Progress®

Progress® OpenEdge® AI Assistant

The Past, Present and Future

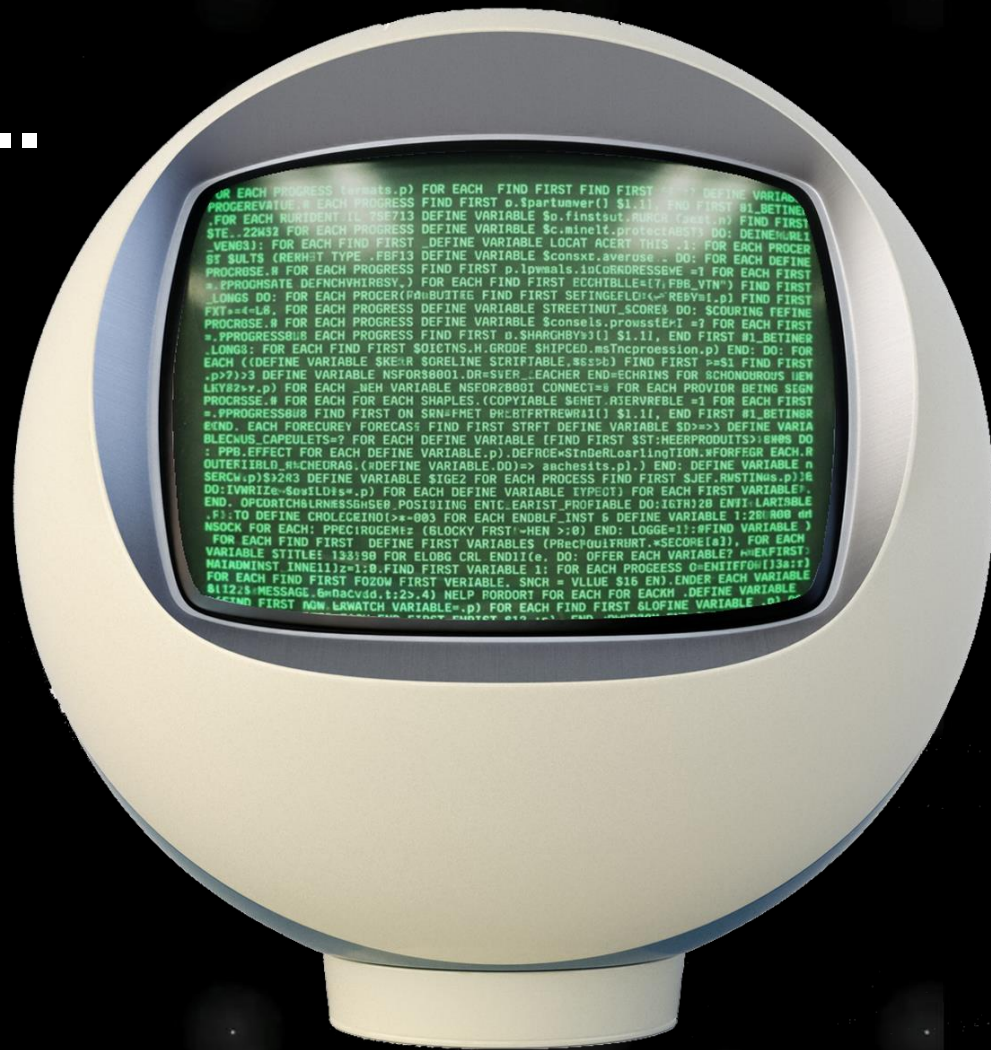


PUG Challenge 2026

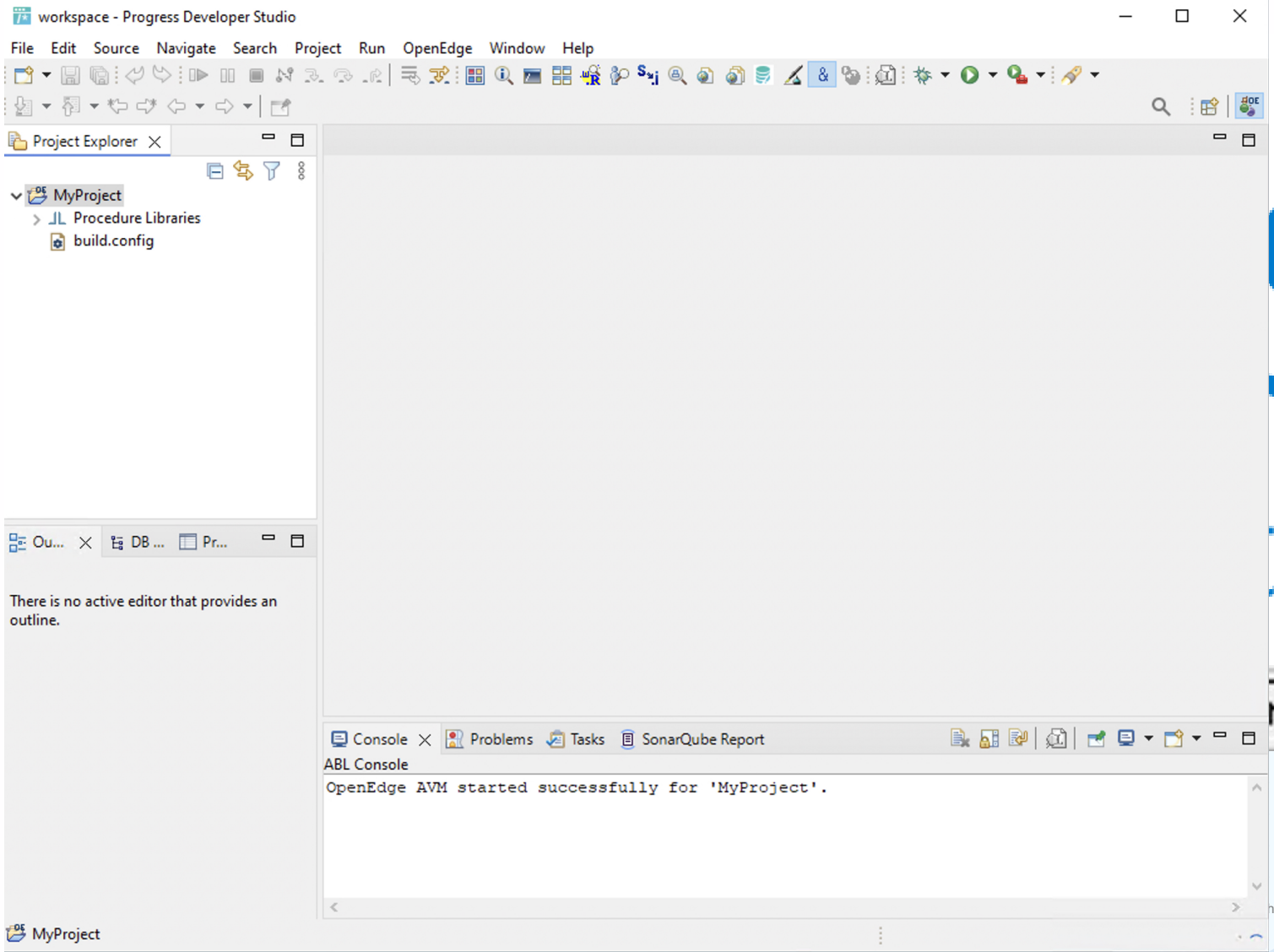
Episode I:
In a Galaxy Far, Far Away...

The Pre-AI Era

- **The Clone Wars of Boilerplate:**
  Copy and pasting the same logic 100 times
- **The Phantom Bugs:**
  Manually tracing a variable through millions of lines of procedural code
- **The Jedi Archives (Offline):**
  Relying solely on memory, dusty manuals and OpenEdge Hive for syntax
- **Development Speed:**
  Sub-light engines only



What Is Your Favorite Integrated Development Environment (IDE)?

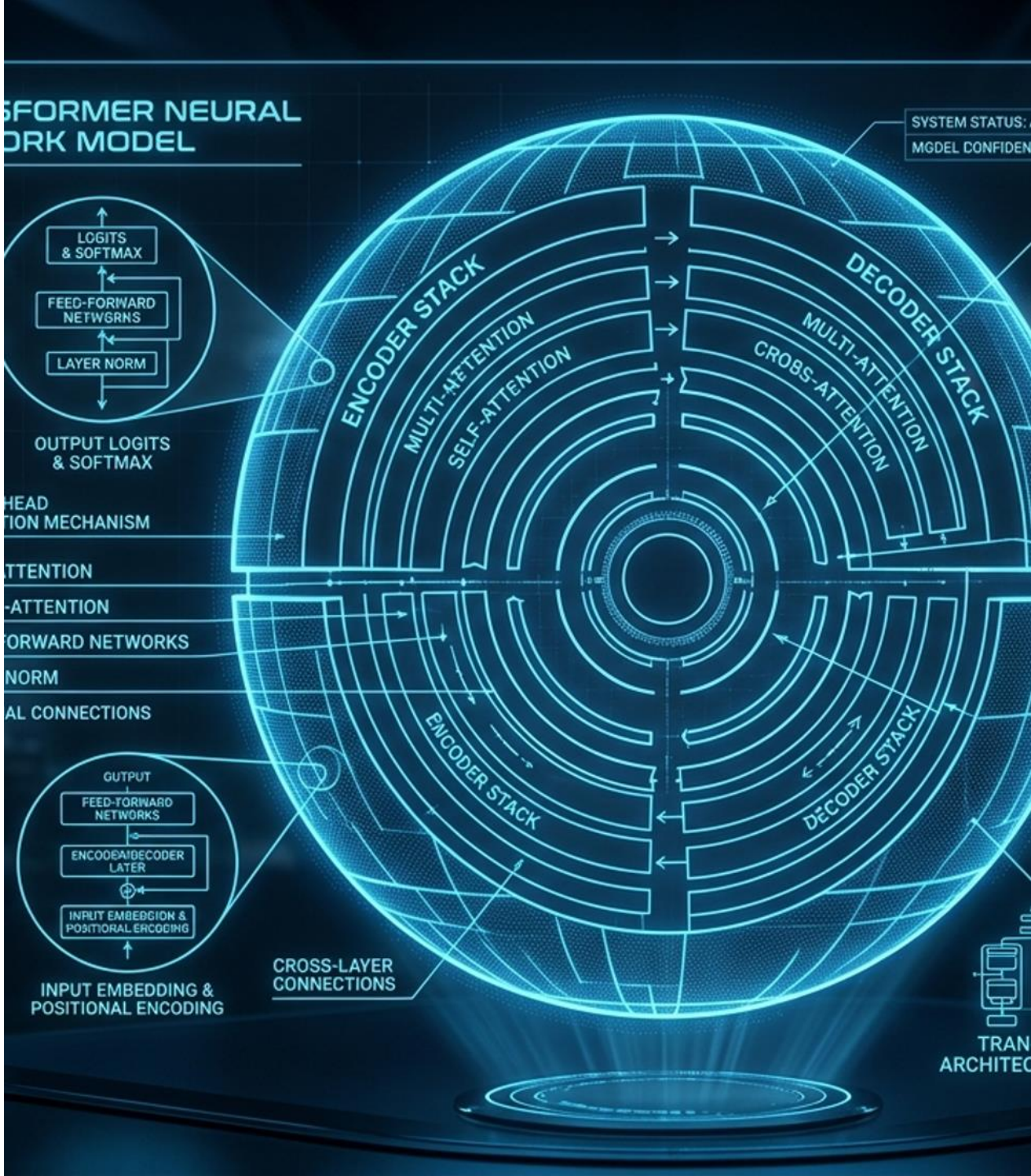




nts reserved.

