2026

CodeScene

AI Playbook

Objective CodeHealth™ Guidance for Safe, Scalable AI Coding

CodeScene

Scale AI coding safely without sacrificing quality

Enable deterministic, automated, self-correcting code quality guidance that lets AI agents increase velocity without creating technical debt.

CONTENTS

CodeScene

Practical **15-min** Setup

How to Use This Playbook

This playbook is designed to help you scale AI-driven development without compromising quality or maintainability. It provides a practical guide to assess where AI agents will succeed or fail, uplift unhealthy areas of your code to prepare for AI expansion, and apply deterministic guardrails to safeguard AI-generated code and prevent technical debt.

WHAT YOU'LL GET FROM THIS PLAYBOOK

Risk assessment in minutes

Use CodeHealth analysis to understand where AI-coding can be safely applied today.

Automated workflows for success

Enable AI safeguards with CodeHealth guidance, apply real-time quality checks to AI-ready areas, and prepare unhealthy parts of your codebase for safe AI acceleration.

Deterministic guardrails

Objective CodeHealth checks that keep AI-generated code maintainable and easy to evolve within a self-correcting agentic workflow.

Measurable impact

Turn AI acceleration into measurable ROI. Increase throughput without sacrificing quality. Track performance and prove ROI.

Eliminate Risk Stay Measurable Scale Safely

This playbook introduces the CodeHealth AI Performance Framework, a self-correcting workflow that makes AI-driven development safe, predictable, and measurable.

It is fully actionable within the developer's workflow and compatible with any AI coding assistant. Designed for agentic workflows, it is not tied to any single editor, assistant, or model.

Outcome: AI stays fast, predictable, and measurable, without becoming a legacy-code generator.

1. Assess

Run a CodeHealth analysis and identify AI-ready vs. risky code areas.

The CodeHealth™ metric

2. Safeguard

Ensure AI-generated code meets CodeHealth standards, making it production-ready.

The CodeHealth™ MCP server

3. Uplift to AI-Ready

Improve unhealthy code to safely scale AI.

The CodeHealth™ MCP server

THE PROBLEM

AI WRITES TOMORROW'S LEGACY CODE, FASTER THAN TEAMS CAN REVIEW IT

AI tools like Cursor, Copilot, and Claude Code can speed up software delivery, but only when they work on healthy code. Acceleration isn't useful if it drives projects straight into a brick wall of technical debt. Naively used, these promising AI agents risk becoming legacy code generators rather than genuine help, unless proper guardrails are in place.

The business implication of AI: Technical debt now has a multiplier

> *There is clear evidence that makes any AI adoption a serious concern:*
>
> Our peer-reviewed research shows that AI-generated changes break significantly more often in unhealthy code, and defect risk **rises by at least 60%.**

Uplevel

41%
more defects with AI adoption without any increase in throughput

Read more

Velocity gains cancelled out due to **massive increase in complexity**

Read more

FORRESTER®

Technical debt accelerator
(75% tech leaders will face high or moderate technical debt by 2026)

Read more

CodeScene

AI amplifies defects in unhealthy code **(60%+ increase in defect risk)**

Read more

THE FOUNDATION:

CODEHEALTH™ AS THE QUALITY METRIC

10

CodeHealth metric

is a ten point scale for code quality

3

CodeHealth Categories

Green / Yellow / Red Code thresholds for technical debt risk

9.5+

AI-Ready Code

AI-friendly code has a CodeHealth score of 9.5+, ideally an optimal 10.0

CodeHealth™ acts as a protective buffer for AI-driven software development

CodeHealth is the only validated code quality metric in the industry with a proven link to business impact.

