Using IBM AI Generative Tooling to Assist with Code Development

Ben Hicks – ben.hicks@ibm.com
Mary Julian – Mary.Julian@ibm.com
February 26, 2026

1

Disclaimer

Anything presented here is subject to change upon release.

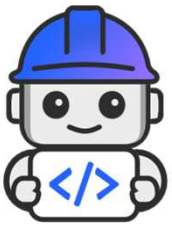
We are not developers of IBM Bob.

2

Agenda

- Overview of IBM Bob
- Modes for Bob
- Providing Context
- Understanding the Code Base
- MCP Servers

3

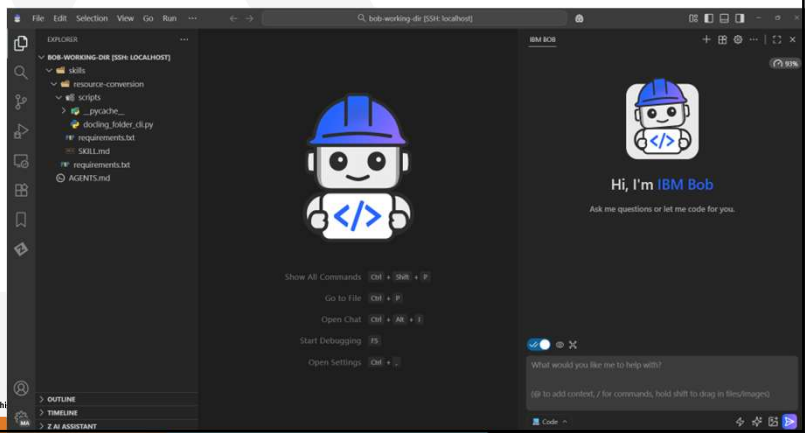


Overview of IBM Bob

4

What is IBM Bob?

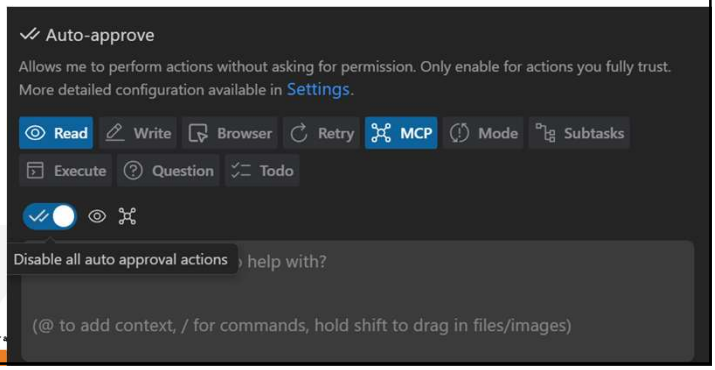
- It an AI-powered IDE and assistant.
  - Bob is a fork of Visual Studio Code, with support for most extensions.
- Security focused, supports on-premises and IBM Z environments.
- Focused on development work, but can also help with managing z/OS systems.



5

Why Bob?

- Security-first architecture.
- Structured behavior to prevent hidden decisions.
  - Developer stays in control.
- Native knowledge of Z and mainframes.
- Built-in compliance, auditable outcomes and predicable governance.



