

A stylized illustration of a mountain landscape. In the background, there are blue mountains with a yellow snow-capped peak under a bright orange sun. In the middle ground, a hiker in a blue silhouette with a backpack and a cap walks on a path. To the left, a person in a brown silhouette stands near some green trees. In the center, a city skyline with orange and green buildings is visible. The text 'ODTUG' is in white, 'KSCOPE' is in large white letters, and '26' is in large, colorful, multi-colored letters.

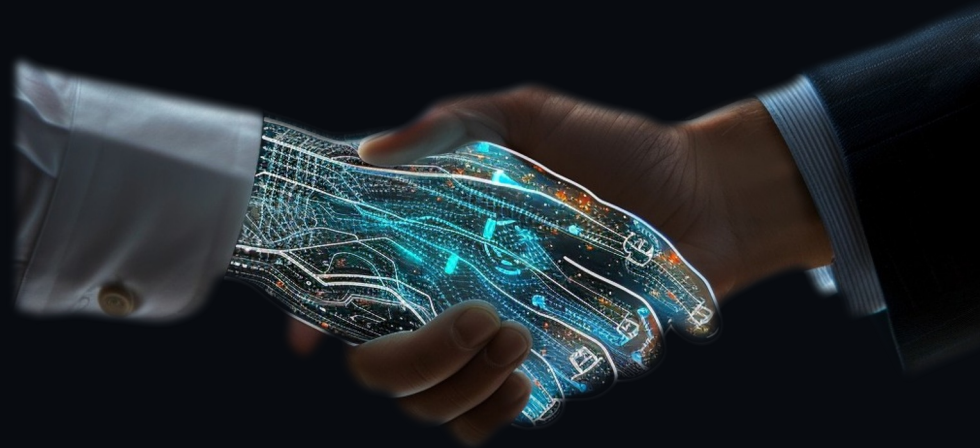
ODTUG KSCOPE 26

DENVER · JUNE 14-18

THE APEX DEVELOPER REINVENTED

From writing code to shaping systems

Dimitri Gielis
United Codes



“

The only constant in life is change.

AI can build an app in minutes.



Frontend



Backend



Database

All wired together.

Would you trust it with your company?

Would you trust it in production... on a Friday night?

It works... until it matters.



Security



Performance



Data Integrity



Maintainability

When it matters...

We don't want magic.

We want control.

Oracle APEX + Database



Reliable



Predictable



Maintainable

“Boring” is underrated.

Boring is predictable.

Boring is maintainable.

Boring is what runs your business.

This talk is NOT

- ✗ AI replacing developers
- ✗ Prompt engineering tricks
- ✗ Hype

This talk IS

- ✓ How AI changes how we build
- ✓ How our role evolves
- ✓ How we stay in control

20+ YEARS OF

APEX Evolution



Every step made us faster...

No step removed responsibility.

RECENT YEARS

AI-Assisted Development

More speed. Same responsibility.



PHASE 1 · You control everything



Autocomplete

*"I write code,
AI helps me type faster."*

Inline suggestions. You stay in control.

```
src > database > worldcup2026_app > package_bodies >  hello_api.sql > ...
```

```
11
```

```
12
```

```
|
```

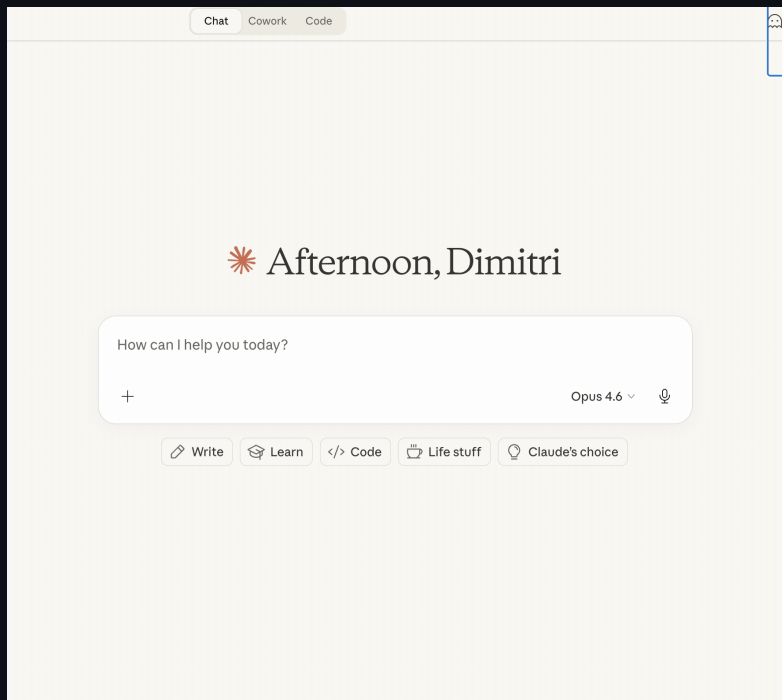
PHASE 2 · You collaborate



Conversational AI

*"I describe what I want,
AI helps me think."*

Ask questions. Generate code. Get explanations.