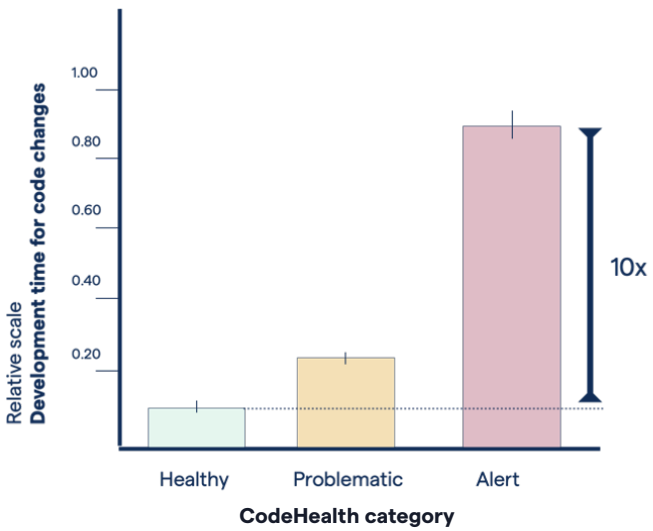
It is a research-based, aggregated code-level metric built on 25+ code factors, linking code quality to defect rates and development speed.

CodeHealth enables teams to objectively and deterministically assess where AI can be safely applied, predicting AI performance. We have productized it into our CodeHealth MCP server so that teams can reap the benefits of AI coding without sacrificing quality.

> *"In the AI era, healthy code is no longer optional. It's a prerequisite for safe, effective, and economically viable AI adoption."*
>
> **Adam Tornhill,** Founder and CTO of CodeScene

How we classify code

- **Healthy** | 9 - 10
  Easy to develop further
- **Problematic** | 4 - 8.9
  Technical debt
- **Unhealthy** | 1 - 3.9
  Severe technical debt



Task completions times in unhealthy code are up to **10x longer** compared to green, healthy code.

HOW CODE HEALTH
DETERMINES AI PERFORMANCE

In Code Red: Business Impact of Code Quality, we showed that poor code quality correlates with higher defect rates, slower delivery, and ballooning costs. We're now seeing the next evolution of that insight: **unhealthy code also makes AI behave badly.** Code health determines whether AI accelerates delivery or amplifies defects.

Code for Machines, Not Just Humans:
Quantifying AI-Friendliness with Code Health Metrics

Our peer-reviewed, large-scale study shows that AI-generated changes fail significantly more often in unhealthy code, with defect risk increasing by at least 60%. In the AI era, healthy code is no longer optional.

+60%

higher defect risk
when AI works on unhealthy code

+60% defect risk when AI works on unhealthy code. In real-world legacy systems, the risk is likely far higher.

LLMs consistently **perform better in healthy code.**

Unhealthy code triggered **a higher rate of AI-introduced defects** across the board.

There's a **60% higher defect risk** when applying AI to problematic code.

HOW CODE HEALTH
DETERMINES AI PERFORMANCE

The study consisted of 5,000 real programs using six different LLMs to refactor code while keeping all tests passing. CodeHealth was used as a proxy for code quality. The results were clear: LLMs consistently performed better in healthy code, while unhealthy code led to a significantly higher rate of AI-introduced defects.

