NEW Meeting summary with AI Companion now supports additional languages in preview. Learn More

Meeting summary for 4DMethod Meeting (04/02/2025)

Quick recap

The meeting focused on integrating Visual Studio Code with 4D development, showcasing features like code highlighting, debugging, and Git operations to improve coding efficiency. Participants explored the potential of AI integration in 4D, including a new vectorization feature for database tables and the use of GitHub Copilot for code generation and documentation. The discussion concluded with ideas for future AI-powered enhancements in 4D development, emphasizing the rapid advancements in AI technology and the need to prioritize features that benefit end-users.

Next steps

• 4D Team: Create and share a GitHub repository for VS Code Copilot instruction files, prompts, and configuration files for 4D development by the end of the week
• 4D Team: Continue development of database vectorization feature for semantic search capabilities
• 4D Team: Investigate the possibility of adding VS Code debugger support for 4D standalone development
• Ricardo: Share the process of downloading and using 4D documentation specifications for AI training
• Mathieu/4D Team: Consider including VS Code integration and AI features presentations at the upcoming 4D Summit in Paris
• Community Members: Test the VS Code debugger with 4D Server R8 and provide feedback
• Community Members: Contribute to the upcoming VS Code Copilot instruction repository by adding custom prompts and training files
• 4D Team: Document the process of setting up custom color themes in VS Code for 4D development
• Community Members: Submit feature requests through the 4D forum regarding VS Code integration preferences and AI needs
• 4D Team: Consider implementing vector field as a special type object in future development
• Mohamed: Keep the demo machine up for 15 minutes after the meeting for attendees to try out the vector search functionality at the provided URL
• Mathieu: Monitor forum and feature requests for community feedback on VS Code

• Mathieu: Monitor forum and feature requests for community feedback on VS Code integration and AI features

---

Summary

Visual Studio Code for 40 Developers

In the meeting, Brent welcomed everyone to the 40 method user group meeting and introduced the special event of the day, which was a discussion on Visual Studio Code for 40 developers, code Debug, and copilot. He also mentioned the recent features and integrations between Visual Studio Code and 40. Brent then introduced Mathieu Ferry and Damien Fuzeau, who were joining from France and work at 40. They were there to show off how far 40 development has come using Visual Studio Code as an extra code editing tool. The conversation ended with Mathieu explaining the origins of their work with Visual Studio Code at 40 and the basics of using the tool.

4D Analyzer Extension Demo in VS Code

Damien and Mathieu demonstrate how 4D and Visual Studio Code can work together. They showcase the 4D Analyzer extension, which provides code highlighting, auto-suggestions, and error detection in VS Code. They also explore features like documentation viewing, code formatting, and search and replace across multiple files. The demonstration highlights how these tools can improve coding efficiency and help quickly fix errors in 4D applications.

Vs Code Project Structure Discussion

In the meeting, Mathieu discussed the basics of a project, including the use of Visual Studio Code (VS Code) and the need for a poll to gauge its usage. Tim and Damien provided insights on customizing VS Code's color schemes and themes. Brent raised a question about the possibility of making structure changes in VS Code, to which Mathieu responded that there is no visual structure editor for the N. VS Code at present. The team also discussed the challenges of using the home folder in the 4D explorer and the potential for using external editors. A suggestion was made to create an extension for labeling files in VS Code. The conversation ended with a discussion on the use of the actual file structure for project files in 4D, and the possibility of adding or moving methods to specific folders.

Visual Studio Code for Git

In the meeting, Mathieu demonstrated how Visual Studio Code (VS Code) can be a valuable partner for Git operations. He showcased how to view and commit changes, revert modifications, and use the timeline to track file changes. Tim asked about the necessity of restarting the interpreter when changes are made in VS Code, to which Damien clarified that the interpreter does not need to be restarted. Instead, the 'Reload Project' command can be used to update the changes.

Git Interface and VS Code Integration

In the meeting, the team discussed the integration of Git interface and VS Code, which provides better styling for code viewing. They also discussed the ability to hide warnings in VS Code, which is not possible in 4D. The team also discussed the compatibility of their project with different versions of 4D and the need to use the same version for compatibility. Lastly, they introduced a new debugging feature available since R.8.

Debugging 4D Code With VS Code
In the meeting, Mathieu demonstrated how to debug 4D code on 4D Server using VS Code's debugger. He showed how to set breakpoints, navigate through variables, and execute lines of code. The team also discussed the limitations of the debugger, such as the need to detach the local debugger and the fact that only one debugger can be used on 4D Server. They also discussed the possibility of using the debugger in 4D Server with a large project, which Mathieu acknowledged as a potential issue. The team also discussed the possibility of using the debugger in 4D Standalone, which was not yet available.

Vs Code and Github Copilot Integration
In the meeting, Brent and Mathieu discussed the integration of Vs code with 40 development, noting its potential to lower the barrier to entry for new developers. They also touched on the feature of attaching the debugger on 40 server to a local 4D copy, which was introduced in 20R8. Tim added that this feature allows for debugging 40 server stored procedures directly on a local 4D copy, eliminating the need for remote access. The team also explored the use of Github Copilot for code development, with Mathieu demonstrating how it can assist with code generation, documentation, and even translation. The conversation ended with a positive reception of the Copilot's capabilities, with one participant expressing excitement for its future developments.

Exploring Copilot's Documentation and Code Features
The group discusses the capabilities and potential uses of Copilot, particularly for documentation creation and code modernization. Mathieu and Damien demonstrate some features, including generating documentation and creating flowcharts. The team agrees to create a shared repository for Copilot instruction files, allowing the community to contribute and improve them. Ricardo mentions that he has successfully used 4D documentation to train AI models, which Tim finds interesting. The discussion concludes with a question about whether a paid Copilot subscription is required for these features.

Vectorizing Database Tables With AI
Mathieu and Mohamed present a new feature in development for 4D that involves vectorizing database tables. The feature will allow users to create vectors for each record in a table based on the content of its fields, using an AI model. These vectors can then be used to answer user prompts or semantic queries about the data. Mohamed demonstrates a prototype of this feature using an employee database, showing how a chatbot can retrieve detailed information about employees based on natural language queries. The team emphasizes that this feature is still in progress and not representative of the final product, but aims to give users an idea of its potential applications.

4D Vector Search System Demo
Mohamed demonstrated the use of a vector search system in a 4D database, which does not require training and can provide real-time results. The system uses an LLM to generate vectors representing the meaning of data, allowing for similarity-based searches. The system was shown to be able to answer questions about employee profiles and skills, and even suggest potential colleagues based on their age. The system was also demonstrated to be able to handle fuzzy searches. The team discussed the potential of this system for various applications, including image searches.

AI Integration in 4D Development
The group discusses the exciting potential of AI integration in 4D development, particularly focusing on the new vectorization feature demonstrated earlier. Mathieu explains that 4D is currently prioritizing features that benefit end-users, such as vectorizing the database, rather than AI-assisted coding for developers. The participants explore ideas for AI-powered documentation generation and code improvement, with Tim suggesting that 4D could train a language model specifically for 4D development. Mathieu acknowledges that they are considering these possibilities but emphasizes the rapid pace of AI advancements and the need to focus on immediate user benefits.

AI-generated content may be inaccurate or misleading. Always check for accuracy.

Edit

Please rate the accuracy of this summary.

Best,

Zoom