6

SHARE
EDUCATE • NETWORK • INFLUENCE

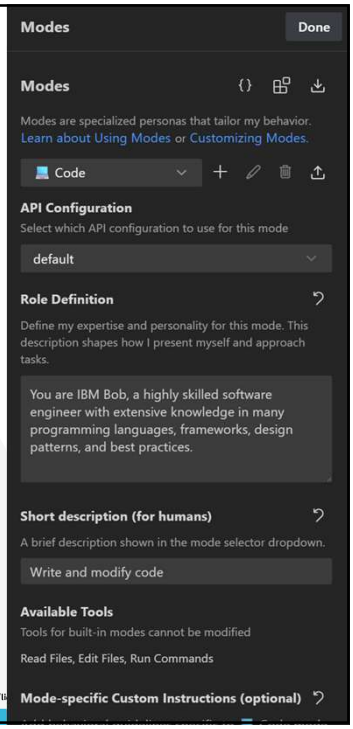
Modes for Bob

7

Modes

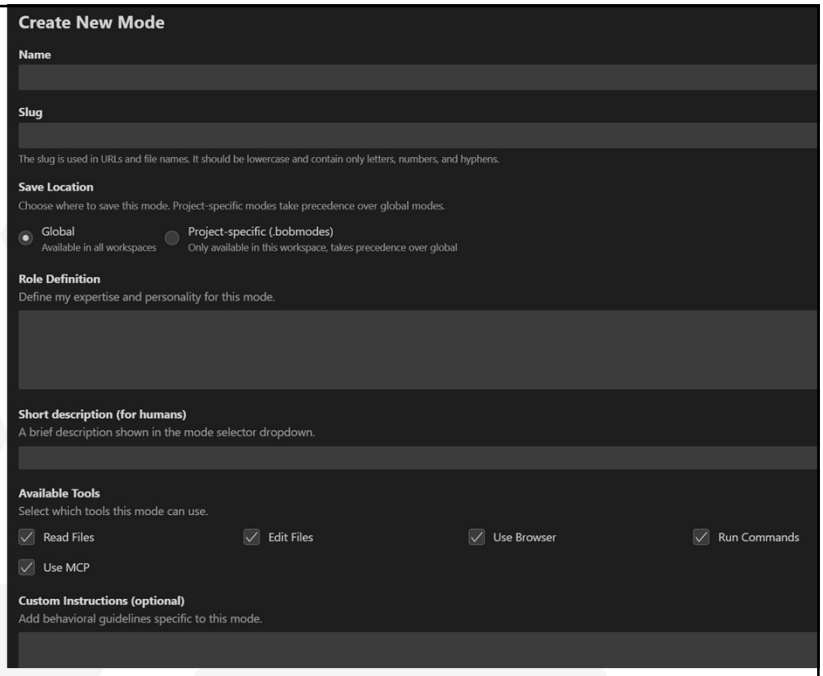
- Code - Write and modify code.
- Plan - Plan tasks, analyze requirements, research and design implementations.
- Ask - Ask questions and get answers.
- Advanced - Full-featured development, including using MCP servers.

- Customization allowed for each mode.
- Custom modes can be created.



Custom Modes

- A mode contains:
  - Name
  - Role definition
  - Available tools
  - Custom instructions
- There are also global instructions for every mode.
- Custom modes can be used for specific tasks.



9

9

SHARE
EDUCATE • NETWORK • INFLUENCE

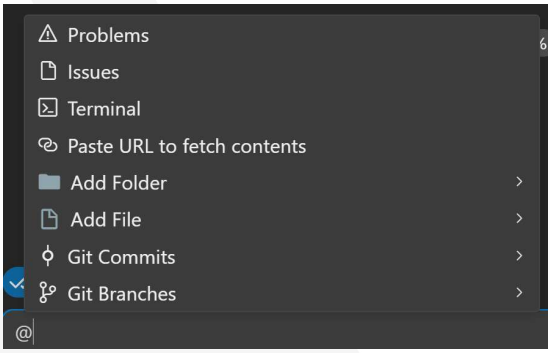
Providing Context

10

10

Providing Context

- Bob is only as smart as the information you give it.
  - Good context is important, but not too much to overload the context window (200,000 tokens).
  - "What information does someone new to the project need to solve this problem?"
- Potential context:
  - File/folder
  - Git commit/branch
  - URL
  - Terminal output
  - Rules files



11

11

Using Rules

- Rules files are important to customize Bob to fit your needs.
  - Codebase-specific or global rules.
- Global rules folder: ~/.bob/rules
- Workspace folder: ./.bob/rules
- Use .md files to provide additional context to Bob.
  - Code layout, design
  - Coding standards to follow for specific languages
  - General response guidelines
- **Important:** Adding more rules increases the token usage on each request.

12

12

Sample Rule File

```
You are an expert in COBOL programming.

## When to Apply
Apply these rules when:
- Working with COBOL source files (*.cbl, *.cob, *.cobol)
- Analyzing or modifying mainframe business applications
- Dealing with CICS, IMS, DB2, or batch COBOL programs
- User explicitly mentions COBOL programming

## Key Principles
Always prioritize code clarity and maintainability over cleverness. COBOL
programs often run for decades, so write code that future developers (human
or AI) can easily understand and modify.
...
```


13

Understanding the Code Base

14

SHARE
EDUCATE • NETWORK • INFLUENCE

I don’t know COBOL

15

I don’t know COBOL

SHARE
EDUCATE • NETWORK • INFLUENCE

- I do know how to create efficient code
- I do know how to check for edge cases
- I know what good object-oriented code looks like
- I know Python

- Between Bob and me, we can understand a legacy codebase, debug issues, and transition to a modern coding language.