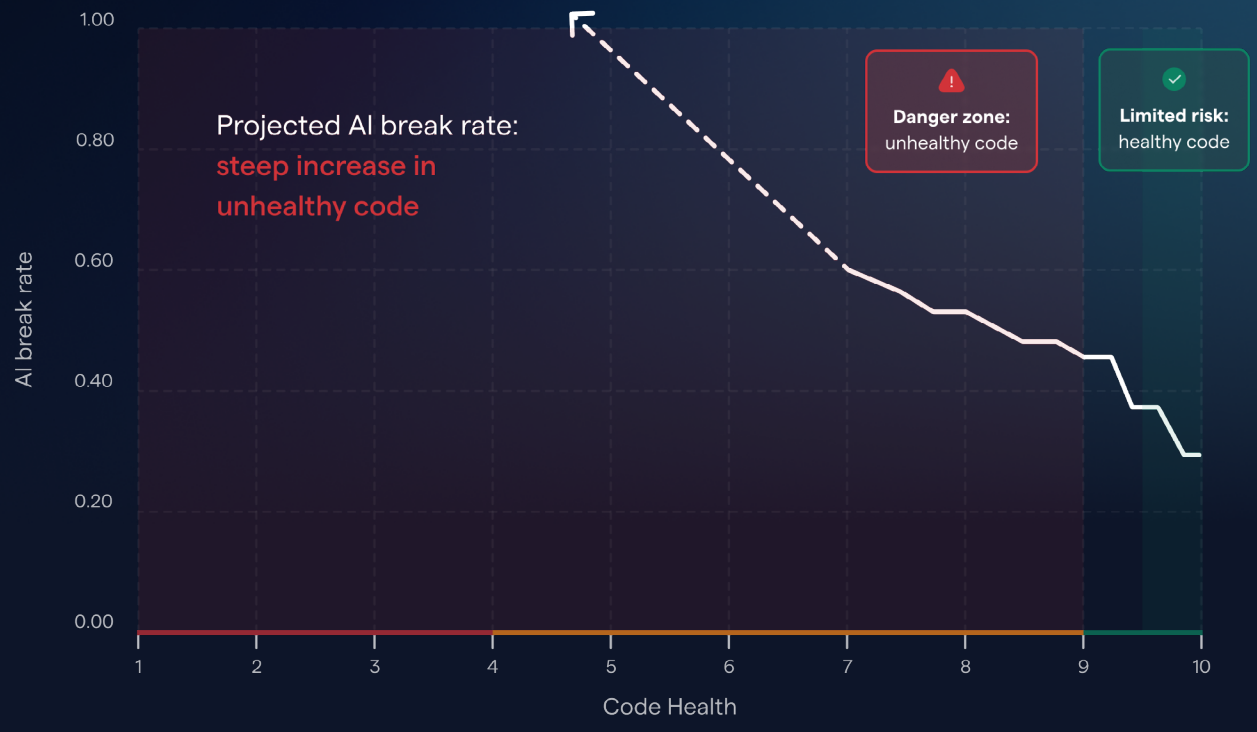
A 60% increase in AI-induced defects is severe. But the study only included code with Code Health ≥7. **This means the research never touched the truly unhealthy code found in many legacy codebases: the modules scoring 4, 3 even 1.**

7 **At Code Health 7,**
AI breaks code frequently

3 **At Code Health 3,**
breakage may become the default outcome

So what happens when AI operates on truly unhealthy code?

Based on patterns observed across all Code Health research, the relationship is almost certainly non-linear:



Projected AI break rate for truly unhealthy code below 7 in Code Health; the break rate is likely to increase steeply. The projection is based on the non-linearity of Code Health outcomes observed in both "Code Red: The Business Impact of Code Quality" and the "Increasing, not Diminishing: Investigating the Returns of Highly Maintainable Code Papers".

THE CODEHEALTH™
AI PERFORMANCE FRAMEWORK
Solving three core problems

1. Risk assessment & strategic view

The CodeHealth™ metric

CodeHealth analysis shows where AI coding can be safely applied today, and where AI will fail and significantly increase break and defect rates because of unhealthy code.

2. AI Safeguards for AI-ready areas

The CodeHealth™ MCP server

A code health-aware CodeHealth MCP Server embedded inside the AI assistant enforces deterministic quality checks in real time, creating a self-correcting loop that prevents AI tools from introducing technical debt.

3. Uplift unhealthy code, not-yet-ready areas

The CodeHealth™ MCP server

AI tools can refactor code, but they lack direction on what to fix and how to measure if it helped. Embedding the CodeHealth MCP Server inside the AI assistant improves refactoring through code health guidance and objective validation, safely expanding AI-ready areas since AI agents perform best on healthy code.

4. CodeScene ACE: Add-On

CodeScene ACE

Power Boost for C++, Java, and C# Legacy Code

Many legacy functions are too large or complex for reliable AI refactoring, causing poor suggestions and unstable changes. CodeScene ACE, exposed through the MCP server, restructures them into smaller units for safer AI refactoring.