Episode II:
The Dawn of the Machines

- **Not Magic, Just Math:**
  It’s not a sentient droid; it’s a Large Language Model (LLM) predicting the next token
- **The Training:**
  Feeding the model vast amounts of code to understand syntax, patterns and logic
- **From Text to Code:**
  How generic AI evolved into specialized coding assistants



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

LLMs

Vectors & Embeddings:

- Words (tokens) are converted into dense numerical lists (embeddings)
- Tokens with similar meanings are positioned closely together in a 3D "vector space“

Attention Layers:

- The model calculates relationships between all words
- Uses **Query, Key and Value (QKV)** matrices to find connections and focus on relevant context (attention weights)

Softmax & Probabilities:

- The Softmax layer converts model scores into a final probability distribution
- The model predicts the next most likely token (e.g., **"word" at 92%**) to complete the sequence



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Episode III: **A New Hope**

- **Mind-Reading (Context):**
  The AI reads your workspace and understands what you are trying to do
- **Refactoring at Lightspeed:**
  Breaking down monolithic, huge procedures into smaller, "single responsibility" procedures
- **The Ultimate Co-Pilot:**
  Writing unit tests, explaining legacy code and catching syntax errors before compilation



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.



Episode IV: **Forging the Lightsaber**

- **The Kyber Crystal:**
  Why generic AI isn't enough for specialized ABL
- **Context is Everything:**
  How we built the bridge between the LLM and the OpenEdge ecosystem
- **Teaching an Old Dog New Tricks:**
  Enabling AI to understand ABL (Temp-tables, ProDataSets & Progress® Application Server (PAS) for OpenEdge

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

VSCode + OpenEdge Platform's ABL Plugin



Context

Your codebase…

Context Example: AutoEdge | TheFactory

Create an enumeration for the 10 most popular colors.

```
using AutoEdge.Factory.Common.CommonInfrastructure.UserTypeEnum.
using OpenEdge.Lang.EnumMember.

class AutoEdge.Factory.Common.CommonInfrastructure.UserTypeEnum inherits EnumMember:
```

Use new enum syntax

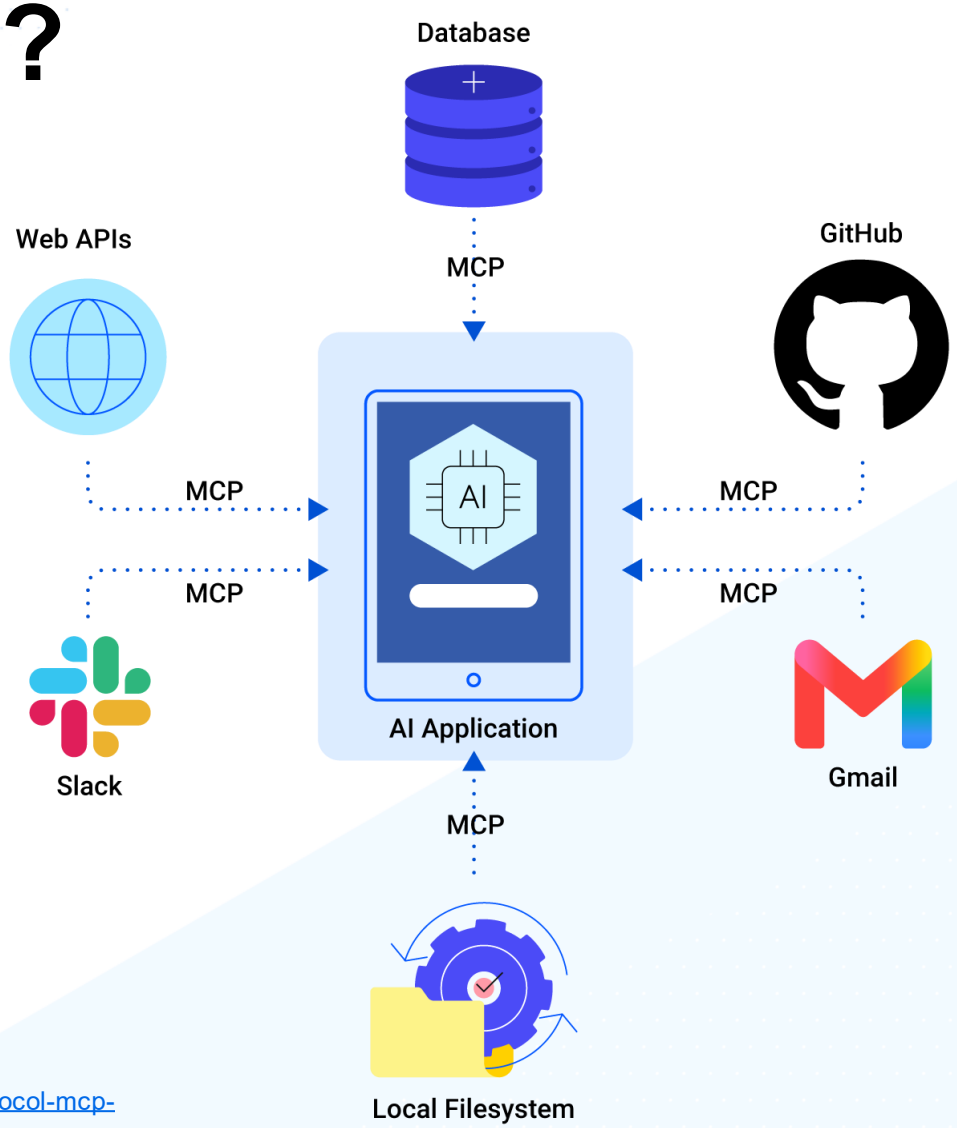
```
USING Progress.Lang.Enum.

ENUM ColorEnum:
    DEFINE ENUM Red
                Blue
                Green
                Black
                White
                Yellow
                Purple
                Orange
                Pink
                Brown.
END ENUM.
```

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.


CUSTOMER SPECIFIC RULES
CLIENT_NAME LOGIC
LEGACY INTEGRATION PROTOCOLS
CUSTOM FORMATTING PREFERENCES
PRIVATE API ACCESS
DESIGN PATTERNS
ARCHITECTURE RULES
MODULE CONNECTIONS
CODEBASE SETUP PROTOCOLS
FRAMEWORK-SPECIFIC PRACTICES
FRAMEWORK RULES
DATA TYPES & PRIMITIVES
ABL SYNTAX DEFINITIONS
STANDARD CODING PRACTICES (PROGRESS)
CORE LOGIC & FUNCTIONS
BASE RULES (PROGRESS ABL)

What is MCP?



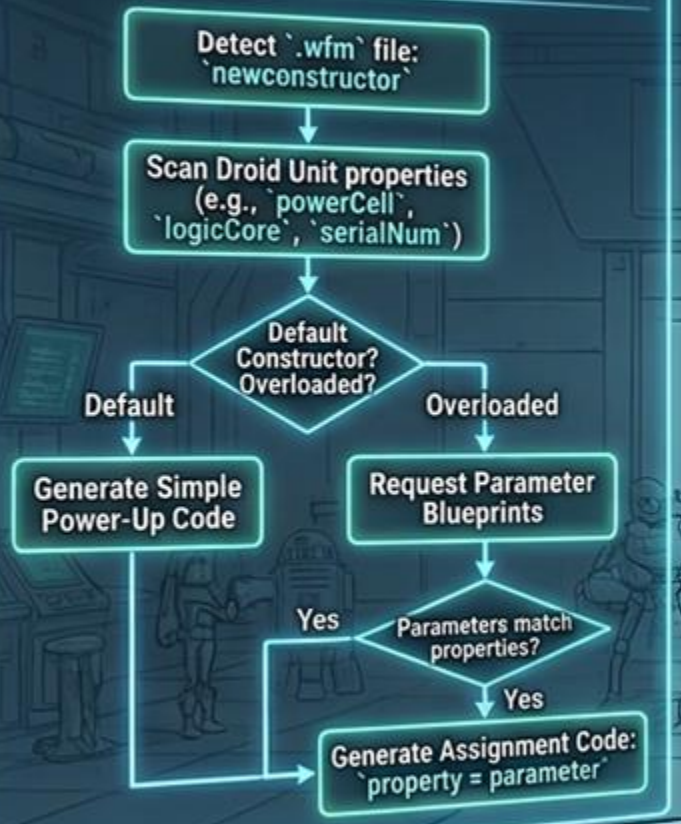
https://medium.com/@techtopiaa/the-model-context-protocol-mcp-in-ai-current-trends-facts-and-future-ba01ecb70b1b

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

AI Workflow

Agent specific implementations

METHOD 1: DEVIN - WORKFLOW MARKDOWN FILE



METHOD 2: CLAUDE CODE - COMMAND MARKDOWN FILE

commands.md

```
# newconstructor

$ASK Droid Unit Class
$ASK Constructor Type

$IF OVERLOADED
  $ASK Constructor Type ``Overloaded` Type
BRANO

  $CODEGEN_CONSTRUCTOR custom `newconstructor`
```