16

Explain My Code – There's a bug

```
IDENTIFICATION DIVISION.
    PROGRAM-ID. SIMPLEMULT.

    ENVIRONMENT DIVISION.

    DATA DIVISION.
    WORKING-STORAGE SECTION.

   * DIMENSIONS OF ARRAYS - A IS M BY N, B IS N BY P, C IS M BY P
    1   M PIC 9(3) COMP VALUE 10.
    1   N PIC 9(3) COMP VALUE 10.
    1   P PIC 9(3) COMP VALUE 10.

   * ARRAY A - 10 x 10
    1   A1.
     2   A2 OCCURS 10 TIMES.
      3   ARRAY-A PIC S9(4) OCCURS 10 TIMES.

   * ARRAY B - 10 x 10
    1   B1.
     2   B2 OCCURS 10 TIMES.
      3   ARRAY-B PIC S9(4) OCCURS 10 TIMES.

   * ARRAY C - 10 x 10 (RESULT)
    1   C1.
     2   C2 OCCURS 10 TIMES.
      3   ARRAY-C PIC S9(8) OCCURS 10 TIMES.

   * LOOP COUNTERS
    1   I PIC 9(3) COMP.
    1   J PIC 9(3) COMP.
    1   K PIC 9(3) COMP.
    1   TEMP PIC S9(3) COMP.

   PROCEDURE DIVISION.
   MAIN-LOGIC.
       DISPLAY "MATRIX MULTIPLICATION PROGRAM START"
       DISPLAY " "

   * INITIALIZE ARRAY A - EACH ELEMENT = I + J
       PERFORM VARYING I FROM 1 BY 1 UNTIL I > M
         PERFORM VARYING J FROM 1 BY 1 UNTIL J > N
           COMPUTE ARRAY-A(I, J) = I + J
         END-PERFORM
       END-PERFORM
       DISPLAY "ARRAY A INITIALIZED"

   * INITIALIZE ARRAY B - EACH ELEMENT = I + J
       PERFORM VARYING I FROM 1 BY 1 UNTIL I > N
         PERFORM VARYING J FROM 1 BY 1 UNTIL J > P
           COMPUTE ARRAY-B(I, J) = I + J
         END-PERFORM
       END-PERFORM
       DISPLAY "ARRAY B INITIALIZED"
```


17

Explain My Code

```
   * MULTIPLY ARRAY A TIMES ARRAY B GIVING ARRAY C
   * C[I,J] = SUM OF A[I,K] * B[K,J] FOR ALL K
       PERFORM VARYING I FROM 1 BY 1 UNTIL I > M
         PERFORM VARYING J FROM 1 BY 1 UNTIL J > P
           MOVE 0 TO ARRAY-C(I, J)
           PERFORM VARYING K FROM 1 BY 1 UNTIL K > N
             COMPUTE TEMP = ARRAY-A(I, K) * ARRAY-B(K, J)
             COMPUTE ARRAY-C(I, J) = ARRAY-C(I, J) + TEMP
           END-PERFORM
         END-PERFORM
       END-PERFORM
       DISPLAY "MATRIX MULTIPLICATION COMPLETE"

   * DISPLAY SAMPLE RESULTS (FIRST ROW OF RESULT MATRIX)
       DISPLAY " "
       DISPLAY "SAMPLE RESULTS - FIRST ROW OF MATRIX C:"
       PERFORM VARYING J FROM 1 BY 1 UNTIL J > P
         DISPLAY "C(1," J ") = " ARRAY-C(1, J)
       END-PERFORM

       DISPLAY " "
       DISPLAY "PROGRAM END"
       STOP RUN.
```

- Where do I start?
- My manager wants this fixed now

18

Explain My Code

Explanation of array_mult_simple for New Grads

This is a **COBOL program** that performs **matrix multiplication**. Here's what it does in simple terms:

What It Does
Multiplies two 10x10 matrices (Array A × Array B = Array C)

Program Structure

1. **Setup (Lines 6-38)**
- **IDENTIFICATION DIVISION**: Names the program "SIMPLEMULT"
- **DATA DIVISION**: Declares all variables and arrays
  - `M`, `N`, `P`: Matrix dimensions (all set to 10)
  - `ARRAY-A`: First 10×10 matrix (holds numbers up to 4 digits)
  - `ARRAY-B`: Second 10×10 matrix (holds numbers up to 4 digits)
  - `ARRAY-C`: Result 10×10 matrix (holds numbers up to 8 digits - larger because multiplication creates bigger numbers)
  - `I`, `J`, `K`: Loop counters (like `i`, `j`, `k` in other languages)
  - `TEMP`: Temporary storage for calculations

2. **Initialize Array A (Lines 46-51)**
```
For each row I from 1 to 10:
  For each column J from 1 to 10:
    A[I,J] = I + J
```
Example: A[1,1]=2, A[1,2]=3, A[2,1]=3, etc.

3. **Initialize Array B (Lines 54-59)**
Same pattern as Array A - each element equals its row + column

19

Explain My Code

4. **Matrix Multiplication (Lines 63-71)** - The Core Algorithm
This implements the standard matrix multiplication formula:
```
For each row I in result matrix:
  For each column J in result matrix:
    C[I,J] = 0  (start with zero)
    For each K from 1 to 10:
      C[I,J] += A[I,K] × B[K,J]
```

Key Concept: To get element C[I,J], you multiply row I of matrix A by column J of matrix B, element by element, then sum them up.

