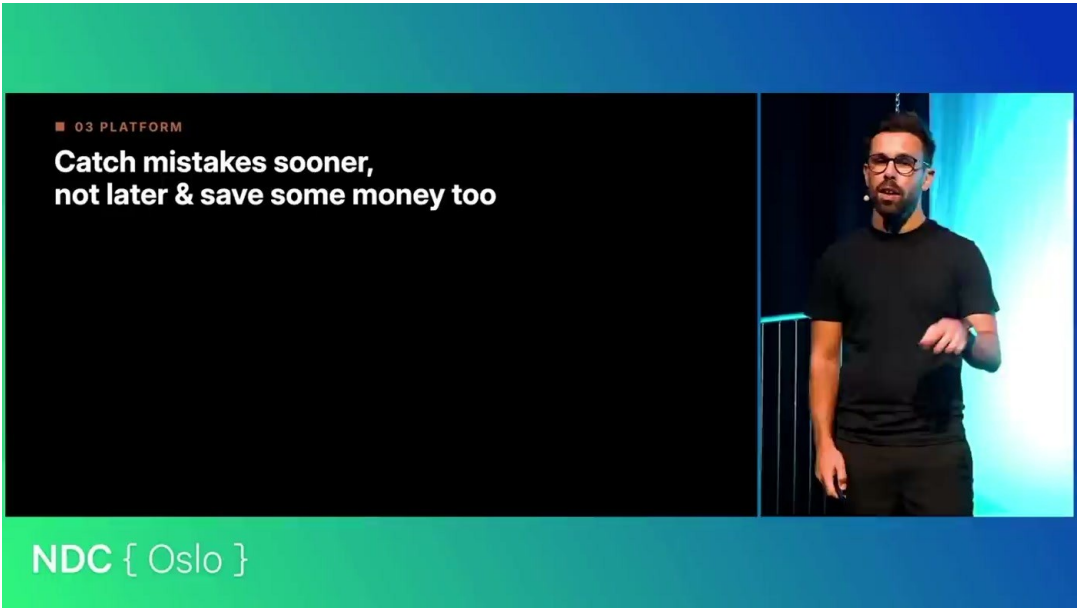
מה זה עושה ומתי	פקודה / קיצור
הדבקת צילום מסך ל-terminal. Cmd+V לא עובד	Ctrl+V (גם ב-Mac)
צבע ושם לכל שיחה כשעובדים במקביל ב-worktrees	/rename, /color
מטרה עם קריטריון אימות. Claude עובד בלולאה עד שהבדיקה עוברת	/goal
מציג רק את ה-prompt ואת התוצאה הסופית, בלי הודעות הביניים	/focus
תצוגה של כל ה-agents: מי רץ, מי מחכה לתשובה ומי סיים	claude agents

Gui Ferreira I'm not aware of any study that proves this. But there's this idea that once you get to — the 40% of the context window, you get into the dump zone

שתי יכולות שינוי לו את אופן העבודה, ולא רק חסכו זמן. הראשונה היא remote control: "If you've never tried". remote control, you should. הוא שולח משימה ארוכה, יוצא (לאסוף את אביו מבית החולים, או לטיול), וממשיך מהאפליקציה בטלפון. ה-push notifications מבקשות אישור או פידבק ומודיעות כשהעבודה הסתיימה. השנייה היא voice. כשמדברים בקול, לדבריו, "your brain works in a different way", כמו שקורה כשמסבירים בעיה לעמית.

לעבודה במקביל הוא משלב כמה כלים: git worktrees כדי לעבוד על feature, code review ותיקון באג באותו repo בו-זמנית; צבע ושם לכל שיחה; /goal עם תנאי עצירה מדיד; ו-focus/. דוגמת ה-goal/ שלו היא API שמפיק חשבונות, עם טסט שקורא ל-API ומשווה את ה-PDF שחוזר לדוגמה. כך לא צריך "babysitting in every single step".