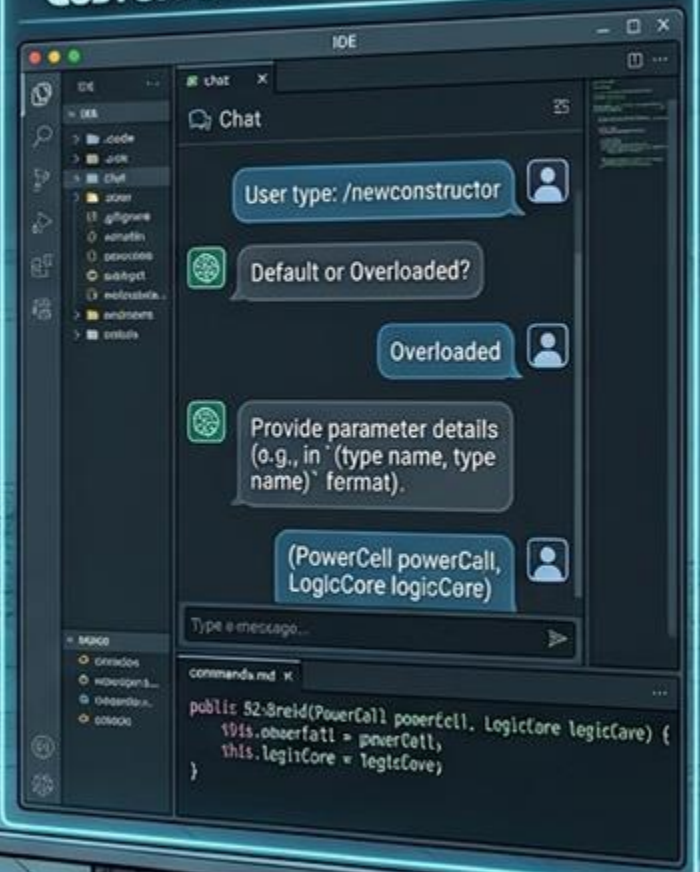
Terminal

```
Executing `newconstructor`...

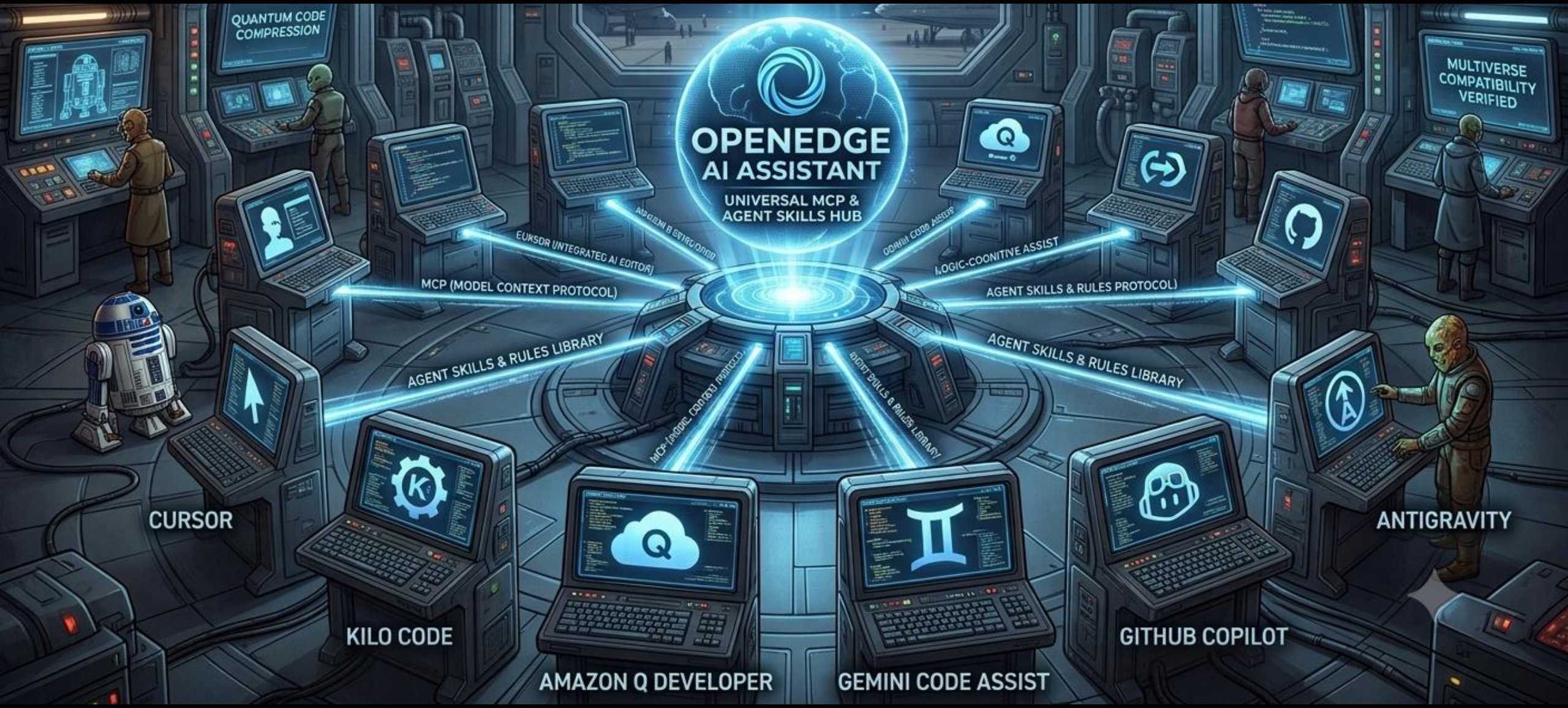
Droid Unit Class: R2-Series
Constructor Type: (O)verloaded
Parameter Blueprints: powerCell, logicCore
Analysis: 2 properties matched, 1 unmatched

Generating code snippet...
```

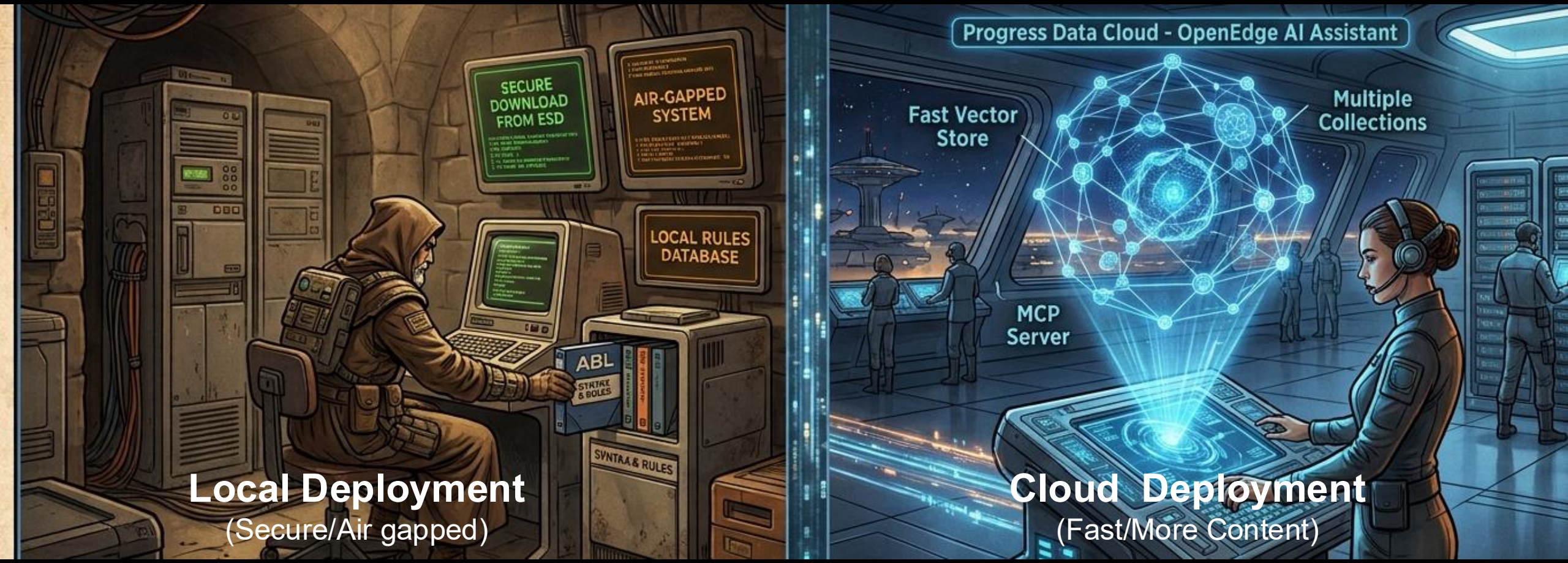
METHOD 3: GITHUB COPILOT - CUSTOM SLASH COMMAND PROMPT



A Generic Solution



OpenEdge AI Assistant Deployment Options



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Local Deployment



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

The ABL Context Files



Extract the ABL_Context.zip

Into your project

On a network share



Add the rules to your IDE

.cursor/rules/abl-syntax.mdc

.devin/rules/abl-syntax.md

./CLAUDE.md



Change the rules **to** point to the (relative) location of the syntax files

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.



```
BLOCK-LEVEL ON ERROR UNDO, THROW statement

Use this statement in a procedure (.p) or class (.cls) file to change the default ON ERROR directive to ON ERROR UNDO, THROW for all blocks

Syntax

    BLOCK-LEVEL ON ERROR UNDO, THROW.


This statement affects the following block types:
•   Procedure (also called main block, external procedure, or .p file, including database trigger .p files)
•   Internal procedures
•   User-defined functions
•   Methods of a class
•   Class constructors
•   Property accessors
•   ON blocks used as database triggers with CREATE, DELETE, WRITE or ASSIGN events
•   REPEAT blocks
•   FOR blocks
•   DO TRANSACTION blocks

This statement does not affect:
•   Destructors
•   Error directives that are explicitly coded in individual, non routine-level blocks
•   ON blocks that are UI triggers.

Note these alternatives to the BLOCK-LEVEL ON ERROR UNDO, THROW statement:
•   Instead of adding the statement to source-code files, you can use the -undothrow 2 startup parameter to change
    the default error-handling to UNDO, THROW on every block affected by the BLOCK-LEVEL statement during
    compilation. See the Startup Command and Parameter Reference for more information.
•   The ROUTINE-LEVEL ON ERROR UNDO, THROW statement on page 895 can be used if you want to change the
    default error-handling only on routine-level blocks. (You can use the -undothrow 1 startup parameter to change
    the default error handling on routine-level blocks to UNDO, THROW during compilation.)

Example
An error propagates from the DO TRANSACTION block to the internal procedure, from the internal procedure up to the main .p file (b-BLOCK-LEVEL-01.p), and

   b-BLOCK-LEVEL-01.p

      BLOCK-LEVEL ON ERROR UNDO, THROW.
```

abl-syntax.mdc

.cursor > rules > abl-syntax.mdc

```
---
description:
globs:
alwaysApply: true
---
#User interface
Always use the ABL_Context/user_interface_statements.txt for ABL user interface statements.

#Transaction
Always use the ABL_Context/transaction_statements.txt for ABL transaction syntax statements.

#String manipulation
Always use the ABL_Context/string_manipulation.txt for ABL string manipulation syntax.

#Memory manipulation
Always use the ABL_Context/memory_manipulation.txt for ABL memory manipulation syntax.

#Punctuation and special characters
Always use the ABL_Context/punctuation_and_special_characters.txt for ABL punctuation and special characters syntax.

#Query manipulation
Always use the ABL_Context/query_manipulation.txt for ABL query manipulation syntax.

#Dynamic statements
Always use the ABL_Context/dynamic_statements.txt for ABL dynamic statements.

#Logical data (datasets, temp-tables)
Always use the ABL_Context/logical_data_(datasets_temp-tables).txt for ABL logical data manipulation using datasets and temp-tables.

#Widget reference
Always use the ABL_Context/widget_reference.txt for usage of ABL widgets.

#Handle reference
Always use the ABL_Context/handle_reference.txt for usage of ABL handle-based objects.

#Handle Attributes and Methods
Always consider the ABL_Context/handle_attributes_and_methods_reference_introduction.txt for an introduction to handle attributes, handle
methods, COM object properties, and COM object methods.
```