5. **Display Results (Lines 75-79)**
Shows the first row of the result matrix as a sample

COBOL-Specific Notes
- **PERFORM VARYING**: COBOL's version of a for-loop
- **COMPUTE**: Does arithmetic operations
- **PIC S9(4)**: Defines a signed 4-digit number
- **COMP**: Stores numbers in binary (more efficient)
- **OCCURS**: Creates an array

Why "Simple"?
This version uses fixed 10×10 arrays and straightforward logic, making it easier to understand than more complex implementations with variable sizes or optimizations.

20

I still have questions

- Why are the loop counters PIC 9(3)? What does that mean?

```
1   I PIC 9(3) COMP.
1   J PIC 9(3) COMP.
1   K PIC 9(3) COMP.
```

What Each Part Means
PIC 9(3) = "Picture 9, 3 digits"
- Can hold values from 0 to 999
- The 9 means numeric digit
- The (3) means 3 digits total

COMP = "Computational" (binary format)
- Stores the number in binary instead of text
- More efficient for arithmetic operations
- Faster for loop counters

Why 3 Digits for This Program?
In this specific program:
- The arrays are only 10×10 (M=10, N=10, P=10)
- Loop counters only go from 1 to 10
- So technically, PIC 9(2) (0-99) would be enough

But PIC 9(3) is used because:
- Future-proofing: If someone wants to expand the arrays to 100×100 or larger, the counters can handle it
- Standard practice: 3 digits (up to 999) is a common choice for array indices
- Minimal overhead: The extra digit doesn't significantly impact performance or memory

21

I still have questions

- How about s9(8)? What does that mean?

```
1   C1.
 2   C2 OCCURS 10 TIMES.
  3   ARRAY-C PIC S9(8) OCCURS 10 TIMES.
1   TEMP PIC S9(8) COMP.
```

Breaking Down S9(8)
S = Signed (can be positive or negative)
- Range: -99,999,999 to +99,999,999

9(8) = 8 digits total

Why 8 Digits? Let's Do the Math!
Input Arrays (S9(4))
- ARRAY-A: PIC S9(4) → range -9,999 to +9,999
- ARRAY-B: PIC S9(4) → range -9,999 to +9,999

Matrix Multiplication Formula
C[I,J] = A[I,1]×B[1,J] + A[I,2]×B[2,J] + ... + A[I,10]×B[10,J]

Worst Case Calculation
1. Single multiplication: 9,999 × 9,999 = 99,980,001 (8 digits!)
2. Sum of 10 multiplications: 99,980,001 × 10 = 999,800,010 (9 digits!)

22

There's a bug to fix!

SHARE
EDUCATE • NETWORK • INFLUENCE

- Why does the math seem wrong? The answer is too small?

The Issue: Data Type Overflow
Line 38: TEMP PIC S9(3) COMP - This is too small!
The problem is that TEMP is defined as a signed 3-digit number (max value 999), but when multiplying two elements from the arrays, the result can exceed this:

Example calculation:
- Array A and B elements can be up to 20 (when i=10, j=10: 10+10=20)
- Multiplying: 20 × 20 = 400 ✓ (fits in S9(3))
- BUT the sum in ARRAY-C accumulates 10 such products
- Total: 10 × 400 = 4,000 (does NOT fit in S9(3))

The Fix
Change line 38 from:

```
1   TEMP PIC S9(3) COMP.
```

To:

```
1   TEMP PIC S9(8) COMP.
```

This matches the size of ARRAY-C elements (S9(8)) and prevents overflow during intermediate calculations.

Why This Causes Small Results
When TEMP overflows during multiplication, it wraps around or truncates, producing incorrect small values that get added to ARRAY-C, resulting in answers that are much smaller than expected.

The intermediate multiplication result needs to be large enough to hold the product before adding it to the accumulator.

23

Python is Easier!

