WHITEPAPER

TOP 12 ABILITIES FOR AI-NATIVE SOFTWARE DEVELOPERS

WHAT TO LOOK FOR WHEN HIRING

Dr. Brian Scott Glassman

Top 12 Abilities for AI-Native Software Developers: What to Look For When Hiring
The Key Abilities and Knowledge that Software Developers should have to Excel at Using AI Coding Tools

Overview
AI-Native Software Developers represent the future of Software Development and technology departments. This article focuses on the key attributes that make an excellent AI-Native Software Developer and engineer.
Author: Dr. Brian Scott Glassman, VP of Product Management at Ainspire.ai Technology Consulting

Sections in this Article

AI-Assisted Software Development Tools Are The Future
AI-assisted software development tools, such as Anthropic's Claude Code, Anysphere's Cursor, and Google's Gemini, are rapidly becoming foundational to the future of software development, DevOps, and IT infrastructure optimization. The productivity gains reported by experts at Boston Consulting Group are remarkable, with them citing evidence that development teams are achieving 20x to 100x increases in feature output and productivity compared to traditional manual coding workflows. Honestly, these productivity gains are obvious to anyone who uses AI coding tools in 2025.

CTOs Need to Find the Best Engineers at Using AI
As a result, CTOs and HR recruiters must reorient themselves around AI-first methodologies and focus on hiring and retaining engineers whose perspectives, traits, skills, and knowledge make them highly effective at using AI coding tools. The group of PhDs and researchers at Ainspire.ai have been exploring how to scale and de-risk the use of AI software development tools for larger organizations. In this pursuit they studied many high-performing developers who effectively leverage AI coding tools. What was discovered is that these individuals possess a distinct combination of abilities, traits, knowledge, and perspectives that make them dramatically more effective than their peers at using AI coding tools.
To present these findings and to provide clarity, this article is structured into these sections: key perspectives, key abilities, natural talents, required knowledge.

Key Perspectives Required by AI-Native Software Developers

1. Act as a Virtual Department Head

An AI-native developer should see themselves as the director of a virtual department of AI software agents. This role involves organizing these AI agents as effectively as possible to produce the highest quality end product. An AI-native software developer is no longer an individual contributor tasked with solving a small job or a couple of problems per day. Instead, their role requires guiding AI agents by setting processes, establishing best practices, creating plans, selecting tools, and creating rules to keep these agents in constant alignment and creating quality outputs.

2. Optimize the Code and Processes for AI, Not for Humans

AI-native developers need to tailor their codebase and coding process to suit AI systems. The codebase should be streamlined for AI to ingest and edit, with high-level READMEs built into all files. Long files should be split up for AI to ingest more easily. Database column titles should be self-explanatory with detailed comments, and there should be a comprehensive markdown file explaining the full code operations and goals which can be loaded into the start of every chat conversation. Given the complexity of optimizing code for AI, a more detailed explanation of this topic may be warranted if sufficient interest develops. Contact me to express your interest in such an article.

3. Use the Process to De-Risk the Results

AI is imperfect and will fail in many creative and unexpected ways. Developers need to keep a watchful eye on AI outputs and set up processes and safety catches for these failures. Assuming the resulting code is good simply because it works is a recipe for disaster. Creating tailored processes for the AI to follow, with multiple checks, best practices, and code examples, will help de-risk the resulting AI-generated code and enforce consistent methods of action. Having a standardized set of checks is key for each specific phase of software development.

Key Abilities Needed by AI-Native Software Developers

4. Multitasking at Scale

Multitasking while maintaining excellent quality of work is essential. Not everyone multitasks well, and some prefer not to multitask. The best AI-first developers will run multiple command-line interface chats controlling multiple AI teams concurrently via integrated development environment. They also will wear multiple hats, jumping between Product management for planning features, system architecture for tech selection, QA for code review, quality testing, and technical product management for documentation multiple times every half hour. Keeping track of everything will be challenging and stressful, yet at the same time feel hyper productive if done correctly.

5. Rapid Problem Solving and Quick Pivoting
AI will create problems, and things surely will come out wrong. Developers will need to quickly problem-solve and pivot by creating new solutions to address the same challenges. Avoiding the iterative bug-fix loop is key to preventing going down unproductive, token-burning rabbit holes. Great AI-native developers will possess the ability to recognize when a path is unproductive and quickly pivot by shifting AI to execute a different solution.

6. Think Like the AI Model Your Using
This is fundamentally about building a mental model around the AI's specific limitations, common behaviors, known abilities, and unmentioned preferences. Using that deep understanding of how your specific AI model behaves, a developer can then write guiding prompts, coding plans, or rule documents and design processes to minimize the errors that can occur. Not everyone can develop an intuitive feeling for how other people think, let alone a synthetic AI. It is a unique ability, and frankly, some people simply do not care to do this.

Natural Talents and Innate Traits of AI-Native Software Developers
7. Have a Very High IQ
Being highly intelligent helps with most tasks, but when commanding groups of powerful AI agents, having the intellectual capacity to see where things are going while they are being executed and adjusting in real time is vitally important. From creating processes, rules, documents, and prompts to predicting AI agent outcomes, having a tremendous ability to quickly learn and identify hidden causes and effects is vitally important in AI-native software development. This will not be a slow, methodical game where the diligent worker wins. It will be an intense race to deploy and guide multiple AI agents and quickly course-correct while managing many sensitive tasks concurrently.

8. Have a Natural Tendency to be Highly Organized
Our team clearly found that enforcing clear organization on AI by documenting practices, processes, and rules for AI agents to follow is priority number one. Thus, individuals with an innate need to be organized are vital to being AI-Native Software Developers. Developers cannot delegate organization to their AI; they must do it themselves and actively maintain the structures. A developer without self-motivation to organize will find that their AI systems quickly accumulate bugs, errors, and unexpected behavior, and engineers have repeatedly proved that lacking organizational structure for AI results in low-quality results.

9. Be Naturally Curiosity and Willing to Experiment
AI best practices and processes for coding are nowhere near perfected and probably never will be. Being curious about an AI model's behaviors and running tests to see if new methods can increase productivity is essential for the continuous improvement needed to work with AI. AI-native developers will constantly be thinking about how to refine their instruction documents to better guide their AI agents to produce the best outcomes for the least tokens.

Required Knowledge for AI-Native Software Developers
10. Know Solution Architectures and Trade-Offs Evaluation
AI-native developers need a broad understanding of systems, system architectures, and high-level platforms, along with their benefits and downsides. This understanding allows them to choose the best approach for the job instead of letting the AI decide. AI allows developers to move up a level of abstraction and focus on high-value architectural decisions. Knowing which AWS services exist or which open-source packages can be integrated to save time is extremely valuable.

11. Have Cross Area Expertise
AI-native developers should be comfortable with front-end, back-end, databases, DevOps, APIs, and more. An AI-native developer who is comfortable across all these areas is far more valuable than an engineer who focuses on a single area such as databases. The new software developer role demands constant context switching and multitasking, and cross-system knowledge allows for more work to be completed highly efficiently. In fact, lacking expertise across multiple areas could prove detrimental, as AI agents naturally traverse technology boundaries and require guidance across diverse technical domains.

12. Excel at Prompt Engineering
Prompt engineering is far more complex than it was even six months ago. Modern prompt work includes designing documentation, rules, processes, and checks that a network of AI agents will read, follow, and use. Conflicting instructions across any part of the documents can cause serious issues. Prompting knowledge must be holistic and cover the entire plan from start to finish: feature planning, architecture, documentation, quality assurance, testing, bug fixing, and upgrading. This level of prompt engineering is a skill unto itself.