5. The Platform Layer — נקודות הרחבה שלא הרצתם



"Catch mistakes sooner, not later & save some money too": שקופית ה-hooks בפרק 03, The Platform Layer

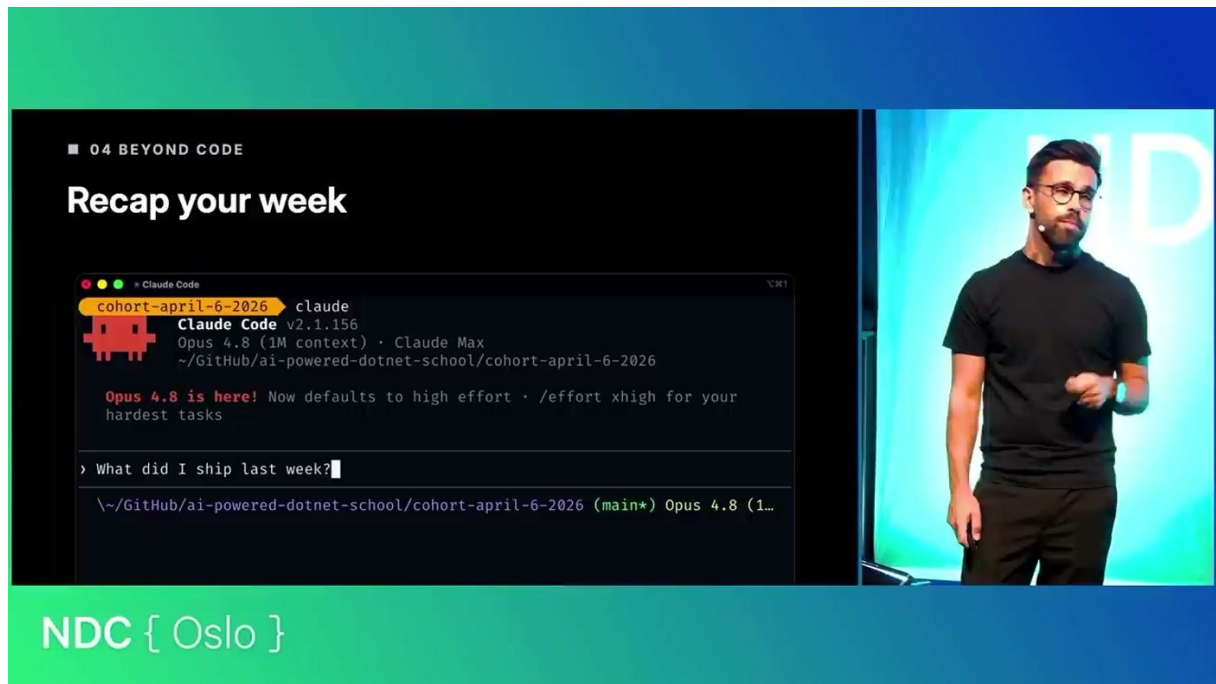
ההבחנה המרכזית בפרק היא בין הוראה לבין אכיפה. כשכותבים ב-CLAUDE.md, "always run the tests", Claude צריך להחליט לבצע את זה, וזה לא תמיד קורה. PostToolUse hook שמריץ טסטים ו-linter רץ בכל פעם, והפלט שלו חוזר לשיחה, כך ש-Claude מתקן לפני שהוא מכריז שסיים. יש לזה גם יתרון כספי: "Because running this instruction, this bash command yourself, doesn't cost anything in terms of tokens".

גם על skills הוא מציע להסתכל אחרת: "The skill.md is". just entry points skill היא תיקייה שאפשר לשים בה סקריפטים (Python, PowerShell) וקובצי markdown שמסבירים איך להשתמש בהם. דוגמה: סקריפטי תיקון שיושבים היום בדף Confluence יכולים להפוך ל-skill.

• **/batch**: מפצל refactor או migration גדולים בין כמה sub-agents שעובדים במקביל. בתמלול נשמע "slash patch".

- **simplify**: ה-skill האהוב עליו. מטפל בכפילויות ובקוד שקשה לתחזק. לדבריו, Anthropic משתמשת בו בפיתוח המוצר עצמו, ולכן כדאי לנסות אותו לפני שכותבים skill של refactor משלכם.
- **loop**: בדיקה חוזרת, למשל כל חמש דקות לבדוק אם ה-deploy ב-GitHub Actions עבר, במקום לרענן את הדף ידנית.
- **mcp.json**. בפרויקט: הגדרת ה-MCPs ב-repo ולא בפרופיל המשתמש. הקובץ נכנס ל-commit, כך שכל הצוות מקבל את אותם MCPs ואפשר לבדוק בו אילו מהם באמת בשימוש.
- **Headless mode**: ק- Claude (או --print) מאפשר להוסיף reasoning לסקריפטים קיימים, כולל piping פנימה והחוצה, במסגרת המנוי.