SHARE
EDUCATE • NETWORK • INFLUENCE

- Can you make this in Python? `# Made with Bob`

```python
def main():
    print("MATRIX MULTIPLICATION PROGRAM START")
    print()

# DIMENSIONS OF ARRAYS - A IS M BY N, B IS N BY P, C IS M
BY P
    M = 10
    N = 10
    P = 10

    # INITIALIZE ARRAY A - 10 x 10
    # Each element = i + j
    array_a = [[0 for _ in range(N)] for _ in range(M)]

    for i in range(M):
        for j in range(N):
            array_a[i][j] = (i + 1) + (j + 1)  # +1
because COBOL uses 1-based indexing
    # INITIALIZE ARRAY B - 10 x 10
    # Each element = i + j
    array_b = [[0 for _ in range(P)] for _ in range(N)]

    for i in range(N):
        for j in range(P):
            array_b[i][j] = (i + 1) + (j + 1)
```

```python
    # MULTIPLY ARRAY A TIMES ARRAY B GIVING ARRAY C
    # C[I,J] = SUM OF A[I,K] * B[K,J] FOR ALL K
    array_c = [[0 for _ in range(P)] for _ in range(M)]

    for i in range(M):
        for j in range(P):
            array_c[i][j] = 0
            for k in range(N):
                temp = array_a[i][k] * array_b[k][j]
                array_c[i][j] = array_c[i][j] + temp

    print("MATRIX MULTIPLICATION COMPLETE")

    # DISPLAY SAMPLE RESULTS (FIRST ROW OF RESULT MATRIX)
    print()
    print("SAMPLE RESULTS - FIRST ROW OF MATRIX C:")
    for j in range(P):
        print(f"C(1,{j+1}) = {array_c[0][j]}")  # 0-based
index, but display as 1-based

    print()
    print("PROGRAM END")


if __name__ == "__main__":
    main()
```


24

What's Next?

- Can you add unit tests?
- Can you create a README?
- Can you create Makefiles?
- Can you suggest performance improvements?
- Can you scale the solution?

Yes!

25

MCP Servers

26

MCP Servers Background

SHARE
EDUCATE • NETWORK • INFLUENCE

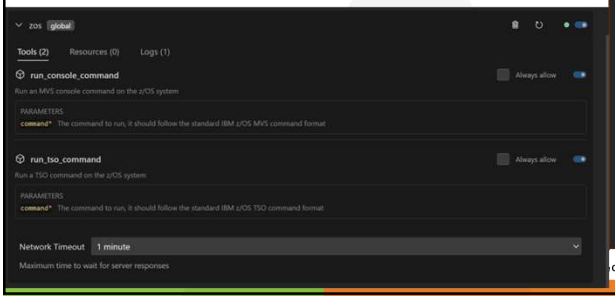
- Allow an AI agent to communicate with a system or server.
  - Uses the Model Context Protocol (MCP): https://modelcontextprotocol.io/.
- The server uses JSON input/output, either through standard IO (local server), or through HTTP POST requests (remote server).
- Extends the functionality of the AI agent, can be used to perform actions on a z/OS system.
- Examples: https://github.com/ibm/mcp
  - MQ Server – view health checks, run MQSC commands.
  - IBM z/OS Connect MCP Server – enables existing z/OS Connect APIs to be used as MCP tools.

27

Sample MCP Server

SHARE
EDUCATE • NETWORK • INFLUENCE

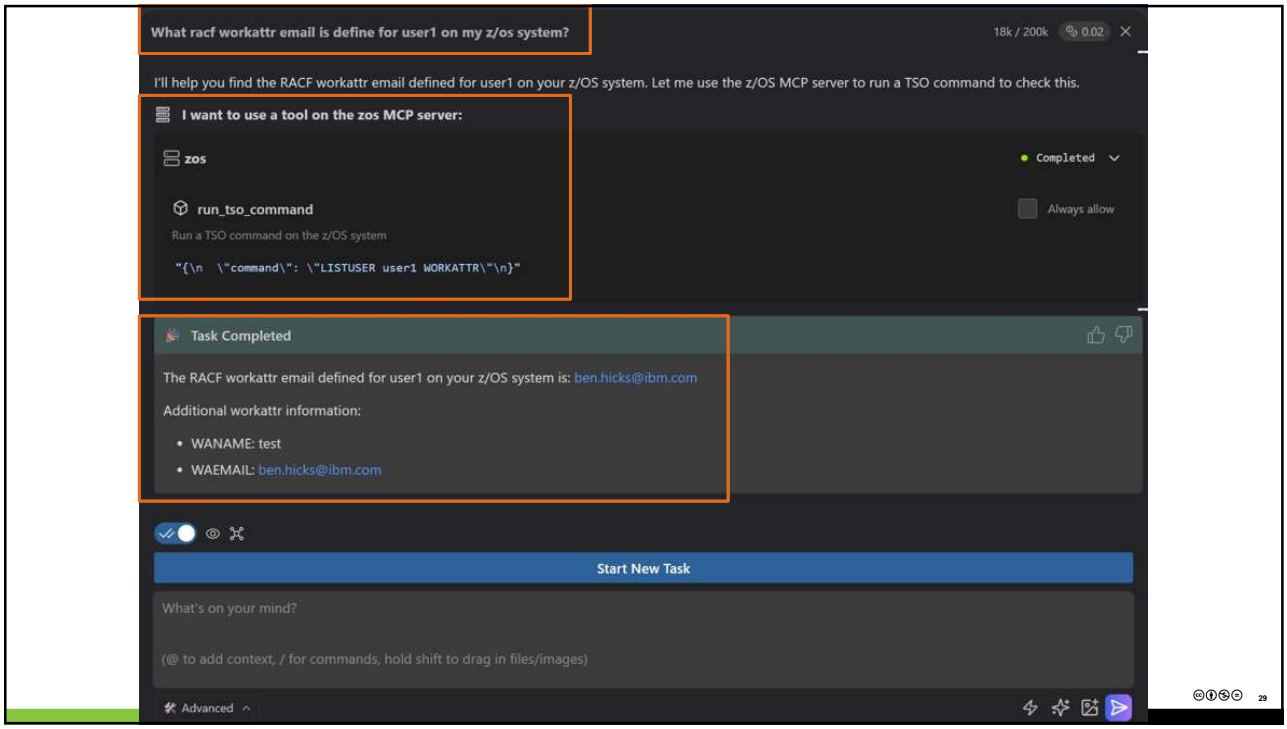
- z/OS currently does not support the Python package FastMCP due to a dependency on Ruby.
- This is a Node.js server that runs natively on z/OS.
- Enables an AI agent to run MVS console and TSO commands.