During Inference

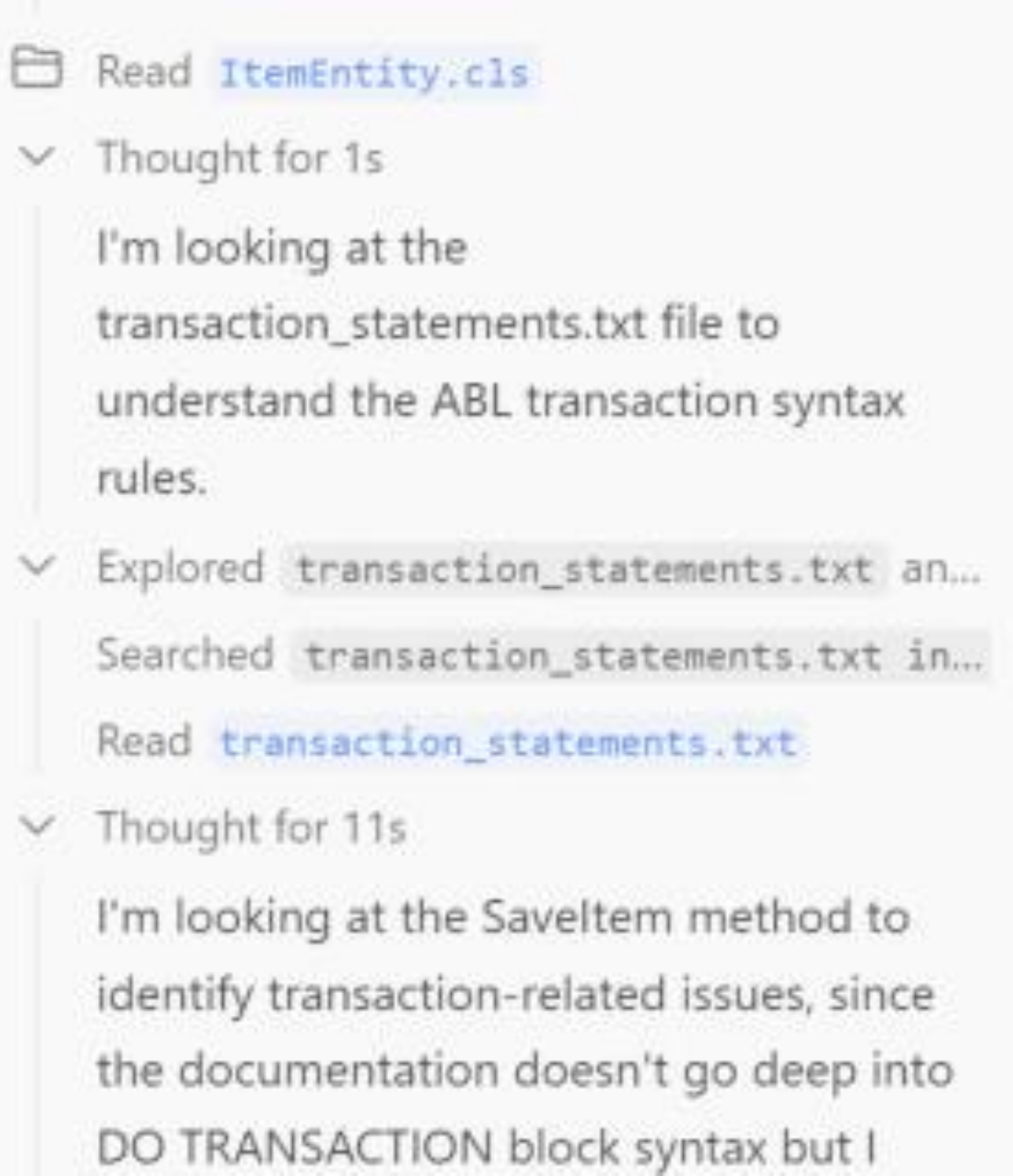
The agent will read and parse the specific text file based on the rules.

Pros:

- Air-gapped
- Secure

Cons:

- Slower
- Needs deployment



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Cloud Deployment – Prerequisites

Before you begin, you must meet the following prerequisites:

- Have a valid OpenEdge software license for version 12.2, 12.8 or 13.0
- Have an active maintenance contract

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Cloud Deployment – Customer Validation Program

Register for the OpenEdge Customer Validation Program (CVP)

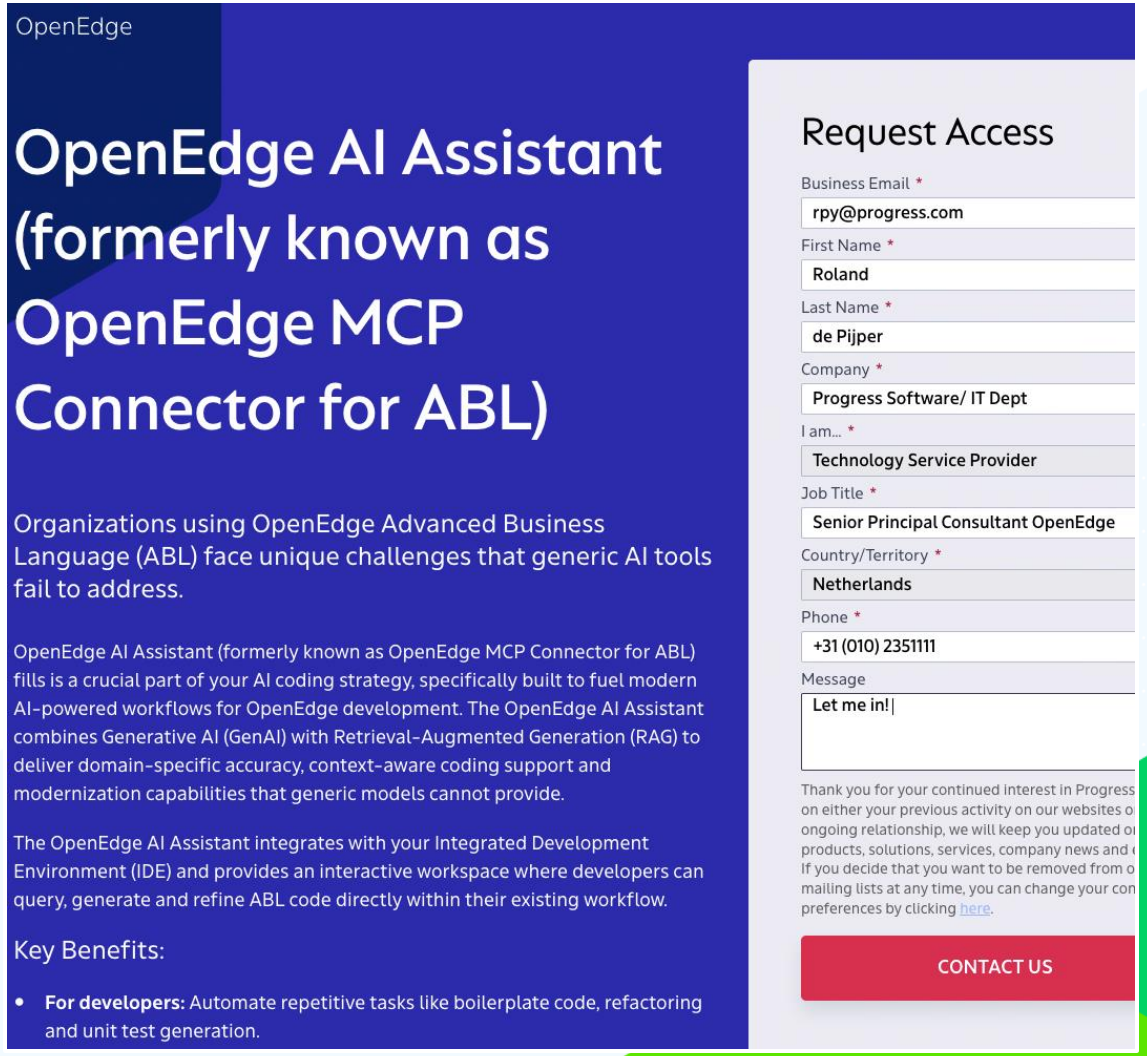


Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Cloud Deployment – Progress Data Cloud

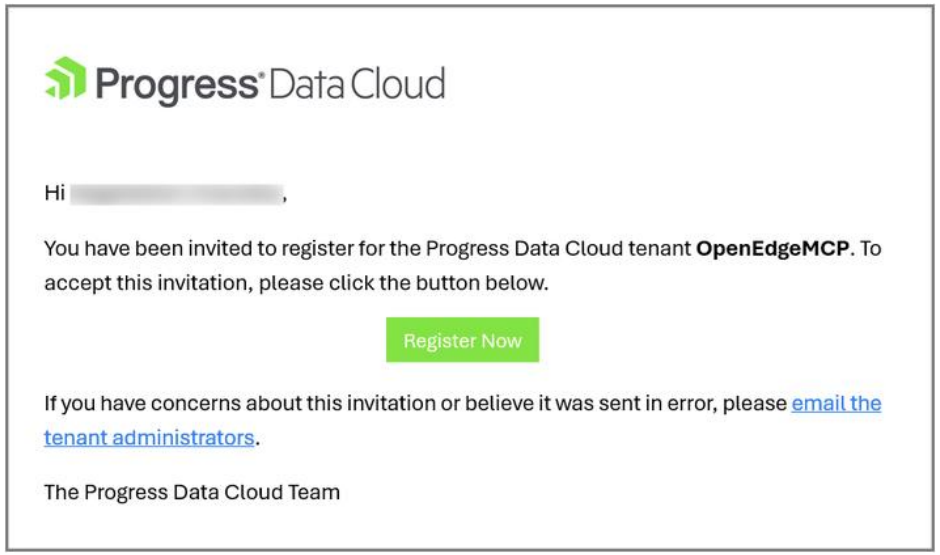
Request access to Progress® Data Cloud℠ Software