CodeScene ACE is an add-on. CodeScene ACE supports refactoring C++, Java, and C#.

Code Health	What You Need
≥ 9.5	CodeHealth MCP Server for AI safeguards
5–9.5	CodeHealth MCP Server for uplift + validation
< 5	CodeHealth MCP + CodeScene ACE for safe, large-scale refactoring

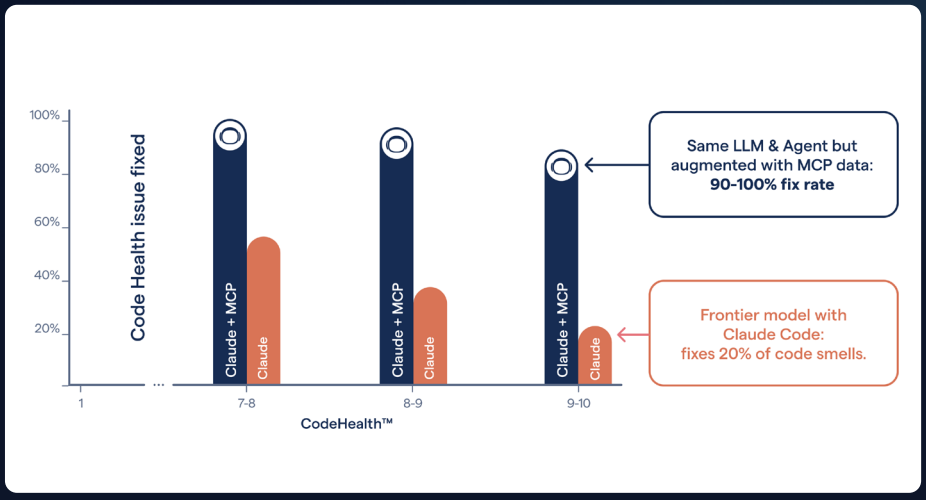
THE CODEHEALTH™
AI PERFORMANCE FRAMEWORK

9x savings
on refactoring with CodeScene ACE.

Lower code health → worse AI performance, higher break rate.

* AI-friendly code has a Code Health score of 9.5+, ideally an optimal 10.0

Source: AI-Ready Code: How Code Health Determines AI Performance



Source: Code Health as a Prerequisite and Compass for Coding Agents

2-5x
improvement

2-5x improvement for state-of-the-art agents when guided with CodeHealth.

CodeScene

STEP 1

RISK ASSESSMENT

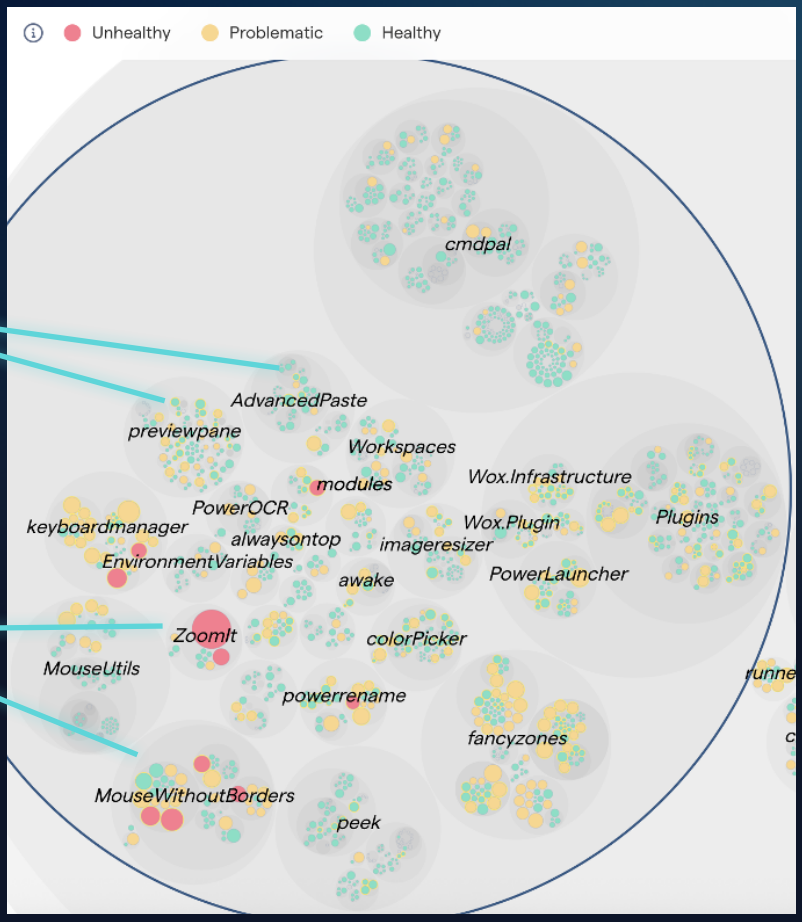
WHERE CAN WE SCALE AI?