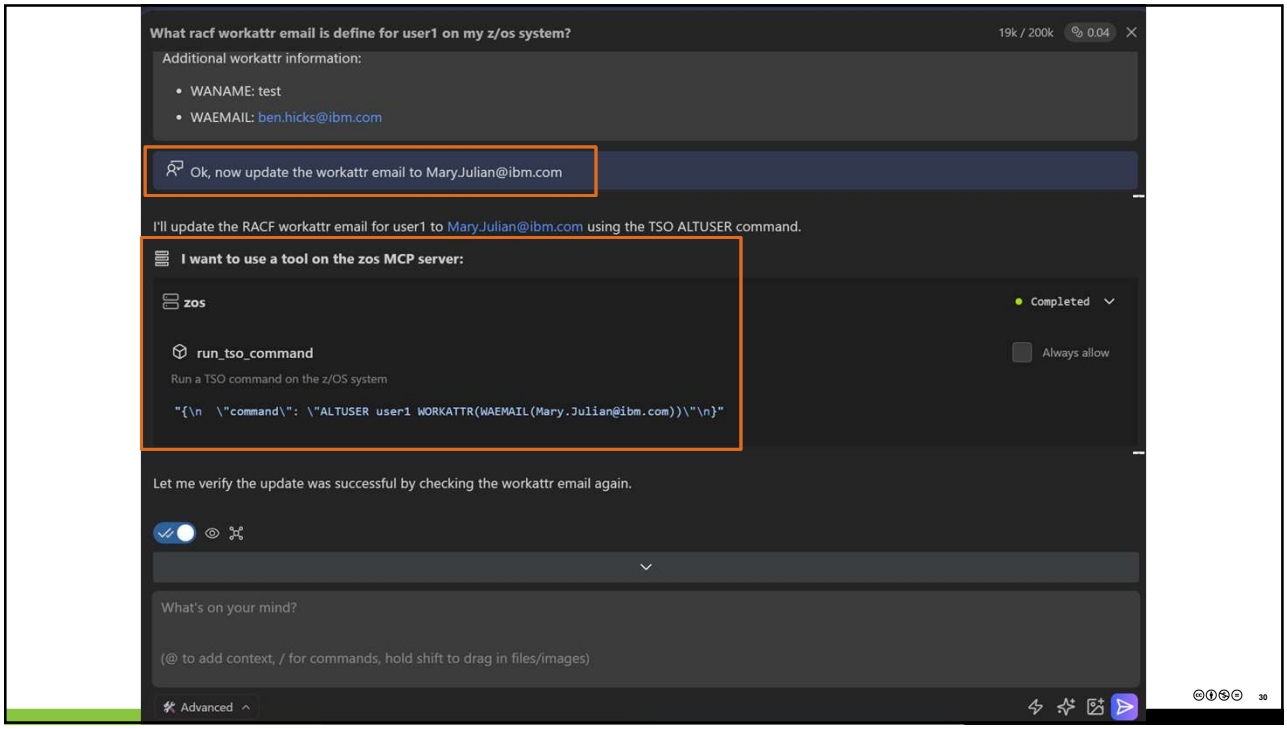

```
server.registerTool(
  "run_tso_command",
  {
    description: "Run a TSO command on the z/OS system",
    inputSchema: z.object({
      command: z.string().describe("The command to run, it
should follow the standard IBM z/OS TSO command format. It
should always be a valid TSO command."),
    }),
  },
  async ({ command }) => {
    try {
      const { stdout, stderr } = await execFileAsync("tsocmd",[
        command,
      ]);

      return {
        content: [
          {
            type: "text",
            text: stdout,
          },
```