6. Claude — Beyond Code כעוזר אישי



"Recap your week" ב-Claude Code ב-terminal עם השאלה "What did I ship last week?". דוגמה לשימוש בהיסטוריית העבודה לסיכום תקופתי

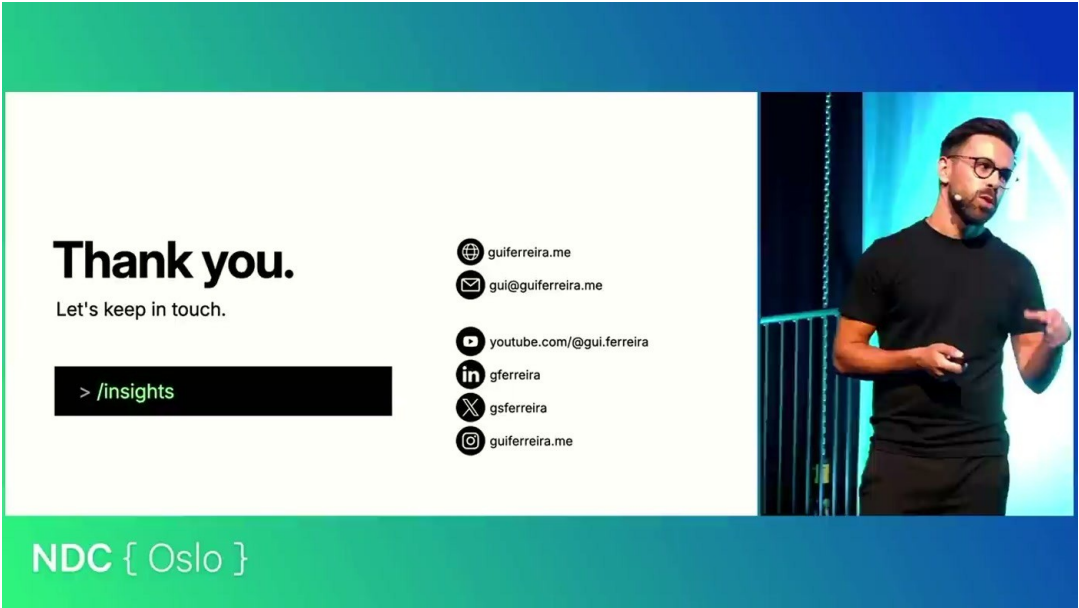
לדבריו Claude Cwork הוא שילוב של Claude Code ו-Claude.ai עם ממשק ידידותי יותר, אבל מפתחים יכולים להסתפק ב-Claude Code. הוא מציג ארבע דוגמאות:

סיכום רבעוני להערכת ביצועים. אם העבודה נעשתה עם Claude, אפשר לשאול אותו מה נבנה, ולהפוך את זה ל-skill בפרופיל המשתמש שמופעל כל רבעון. ארגון תיקיית ההורדות. היו בה 1,626 פריטים, Claude מיון אותם לתיקיות, וכך ההחלטה מה למחוק נעשתה קלה: קוד כבר נמצא ב-GitHub, צילומי מסך ו-installers נמחקים, ומסמכים נבדקים ידנית. Obsidian: vault של markdown שאפשר לעבוד עליו עם Claude Code כדי לסכם פגישות, להציף משימות לחודש הבא ולקשר בין פתקים, כשהנתונים נשארים בשליטתו ואינם תלויים בכלי מסוים. ותוכנית ריצה: בגלל מחלה כרונית נאמר לו בעבר שאסור לו לרוץ. הוא גילה שהבעיה היא ריצה לא עקבית, ובנה שני skills: אחד מושך כל יום את הריצות מ-Strava ל-vault, והשני בונה כל שבוע תוכנית שמונעת עומס יתר.