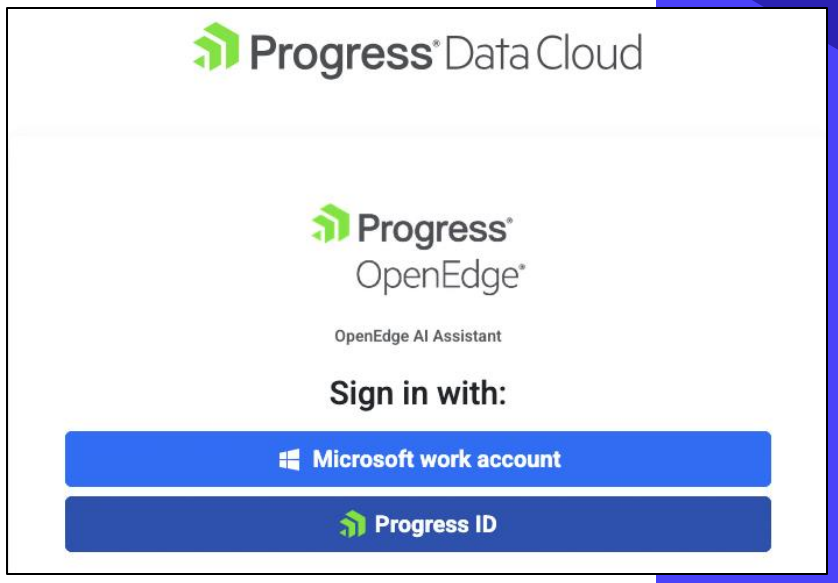


© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Cloud Deployment

Accept the invitation and sign-in on PDC with your Progress ID

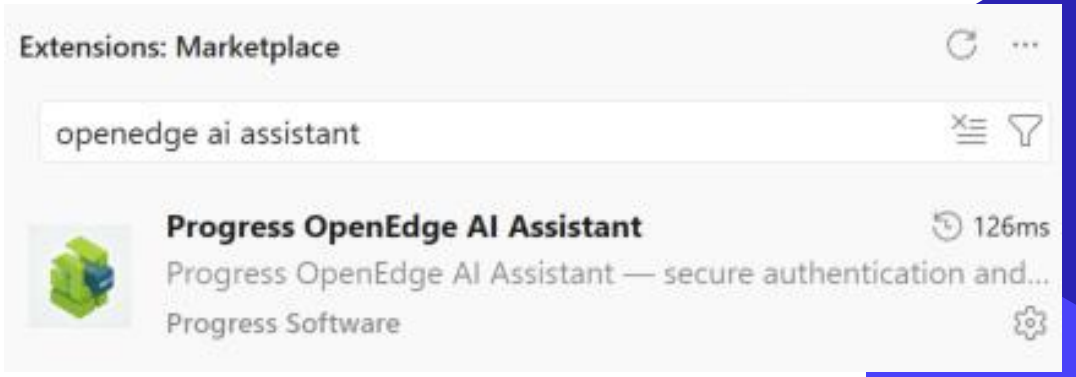


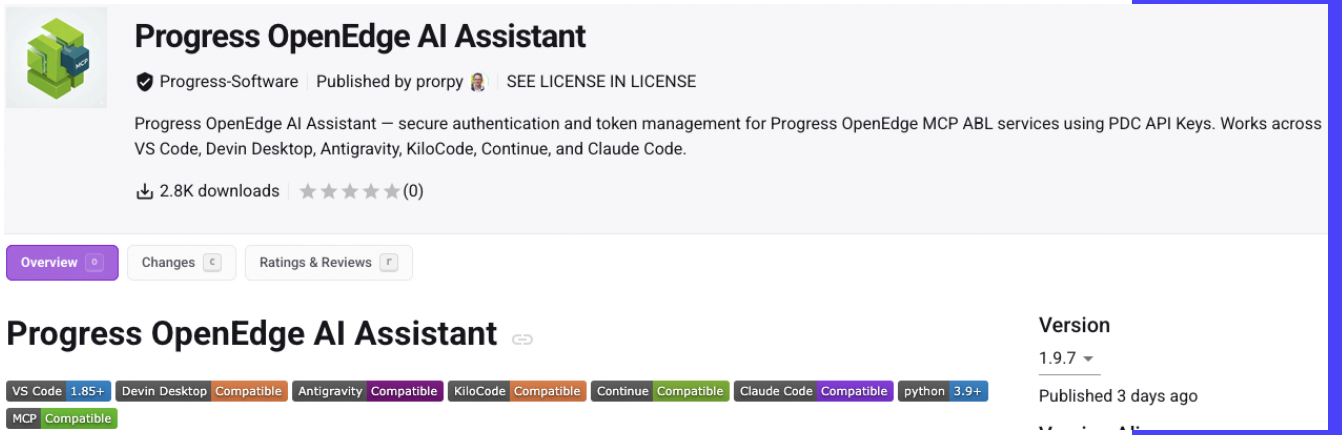


© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Cloud Deployment – IDE Plugin

Install the OpenEdge AI Assistant plugin

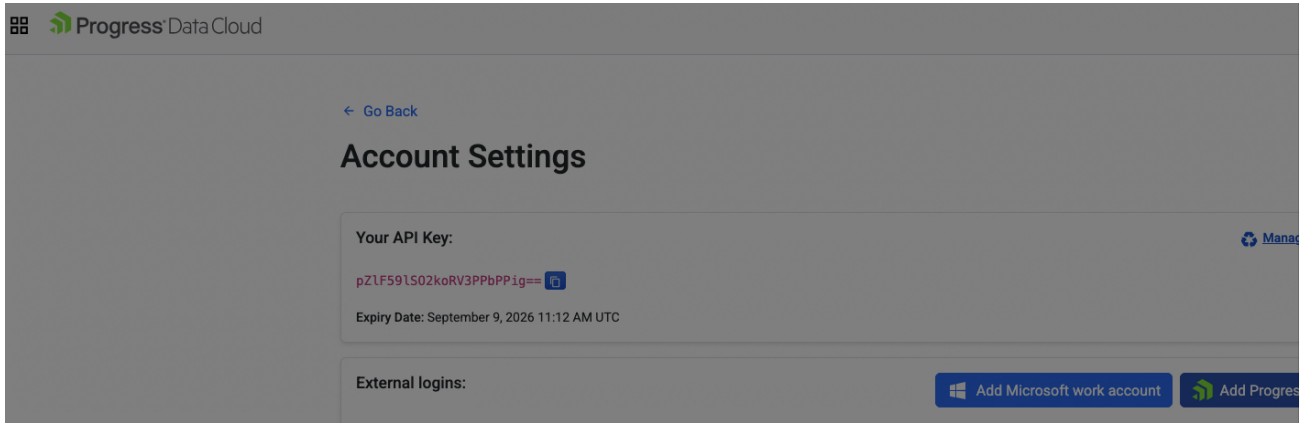


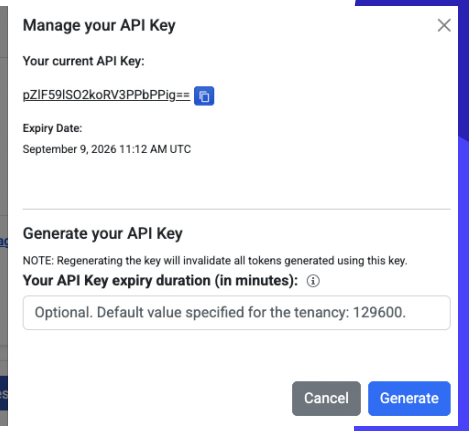


© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

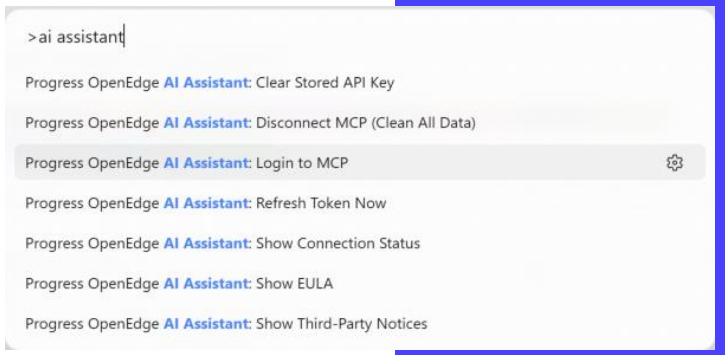
Cloud Deployment

Generate an API key on Progress Data Cloud





Login to Model Context Protocol (MCP) on your IDE (use CTRL-SHIFT-P for the popup menu)



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Cloud Deployment – Rules

Add the rules to your IDE

.cursor/rules/abl-syntax.mdc
.devin/rules/abl-syntax.md
./CLAUDE.md

CLAUDE.md U ●

CLAUDE.md > abc ## Technology Stack

```
1  ## Technology Stack
2
3  - OpenEdge ABL (Progress)
4  - - Important:  When assisting with any ABL coding task – writing new code, reviewing existing code, answering syntax
   or API questions, or generating code examples – always use the OpenEdge AI Assistant (`openedge-abl-mcp-server`) to
   look up accurate language documentation, syntax, and examples. Do not rely solely on training knowledge for ABL
   language details; consult the MCP tools first to ensure correctness.
5  - - Important: For issues related to the OpenEdge database, AppServer, configuration, or system/environment errors (e.
   g. broker startup failures, lock files, licensing), query the `KnowledgeBase` collection via the
   `openedge-abl-mcp-server` MCP tools before answering, to check for relevant Progress KB articles.
6  - - Version: 12.8
```


© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

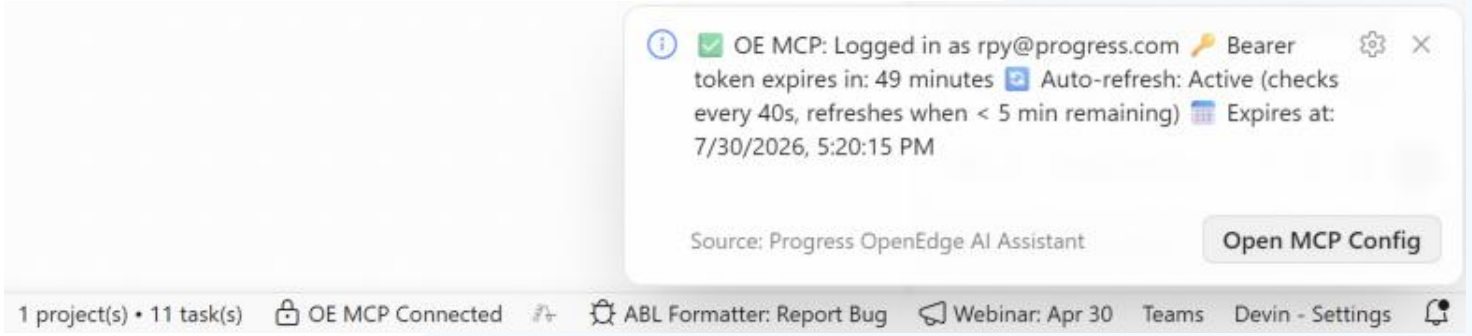
All Set!

Which collections are in the OpenEdge AI Assistant

The OpenEdge AI Assistant currently has these collections:

1. `12.2`
2. `latest`
3. `12.8`
4. `KnowledgeBase`

Total: 4 collections.



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

During Inference

The agent will call out to the MCP server (AI Assistant) based on the rules.

Pros:

- Fast
- Token efficient

Cons:

- Requires internet

Ran `query_parameters_only` – openedge-abl-mcp-server (MCP Server)

Input

```
{
  "collection_name": "12.8",
  "query_texts": [
    "READ-XML dataset parameters source
    merge-mode schema-location"
  ]
}
```

Output

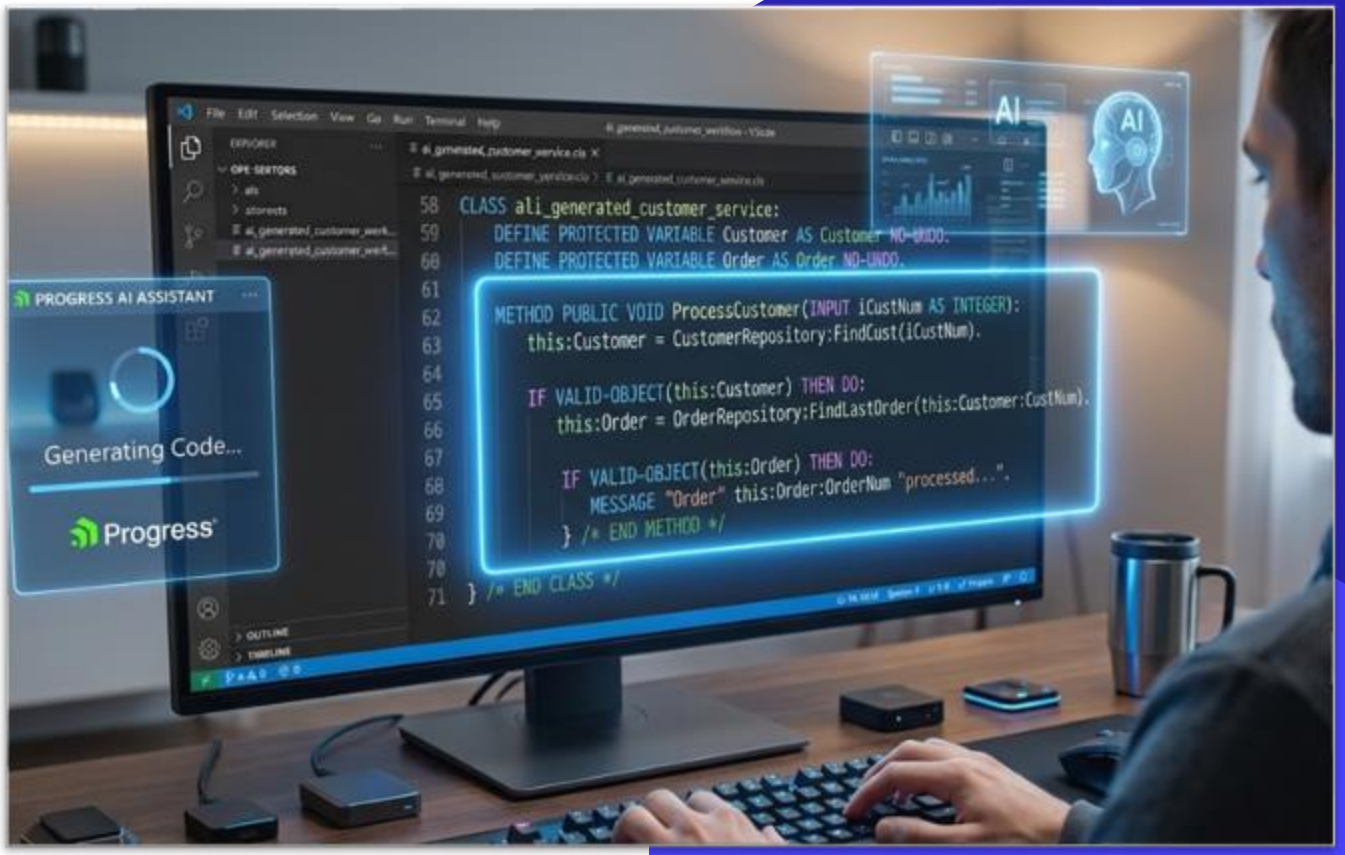

```
"n_results":3,"results":{"ids":
[["ADD-SCHEMA-LOCATION( ) method",
"READ-XML( ) method","SCHEMA-LOCATION
attribute"]],"metadatas":[[{"type":"#
Handle Attributes and Methods Reference",
"source":"Handle_Attributes_and_Methods_R
eference_-_ADD-SCHEMA-LOCATION_OPEN-PAREN
THESIS__CLOSE-PARENTHESIS_method.md",
"title":"ADD-SCHEMA-LOCATION( ) method",
"OpenEdgeVersion":"12.8"},
{"source":"Handle_Attributes_and_Methods_
Reference_-_READ-XML_OPEN-PARENTHESIS__CL
OSE-PARENTHESIS_method.md",
```