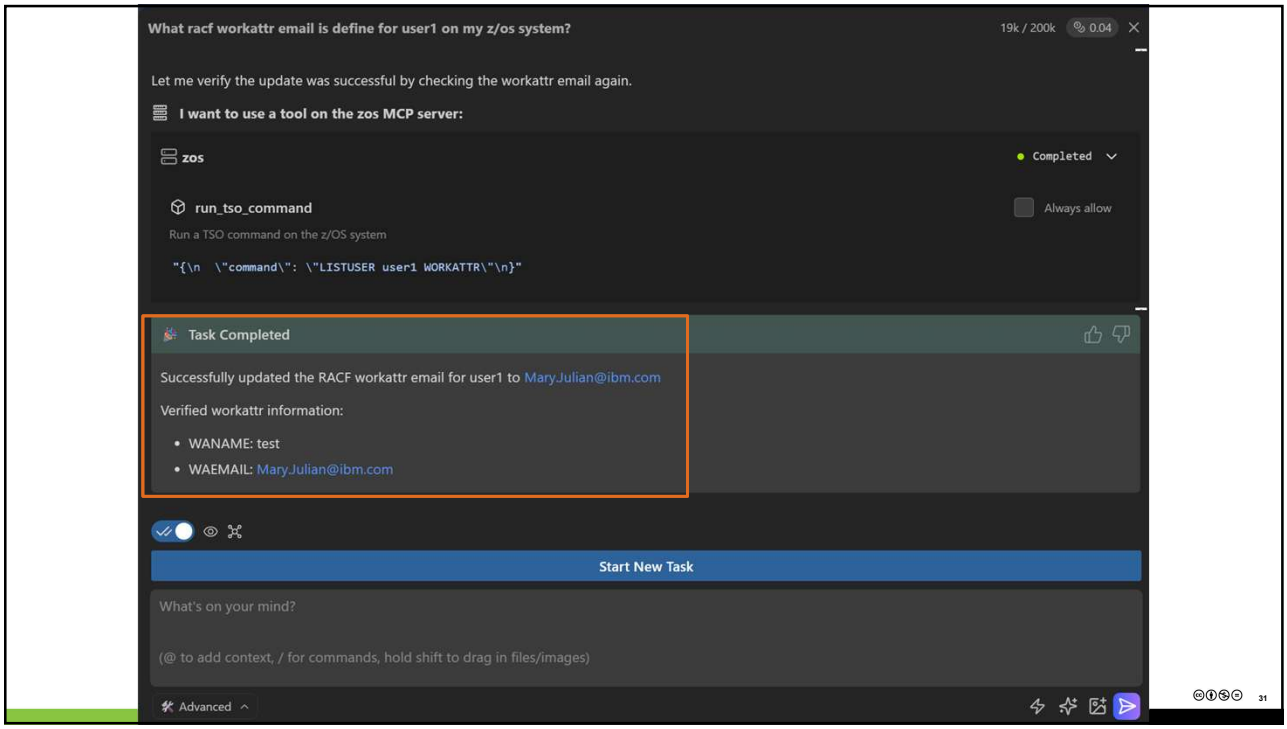

28


What racf workattr email is define for user1 on my z/os system?
18k / 200k
0.02
I'll help you find the RACF workattr email defined for user1 on your z/OS system. Let me use the z/OS MCP server to run a TSO command to check this.
I want to use a tool on the zos MCP server:
zos
Completed
run_tso_command
Always allow
Run a TSO command on the z/OS system
"{\n \"command\": \"LISTUSER user1 WORKATTR\"\n}"
Task Completed
The RACF workattr email defined for user1 on your z/OS system is: ben.hicks@ibm.com
Additional workattr information:
WANAME: test
WAEMAIL: ben.hicks@ibm.com
Start New Task
What's on your mind?
(@ to add context, / for commands, hold shift to drag in files/images)
Advanced
29

29


What racf workattr email is define for user1 on my z/os system?
19k / 200k
0.04
Additional workattr information:
WANAME: test
WAEMAIL: ben.hicks@ibm.com
Ok, now update the workattr email to MaryJulian@ibm.com
I'll update the RACF workattr email for user1 to MaryJulian@ibm.com using the TSO ALTUSER command.
I want to use a tool on the zos MCP server:
zos
Completed
run_tso_command
Always allow
Run a TSO command on the z/OS system
"{\n \"command\": \"ALTUSER user1 WORKATTR(WAEMAIL(Mary.Julian@ibm.com))\"\n}"
Let me verify the update was successful by checking the workattr email again.
What's on your mind?
(@ to add context, / for commands, hold shift to drag in files/images)
Advanced
30

30



31

SHARE
EDUCATE • NETWORK • INFLUENCE

Questions?