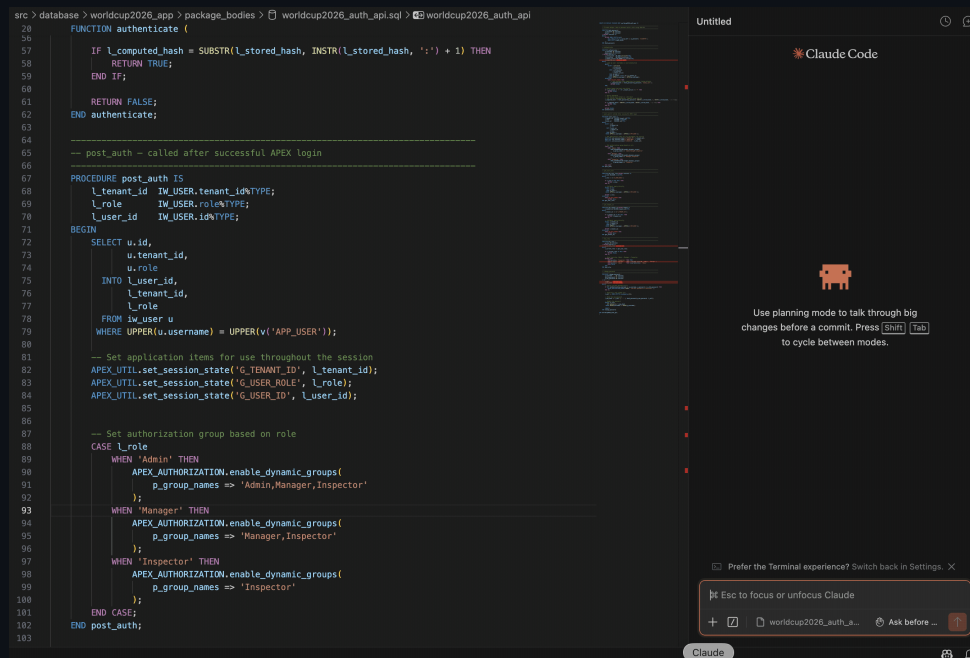
PHASE 3 · You co-build



Integrated AI

*"AI works with my code,
not just beside it."*

IDE awareness. Codebase context.
Direct refactoring.



PHASE 4 · You delegate



Agentic AI

"Build X. Fix Y."

Goal-driven. Multi-step. Uses tools.

Ex. Fix failing test → update API → regenerate UI

The screenshot displays the Claude AI interface with a task delegation workflow. At the top, tabs for 'Chat', 'Cowork', and 'Code' are visible. The main chat area shows a sequence of messages: a user prompt 'Ask Claude to create something', a system response 'context. If I don't, just pick something useful and make it for me.', a 'Thought process' section with a welcome message 'Hi Dimitri, welcome to Cowork! Let's make something together.', and a series of questions about file creation and context. The response from Claude is a detailed plan for creating an Oracle APEX app, including SQL scripts and an HTML prototype. On the right side, there are three panels: 'Progress' showing a 3-step task progress indicator, 'Working folder' with a file explorer icon, and 'Context' showing a document icon. At the bottom, there is a 'Reply...' input field, a 'Queue' button, and a version indicator 'Opus 4.6'. A footer note states: 'Claude is AI and can make mistakes. Please double-check responses. Give us feedback'.

This is not a feature evolution.

This is a responsibility shift.

Delegation requires trust.

Trust requires control.

Control requires you.

OLD ROLE

Page builder

Code writer

NEW ROLE

Designer

Orchestrator

Validator

Bad developer + AI = faster mistakes

Good developer + AI = exponential impact

AI doesn't fix bad engineering. It amplifies it.

Why APEX is perfectly positioned for AI

AI needs structure. APEX provides it.

- Data: tables, relationships, constraints
- Logic: PL/SQL, validations, processes
- UI: components, not code

The best developers don't just use AI.

They control it.

- Define boundaries
- Enforce standards
- Stay accountable

PROGRAMMING LANGUAGE

English

Intent

What you want

Constraints

What you enforce

Validation

How you define success

Ex. Not 'build a table', but 'build a secure, multi-tenant scoring system with audit trail'

Enough slides. Let's build something.



World Cup Prediction App — from idea to production

<https://dgielis.com/the-apex-developer-reinvented>

Real tools. Real workflow.



VS Code + Claude + superwhisper



AI / Dev Kanban



Docker



Database



Git



Jenkins

APEX Bootstrap to create it entirely.

Docker clean complete. Images were preserved.

[dgielis@Dimi-MBPro16-<https://github.com/dgielis/apex-developer-reinvented>

✓ Checking prerequisites 3s

✓ Preparing bootstrap runtime 0s

✓ Preparing environments 0s

✓ Starting containers 1m 7s

 Provisioning DEV 1m 17s

 Provisioning TEST 1m 17s

 Installing utPLSQL

 Deploying application

 Initializing SQLcl project

 Configuring GitLab & Jenkins

 Writing report

 Installing AI skills



Flow

1 Gather requirements

2 Design data model

3 Build PL/SQL API

4 Build APEX app

5 Deploy

6 Enhance with tickets



LIVE DEMO

Nothing disappeared.

We still think.

The Loop



AI proposes



You approve



AI builds



You validate



AI merges



You deploy



AI executes



You are accountable

Where humans matter most

If it goes wrong... it's on **YOU!** And it will go wrong.



Data modeling



Security



Production readiness



Consistency

AI accelerates.

Humans decide.

This is not about AI.

This is about who **YOU** become as a
developer.

AI won't replace you.

But **someone** using it might.

APEX + AI



Speed with control



Innovation with safety



Productivity without chaos

The future is not no-code.

It's better developers.

Thank you!

Questions?

<https://dgielis.com/the-apex-developer-reinvented>

<https://www.united-codes.com> - **dbLinter**