7. מה עושים ביום שני

- להריץ **insights**: זו ההמלצה שסוגרת את ההרצאה, "Slash Insights will tell you a lot about the way that you use Quoth". (Quoth הוא שיבוש תמלול של Claude).
- **לנקות את CLAUDE.md**: להריץ /doctor בפרויקט, להעביר הוראות נקודתיות ל-skills ול-rules, ולהשתמש ב-import@ במקום לשכפל תוכן.
- **לחסום סודות**: להוסיף permissions.deny ב-claude/settings.json. לקבצי env, secrets וכדומה.
- **לבנות status line**: שמציג מודל, branch ואחוז context.

- **לאבחן session ריק:** context / בתחילת עבודה. אם יותר מכמה אחוזים תפוסים כבר בהתחלה, לבדוק אילו skills ומCPs תופסים אותם.
- **להעביר בדיקות ל-hook:** PostToolUse שמריץ טסטים ו-linter, במקום הוראה ב-CLAUDE.md.
- **לנסות skills מובנים:** /simplify, /batch, ו-loop/ לפני שכותבים skill משלכם.



שקופית הסיום "Thank you. Let's keep in touch". עם הפקודה /insights ופרטי הקשר של הדובר

שאלה פתוחה	למה היא חשובה
מה בדיוק מדד מחקר ETH Zurich?	המספרים (3%- ביצועים, +20% עלות) מצוטטים בלי מקור מלא ובלי לפרט על אילו משימות נמדדו
האם ה-dump zone של 40% אמיתי?	הדובר עצמו מציין שאין מחקר שתומך בכך, ולכן זה לא בסיס טוב לכלל אצבע
עד כמה permissions.deny אוקף?	לא נאמר אם הוא חוסם גם גישה עקיפה, למשל דרך סקריפט או כלי אחר
אילו פקודות יציבות?	diff-ו claude agents /,goal, /focus, /diff הוצגו כחדשות. לא צוינו גרסה או זמינות
אבטחה ב-remote control	אישור פעולות מהטלפון ללא סקירה של ה-diff לא נדון

8. מילון מונחים

מונח	משמעות
CLAUDE.md / CLAUDE.local.md	קובץ הוראות שנטען לכל session. גרסת local נשמרת ב-gitignore וגרסת ~/.claude חלה על כל הפרויקטים
AGENTS.md	המקבילה ל-CLAUDE.md בכלים אחרים כמו Codex
permissions.deny	רשימה ב-settings.json של קבצים ופקודות ש-Claude לא ייגש אליהם
status line	שורת מצב קבועה ב-terminal שמציגה מודל, branch ואחוז context
plan mode	מצב שבו Claude חוקר ומנסח תוכנית לפני שהוא כותב קוד
rewind	חזרה להודעה קודמת בשיחה ומחיקת מה שבא אחריה מה-context
effort level / ultrathink	רמת ה-reasoning של המודל. ultrathink מפעיל max effort ל-prompt אחד
context window	חלון ההקשר שהמודל רואה. כשהוא מתמלא, האיכות יורדת והעלות עולה
compaction	החלפת היסטוריית השיחה בסיכום שלה כדי לפנות context
dump zone	השם הלא-פורמלי לטווח שמעל כ-40% ניצול context, שבו לפי דעה רווחת האיכות יורדת

מונח	משמעות
memory	העדפות ש-Claude שומר אוטומטית. כדאי לבדוק אותן מדי פעם
worktree	תיקיית עבודה נוספת של git לאותו repo, לעבודה על כמה branches במקביל
hook (PostToolUse)	פקודה שרצה אוטומטית אחרי שימוש בכלי, והפלט שלה חוזר לשיחה
skill	תיקייה עם SKILL.md, סקריפטים ותיעוד, שמגדירה יכולת ש-Claude מפעיל לפי צורך
sub-agent	agent משני ש-Claude מפעיל, לפעמים עם מודל אחר
MCP /.mcp.json	פרוטוקול לחיבור כלים חיצוניים. הקובץ מגדיר אותם ברמת הפרויקט
headless mode (-p)	הרצה לא אינטראקטיבית של Claude מתוך סקריפט
remote control	המשך של session מקומי מהאפליקציה בטלפון