32

Other IBM Z AI Offerings

- Watsonx Assistant for Z
  - https://www.ibm.com/products/watsonx-assistant-for-z
- Watsonx Code Assistant for Z
  - https://www.ibm.com/products/watsonx-code-assistant-z
- AI-infused z/OS
  - https://www.ibm.com/docs/en/zos/3.2.0?topic=zos-ai-providers-use-cases
  - AI-powered WLM batch initiator management
  - AI-powered network outbound packet batching

33

Experience more with IBM

Visit us at the IBM Booth #113

After a full day of technical sessions, take a break with us!

Connect with our experts, snap a photo with the z17 Plexi or the latest Telum II, and get an up-close look at our Spyre Accelerator.

Come back each day for fresh topics and demos at our expert stations.

Think 2026

Join 5000+ senior business and technology leaders who are seizing the AI revolution to unlock unprecedented growth and productivity at Think 2026.

Find out more information using the QR code below.

IBM Digital Asset Haven

IBM Digital Asset Haven is the operational backbone for financial institutions and regulated enterprises entering the digital asset economy.

Find out more information using the QR code below.

34

Your feedback is important!

SHARE
EDUCATE • NETWORK • INFLUENCE

Submit a session evaluation for each session you attend:

www.share.org/evaluation

35

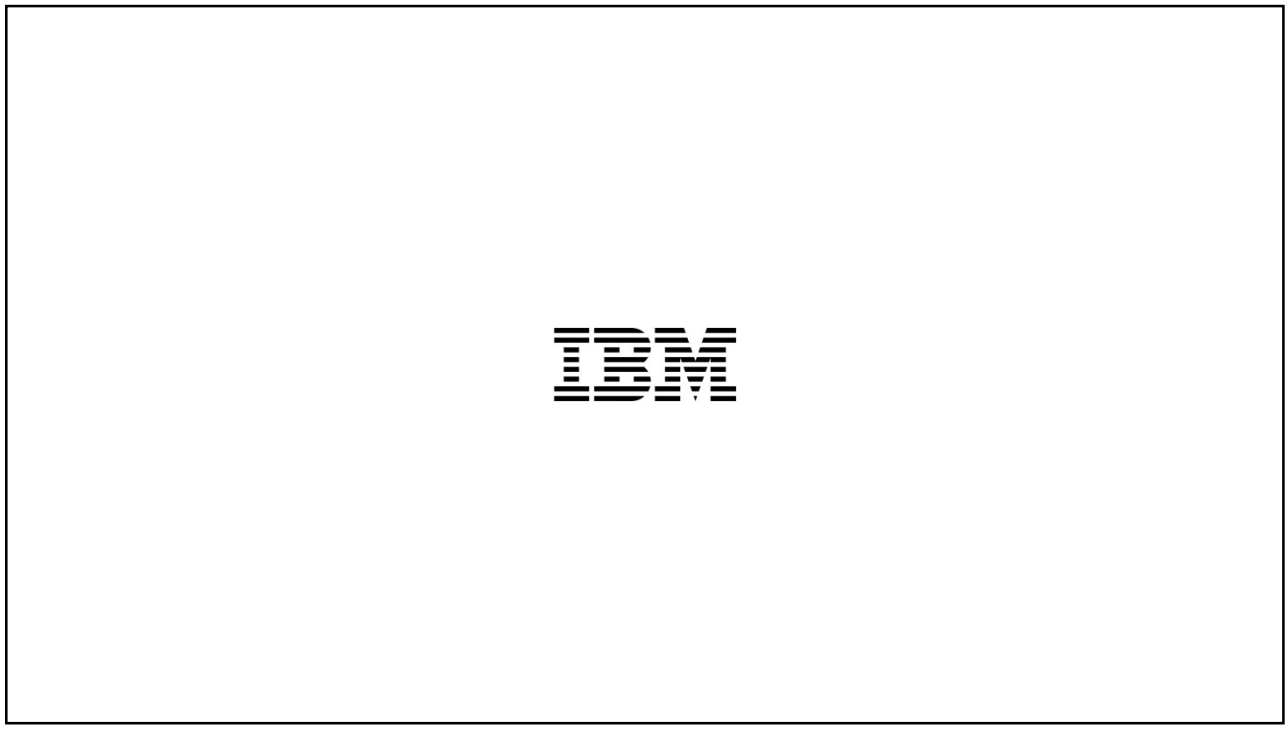
SHARE
EDUCATE • NETWORK • INFLUENCE

Thank You

36

Notices and disclaimers

Notices and disclaimers



39