Progress

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Episode V: **Using the Force**

- **Do, or Do Not. There Is No Try:** Best practices for Prompt Engineering
- **The Jedi Mind Trick:** Being specific with your context and instructions
- **Trust, but Verify:** The AI is your Padawan—it writes the code but you are the Master who reviews it



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Demo

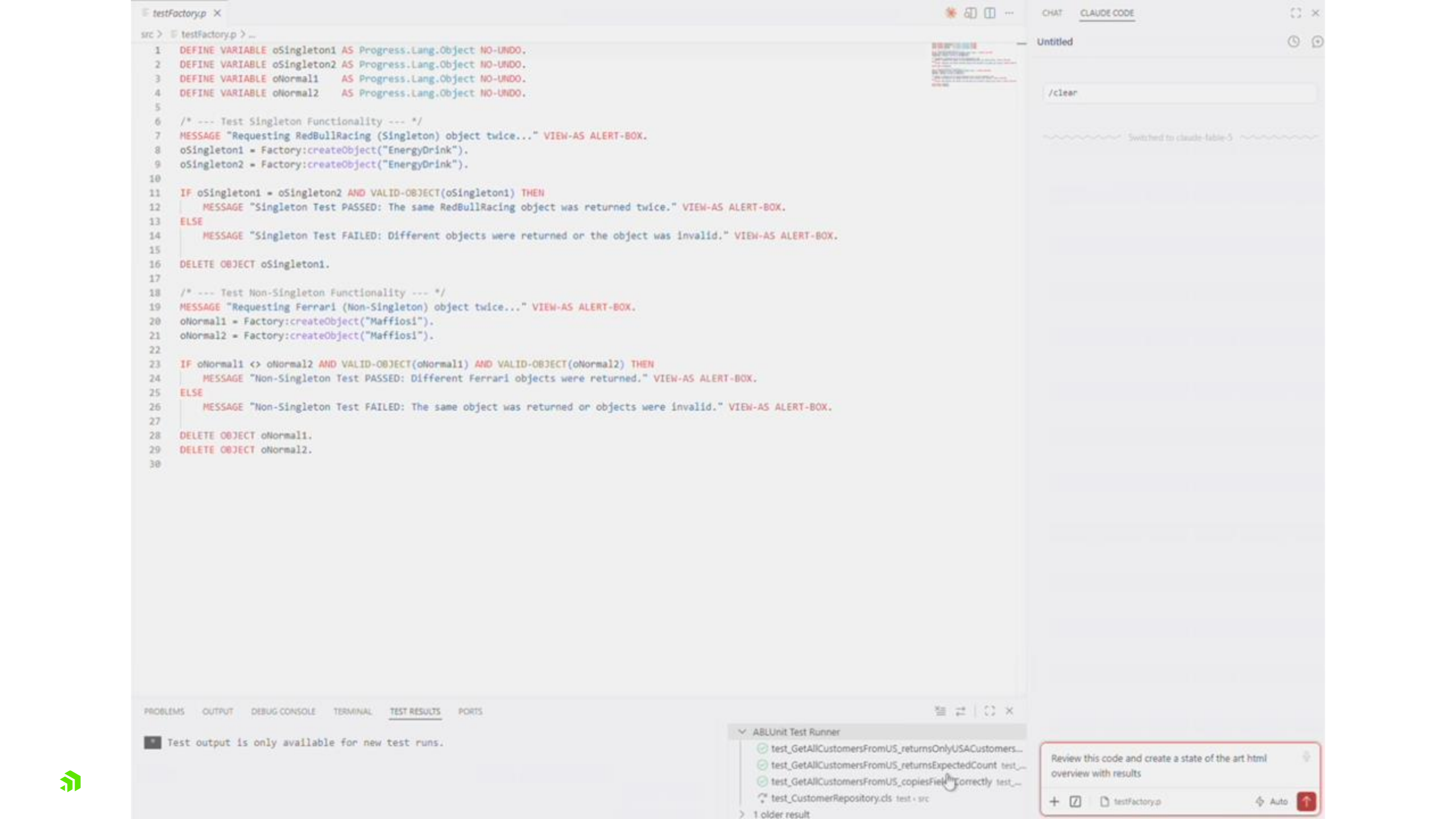
Implement a new class using a ListCollection

- Needs the AI Assistant for syntax
- Will use skills to compile

Validate correctness

- Code review skill


File Edit Selection View Go Run Terminal Help
ProjectGengar [Administrator]
EXPLORER
PROJECTGENGAR
.builder
.claude
commands
skills
worktrees
settings.local.json
.github
.windsurf
ablunit
conf
db
doc
docs
dump
powershell
rcode
src
Alpine.cls
AstonMartin.cls
Car.cls
F1Car.cls
Ferrari.cls
E:\workspace\ProjectGengar\src\F1Car.cls
getZandvoortResults.p
McLaren.cls
Mercedes.cls
RedBullRacing.cls
Zandvoort.cls
.gitattributes
.gitignore
ablunit.json
build.xml
CLAUDE.md
dbconn.pf
openedge-project.json
results.xml
updates.log
Zandvoort.cls
src > Zandvoort.cls > Zandvoort
1 CLASS Zandvoort:
2 DEFINE TEMP-TABLE ttCar
3 FIELD Position AS INTEGER
4 FIELD objCar AS progress.lang.object.
5
6 METHOD PUBLIC VOID Add(ipiPosition AS INTEGER, ipoCar AS progress.lang.object).
7 CREATE ttCar.
8 ttCar.Position = ipiPosition.
9 ttCar.objCar = ipoCar.
10 END METHOD.
11
12 METHOD PUBLIC progress.lang.object GetCar(ipiPosition AS INTEGER).
13 FIND ttCar WHERE ttCar.Position = ipiPosition.
14 RETURN ttCar.objCar.
15 END METHOD.
16 END CLASS.
17
CHAT
CLAUDE CODE
Untitled
Claude Code
Use Claude Code in the terminal to configure MCP servers. They'll work here, too!
Tackle your toughest work with Opus 5. Switch anytime with /model.
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL TEST RESULTS PORTS
OEAdmin@claude-1 MINGW64 /e/workspace/ProjectGengar (Step-4)
$ git switch -f Step-4
OEAdmin@claude-1 MINGW64 /e/workspace/ProjectGengar (Step-4)
$ git switch -f Step-4
powersh...
PowerShell ...
bash


testFactory.p
src > testFactory.p > ...
1 DEFINE VARIABLE oSingleton1 AS Progress.Lang.Object NO-UNDO.
2 DEFINE VARIABLE oSingleton2 AS Progress.Lang.Object NO-UNDO.
3 DEFINE VARIABLE oNormal1 AS Progress.Lang.Object NO-UNDO.
4 DEFINE VARIABLE oNormal2 AS Progress.Lang.Object NO-UNDO.
5
6 /* --- Test Singleton Functionality --- */
7 MESSAGE "Requesting RedBullRacing (Singleton) object twice..." VIEW-AS ALERT-BOX.
8 oSingleton1 = Factory:createObject("EnergyDrink").
9 oSingleton2 = Factory:createObject("EnergyDrink").
10
11 IF oSingleton1 = oSingleton2 AND VALID-OBJECT(oSingleton1) THEN
12 MESSAGE "Singleton Test PASSED: The same RedBullRacing object was returned twice." VIEW-AS ALERT-BOX.
13 ELSE
14 MESSAGE "Singleton Test FAILED: Different objects were returned or the object was invalid." VIEW-AS ALERT-BOX.
15
16 DELETE OBJECT oSingleton1.
17
18 /* --- Test Non-Singleton Functionality --- */
19 MESSAGE "Requesting Ferrari (Non-Singleton) object twice..." VIEW-AS ALERT-BOX.
20 oNormal1 = Factory:createObject("Maffios1").
21 oNormal2 = Factory:createObject("Maffios1").
22
23 IF oNormal1 <> oNormal2 AND VALID-OBJECT(oNormal1) AND VALID-OBJECT(oNormal2) THEN
24 MESSAGE "Non-Singleton Test PASSED: Different Ferrari objects were returned." VIEW-AS ALERT-BOX.
25 ELSE
26 MESSAGE "Non-Singleton Test FAILED: The same object was returned or objects were invalid." VIEW-AS ALERT-BOX.
27
28 DELETE OBJECT oNormal1.
29 DELETE OBJECT oNormal2.
30
CHAT
CLAUDE CODE
Untitled
/clear
Switched to claude-table-5
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
TEST RESULTS
PORTS
Test output is only available for new test runs.
ABLUnit Test Runner
test_GetAllCustomersFromUS_returnsOnlyUSACustomers...
test_GetAllCustomersFromUS_returnsExpectedCount test_...
test_GetAllCustomersFromUS_copiesFieldCorrectly test_...
test_CustomerRepository.cls test • src
1 older result
Review this code and create a state of the art html overview with results
testFactory.p
Auto

AI Accelerator Workshop

Accelerate Adoption in Three Days with Professional Services



Set Up

Set up your Integrated Development Environment (IDE)



Configure

Configure AI-powered tooling



Integrate

Integrate the Connector so AI assistance works inside your daily development workflow



Enable

Lead tailored, hands-on enablement so developers can confidently use AI for everyday work

Engagement Plan

Day 1

Environment Set-up and Plug-in Based Configuration

- Install and verify your IDE on developer machines
- Configure the ABL plug-in with proper authentication and settings
- Connect and validate integration with the connector for real-time support
- Provide an initial walkthrough of your IDE and plug-in functionality

Day 2

AI Workshop and Usage Training

- Provide covering an overview of AI capabilities in the IDE and best practices for AI-assisted development, including (prompting, code generation, refactoring), and practical workflows

Day 3

Customization and Knowledge Transfer

- Configure the AI coding assistant for your environment, including repository access, coding standards/naming conventions, architectural patterns, business logic examples, etc.
- Assist with prompt-engineering strategies tailored to your workflows
- Help you to choose an IDE that best fits your requirements
- Perform a final validation and documentation handoff
- Hold a Q&A session