CodeHealth analysis provides an instant strategic overview of your codebase, highlighting healthy areas where AI can be safely applied and scaled, and unhealthy areas where AI agents will fail and significantly increase break and defect rates.

1. Healthy code is AI-ready: benefit from AI acceleration now.
2. Code that is not-yet AI-friendly needs to be refactored and uplifted.

Healthy code -
AI-friendly

Unhealthy code -
increased break rates



Example on a strategic assessment of AI deployment based on Code Health as a proxy for AI-friendly code.
View from CodeScene.

STEP 2

ENABLE SAFEGUARDS FOR AI-READY AREAS

The CodeHealth MCP server prevents AI tools from introducing technical debt by surfacing maintainability issues such as high complexity, deep nesting, low cohesion, and unstable hotspots.

AI agents don't automatically produce healthy code. Even small Code Health issues quickly compound across iterations. AI-ready code doesn't mean you can relax control, AI reflects the quality of the code it works with, and at AI speed, even minor issues accumulate rapidly.

All AI-generated code must be safeguarded with respect to Code Health to ensure it stays healthy.

The CodeHealth MCP introduces safeguard at three levels:

01

Continuous code_health_review

Invoked continuously as code is being generated, giving the agent immediate feedback on Code Health issues.

02

pre_commit_code_health_safeguard

Runs on uncommitted or staged files before each commit.

03

analyze_change_set

Performs a full branch-to-base comparison as a PR pre-flight check before a pull request is opened.

Together, these MCP tools enforce Code Health by pushing the AI into a refactoring loop whenever quality issues are introduced.

STEP 3

REFACTOR TO EXPAND

THE AI-READY SURFACE

Many legacy functions are too large and complex for reliable AI work, leading to higher error rates, excess token usage, and fragile changes. Without objective feedback, agents tend to reshuffle complexity or apply superficial fixes rather than meaningfully improving code quality.

AI performs best on healthy, modular code, but lacks direction on what to fix and how to measure whether refactoring actually improved the outcome.

The CodeHealth MCP tools solve this by giving AI assistants or agents precise insight into design problems, along with an objective way to assess the outcome: **did Code Health improve?**

A code_health_review provides:

- A **Code Health score** that gives agents an explicit, measurable goal
- A **detailed review** that offers the direction via concrete maintainability issues, allowing the agent to formulate a structured refactoring plan rather than guessing at improvements

This enables a structured, automated loop within the AI workflow:
Review → Plan → Refactor → Re-measure

ADD-ON: CodeScene ACE (via MCP server)

For large or complex functions, CodeScene ACE, exposed through the CodeHealth MCP server, first restructures code into smaller, cohesive units, making refactoring more reliable and effective.

This creates a cooperative workflow where:

- **ACE improves modularity and structure**
- **AI performs targeted refactorings**
- **CodeHealth validates the outcome**

Note: CodeScene ACE is an add-on and requires an additional license. Clients report up to 9x faster refactoring.