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

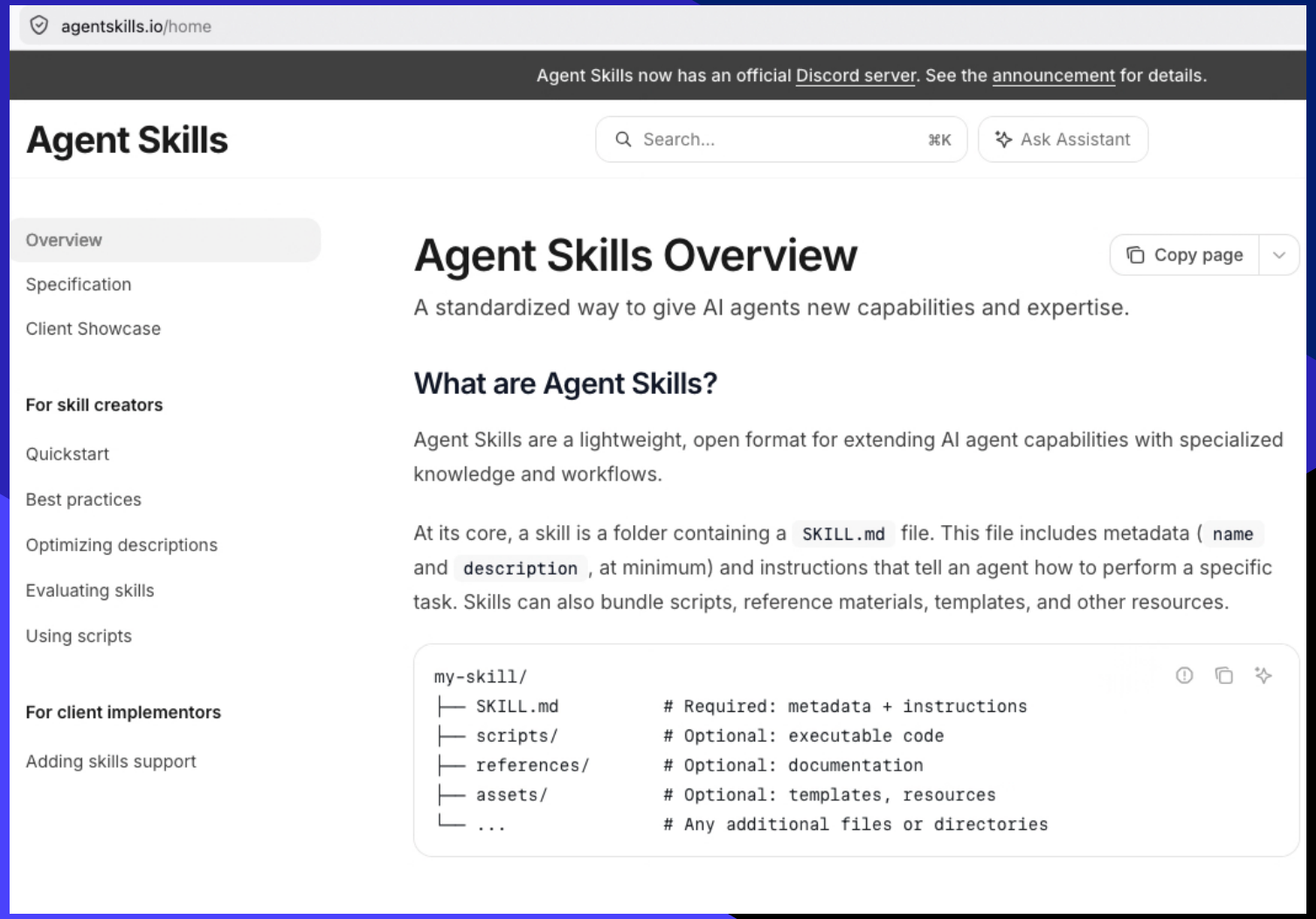
Extending with ABL-Specific Agent Skills



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

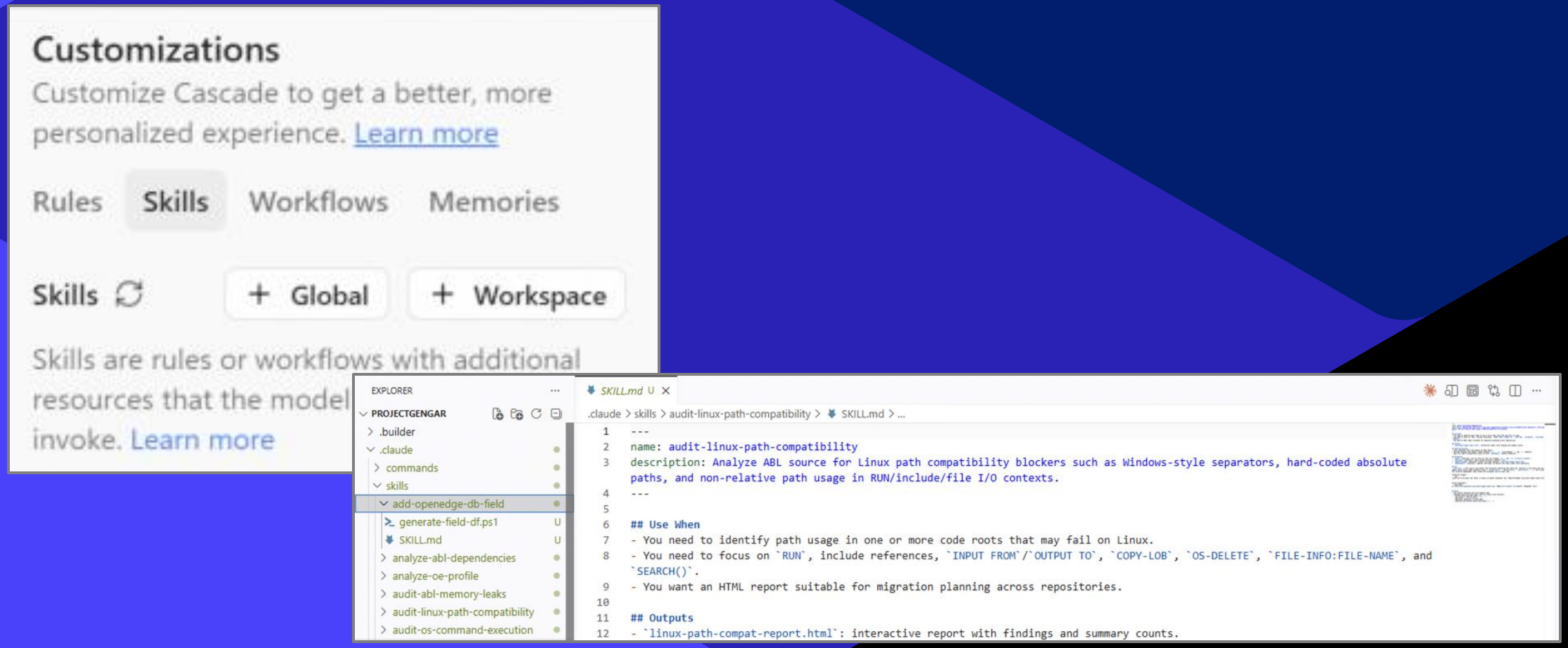
What Are Agent Skills?



Progress

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

What Are Agent Skills?



Progress

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Why Should I Use Agent Skills?

Agents load skills through **progressive disclosure** in three stages:

- **Discovery**: At startup, agents load only the name and description of each available skill—just enough to know when it might be relevant
- **Activation**: When a task matches a skill's description, the agent reads the full SKILL.md instructions into context
- **Execution**: The agent follows the instructions, optionally executing bundled code or loading referenced files as needed

Full instructions load only when a task calls for them, so agents can keep many skills on hand with only a small context footprint

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

The PS Skills Repository

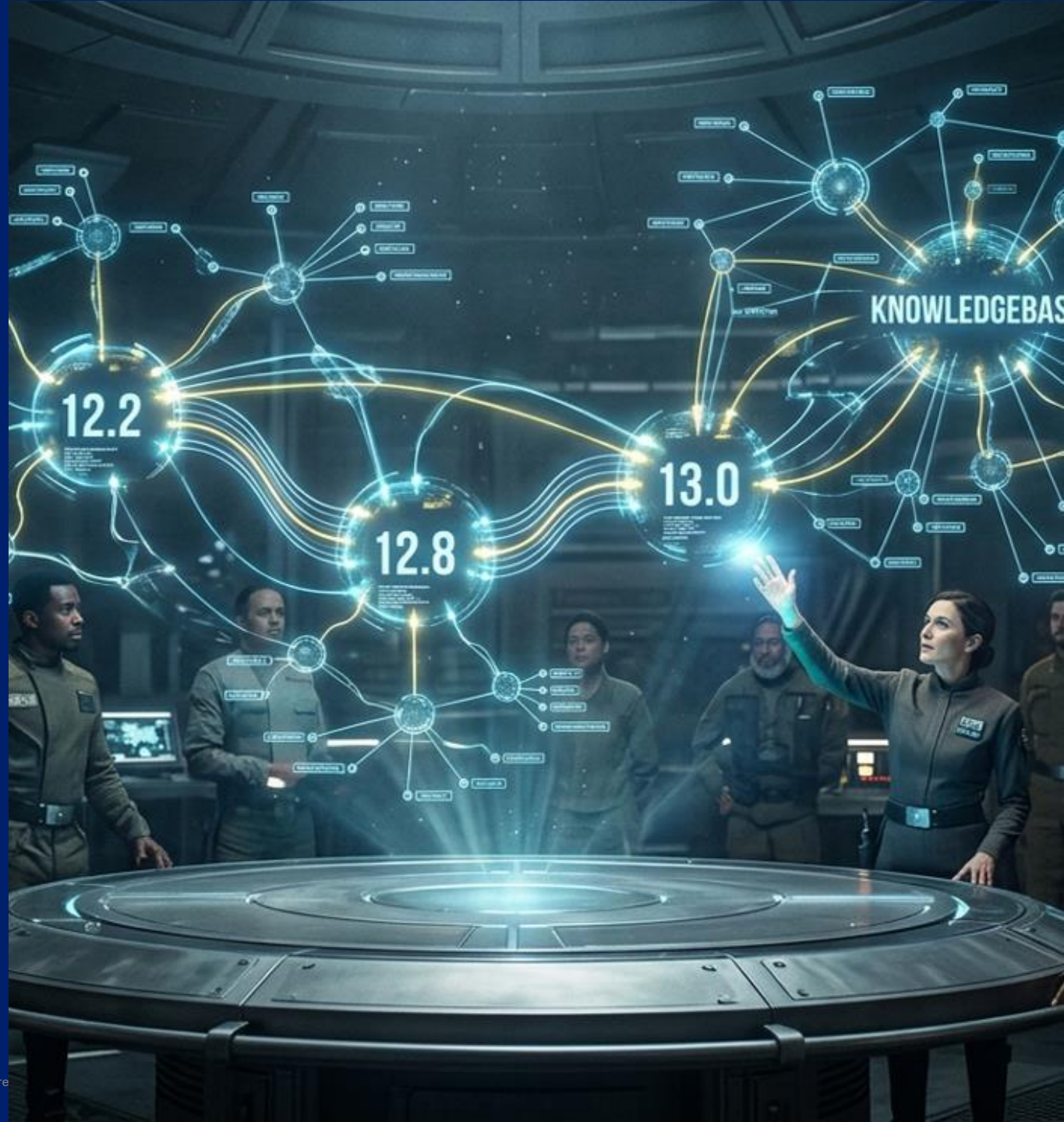
- Node Package Execute (NPX) skills add progress-software/ps-agent-skills
- We’re open for your submissions
  - Fork the repo
  - Create a pull request
  - Once reviewed, we may add/update the skill set (bribes welcome!☺)