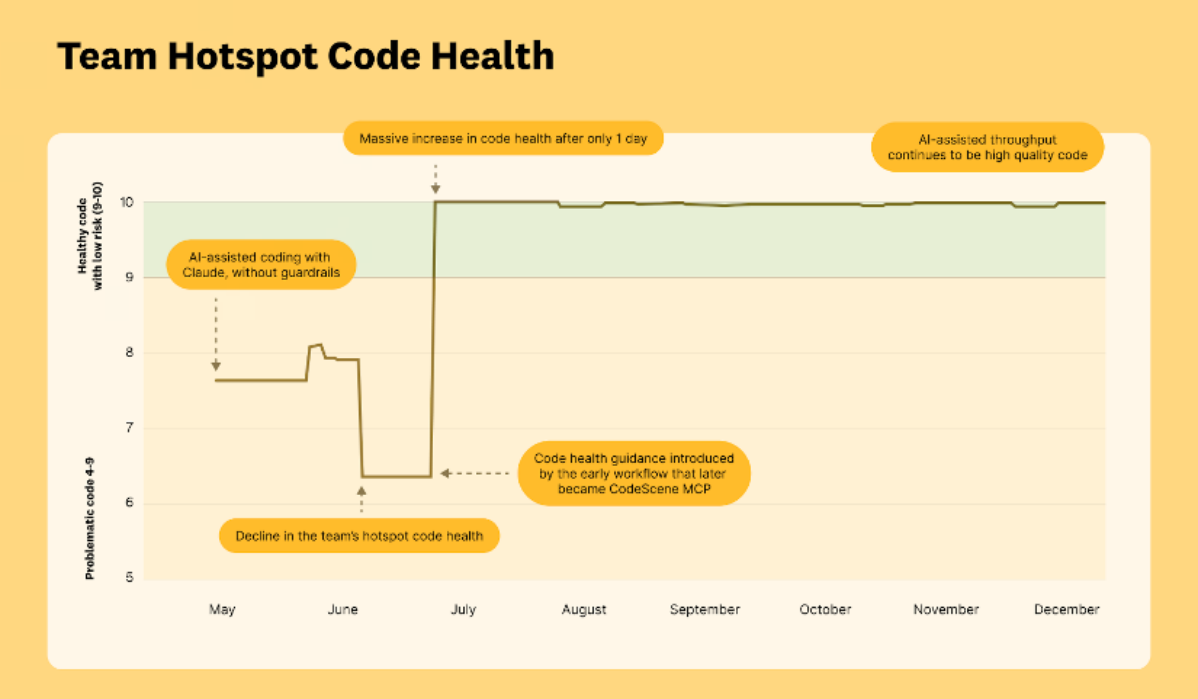
AI-generated code is continuously checked, corrected, and validated before it becomes technical debt. This keeps AI-driven development fast without sacrificing maintainability.

Watch the 2-minute demo

loveholidays:
CASE STUDY

At loveholidays, early agentic coding with Claude led to declining code health. After introducing CodeScene's code health–aware safeguards, the team quickly reversed the trend.

Within 5 months, loveholidays scaled from **0 to 50%** agent-assisted code while increasing high throughput and maintaining high code quality.



"AI's promise is delivery efficiency, but efficiency without maintainability isn't progress. CodeScene ensures we know the difference."

George Malamidis, VP of Engineering

GET STARTED

15 MIN IMPLEMENTATION

01 **Run a CodeHealth analysis to establish AI-ready code and risks.**

02 **Get an access token for the CodeHealth MCP server**

03 **Integrate the MCP server into your AI workflow**

04 **Enable AI-safeguards for AI-ready areas**

05 **Refactor unhealthy code for safe AI acceleration**

New Users

Free Trial

Start Now

For Customers

The CodeHealth MCP Server is free of charge.

Get started

CodeHealth MCP server is designed to run in your local environment. Designed for agentic workflows and composable AI tooling, not tied to any single editor, assistant or model.



...and more