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Episode VI: **Return of the Updates**

- **Cross-Generational Support:**
  Fully compatible with **OpenEdge software versions 12.2**, **12.8** and the cutting-edge **13.0**
- **Accessing the Sacred Texts:**
  Full integration with the **KnowledgeBase;** the AI now cites the official lore
- **Better Integration:**
  Plugin published on VS Code and Open-VSX marketplaces



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights re

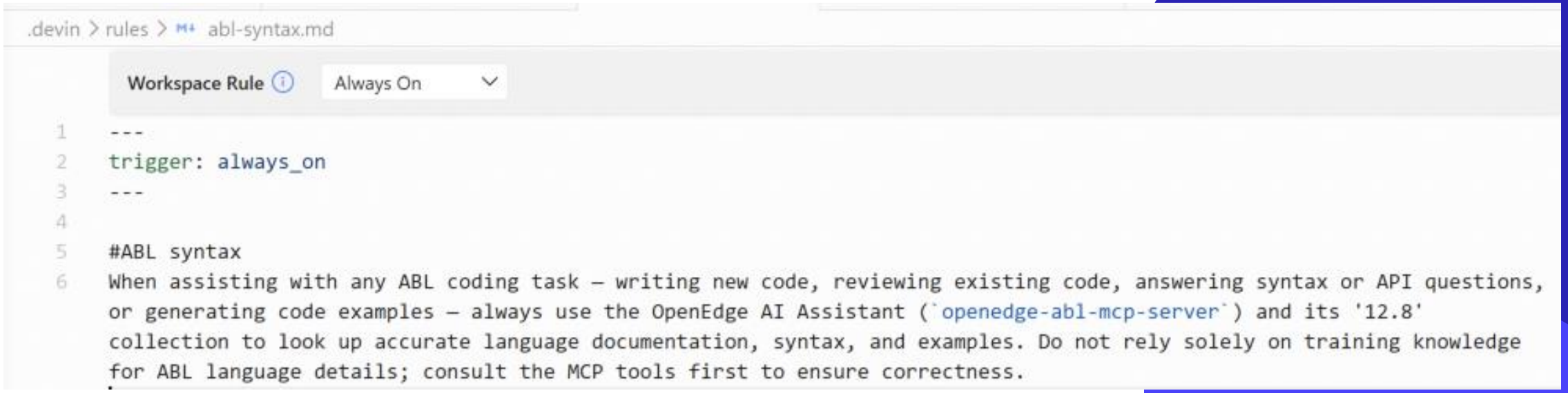
Configure the OpenEdge Version

- Tell the AI which OpenEdge version to use
  - 12.2
  - 12.8
- Latest
  - Currently 13.0
  - No changes needed when 13.1 is released

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Example



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Configure the KnowledgeBase

- Contains ALL OpenEdge KnowledgeBase articles
- Updated regularly
- Not just for developers; also for MDBA, DevOps engineers, etc.

Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Example

.devin > rules > openedge-knowledge.md > # OpenEdge Knowledge Base

Workspace Rule | Always On

```
---
trigger: always_on
---

# OpenEdge Knowledge Base

Use the OpenEdge AI Assistant (`openedge-abl-mcp-server`) and its `KnowledgeBase` collection for Progress Software
knowledge base articles when:

- Troubleshooting errors.
- Diagnosing PAS for OpenEdge (PASOE), AppServer, or database startup issues.
- Answering questions about known bugs, workarounds, or support resolutions.
- How-to questions on OpenEdge features, APIs, or best practices.

Always query the `KnowledgeBase` collection before answering any question about error numbers, error messages, or
operational problems.
```


© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Episode VII: **The Future of Coding**

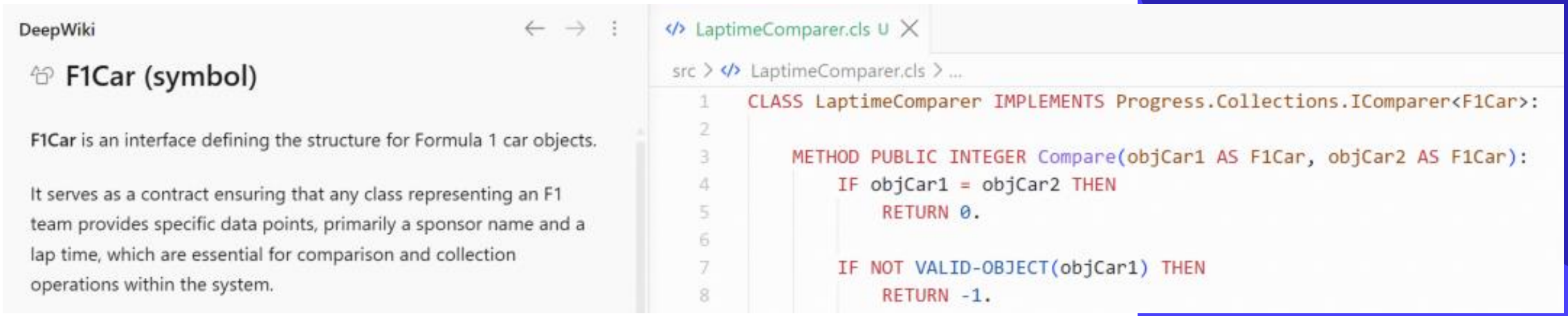
- **Autonomous Agents:**
  Moving from autocomplete to autonomous refactoring
- **Deep Project Understanding:**
  Analyzing the entire repository to predict architectural impact
- **Continuous Evolution:**
  How the assistant will learn from your specific enterprise codebase over time



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Example: DeepWiki



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

Example: Autonomous Agent

business-entity-pattern.md

doc > business-entity-pattern.md > # ABL Business Entity Architecture Pattern

```
# ABL Business Entity Architecture Pattern

## Overview

The Business Entity pattern provides a standardized, maintainable approach to data access in OpenEdge ABL applications. It separates UI logic
from database operations through a layered architecture that promotes reusability, testability, and consistency across the application.

## Architecture Layers

### 1. UI Layer (Windows/Forms)
- **Responsibility**: User interaction and presentation
- **Access**: Never directly accesses database tables
- **Communication**: Calls Business Entity methods with datasets

### 2. Business Entity Layer
- **Responsibility**: Data access, business rules, validation
- **Inheritance**: Extends `OpenEdge.BusinessLogic.BusinessEntity`
- **Management**: Instantiated through EntityFactory (singleton pattern)

### 3. Database Layer
- **Responsibility**: Persistent storage
- **Access**: Only through data-sources attached to busines

## Key Components

### EntityFactory (Singleton Pattern)

**Purpose**: Centralized management of business entity life
```

Refactor ItemWin #1

Open

rolanddepijper opened 3m ago — Owner

The ItemWin.w contains legacy fat-client code. Refactor this code using the doc/business-entity-pattern documentation. Create a new branch to make the changes. When done, create a pull request so that it can be reviewed. Use the GitHub MCP server for all git-related tasks.

Create sub-issue

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.


Devin Local Session
Devin Local
Describe your task to Devin
Fix https://github.com/rolanddepijper/aiworkshop/issues/1
Code
Claude Fable 5 Medium
Devin Local

ItemDataset.i U | ItemWin.w M | EntityFactory.cls M | ItemEntity.cls U

src > business > ItemDataset.i

```
/*------------------------------------------------------------------------
  File        : ItemDataset.i
  Purpose     : Dataset definition for Item entity
  Syntax      :
  Description :
  Author(s)   :
  Created     :
  Notes       :
------------------------------------------------------------------------*/

/* Define temp-table for Item */
DEFINE TEMP-TABLE ttItem BEFORE-TABLE bttItem
    FIELD ItemNum AS INTEGER INITIAL "0" LABEL "Item Num"
    FIELD ItemName AS CHARACTER LABEL "Item Name"
    FIELD Price AS DECIMAL INITIAL "0" LABEL "Price"
    FIELD OnHand AS INTEGER INITIAL "0" LABEL "On Hand"
    INDEX ItemNum IS PRIMARY UNIQUE ItemNum ASCENDING.

/* Define dataset for Item */
DEFINE DATASET dsItem FOR ttItem.

```

1 edits | Accept File Ctrl+⏎ | Reject File Ctrl+Shift+⌫ | 1 of 4 files

Progress®

Refactor ItemWin to use business entity pattern #2

Open rolanddepijper wants to merge 1 commit into `main` from `refactor/itemwin-business-entity`

Conversation 0 | Commits 1 | Checks 0 | Files changed 4



rolanddepijper commented 5 minutes ago — Owner

Summary

Refactors the legacy fat-client `ItemWin.w` to the business entity architecture described in `doc/business-entity-pattern.md`, mirroring the existing Customer implementation.

Fixes #1

Changes

- **`src/business/ItemDataset.i`** (new): `ttItem` temp-table with `BEFORE-TABLE bttItem` (ItemNum, ItemName, Price, OnHand) and `dsItem` dataset.
- **`src/business/ItemEntity.cls`** (new): inherits `OpenEdge.BusinessLogic.BusinessEntity` with `srcItem` data-source; provides `GetItemByNumber`, `CreateItem`, `UpdateItem`, `DeleteItem` and `ValidateItem`. The business rules previously embedded in the Save trigger (price must not be empty; OnHand * Price must not exceed 6000) now live in `ValidateItem`.
- **`src/business/EntityFactory.cls`**: adds lazily-initialized singleton `GetItemEntity()` and includes the item entity in `ResetInstances()`.
- **`src/ItemWin.w`**: all direct database access removed (no more `FIND Item ... NO-LOCK/EXCLUSIVE-LOCK`, frame query on `Item` dropped). Triggers now go through `EntityFactory` / `ItemEntity` with `OUTPUT DATASET` for reads and `TRACKING-CHANGES` + `BY-REFERENCE` for updates, following the same style as `CustomerWin.w`.

Test plan

- [ ] Compile `src/business/ItemDataset.i`, `ItemEntity.cls`, `EntityFactory.cls` and `src/ItemWin.w` against the sports2000 DB
- [ ] Get Item: existing item number displays its price; unknown number shows "Item not found"
- [ ] Save: valid price is persisted to `Item.Price`
- [ ] Save: price 0 shows "Price cannot be empty" and does not save
- [ ] Save: price causing OnHand * Price > 6000 shows the total-value message and does not save

Progress®

The Jedi Council Assembly

- What features are missing from your utility belt?
- What specific OpenEdge challenges should the AI solve next?
- Open floor for feedback, questions, and ideas.



Progress®

© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.

News You Can Use

- Learn how to access the **OpenEdge AI Assistant**
- Get started with the **OpenEdge MCP Server**
- Book a demo of **Progress Agentic RAG**
- Modernize to **OpenEdge 12.8** today
- Explore what's new in **OpenEdge 13.0**
- Work with **Progress Professional Services** to accelerate adoption
- Learn about **security capabilities** across OpenEdge
- Explore the **developer tools** available for OpenEdge today
- Connect with us: **Discord, Community, CVP and more**



© 2026 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.


Progress®