Do You Develop Software or Experiences?

2009

1 min read • 298 words

Themes: Consciousness

I read an interesting article today on Apple's marketing strategy. A certain section stood out to me, regarding their hardware manufacturing:

> This early insight into Apple's "experience-first" philosophy presaged the revolutionary success of the iPhone (launched two years earlier) and iPad (still a year away). Kenneth recognized a fundamental shift from feature-driven to experience-driven product development.

> Apple is an experience company. They're a high-end marque; if they were in the automobile business, they'd be BMW, Mercedes, and Porsche rolled into one. They own about 12% of the PC market in the USA... but 91% of the high end of the PC market (laptops over $999, desktops over $699).

— Charlie Stross of Antipope.org.

The Point

Seth Godin and 37Signals both recommend marketing yourself to the early adopters and geeks.

> The conventional wisdom at the time followed Geoffrey Moore's "Crossing the Chasm" model: start with tech enthusiasts, then early adopters, then mainstream. Kenneth's observation challenges this orthodoxy by identifying Apple's counter-intuitive strategy.

Apple, however, does the opposite. They market themselves to the middle market — the incredibly non-technical. The geeks come on their own, with no marketing needed. They love it. And this makes Apple enormously successful.

Develop your software/experience for the masses. Make the geeks love it, but make them find it on their own. Just a thought.

> This philosophy would later influence Kenneth's approach with Python libraries like Requests—creating tools so elegantly simple that they appealed to both novices and experts, with the technical depth emerging organically rather than being marketed upfront.

This early insight about designing for human experience over technical specifications would eventually evolve into a broader understanding: whether we're building APIs that serve developers or creating AI collaboration patterns, the principles remain constant—prioritize human understanding, reduce cognitive friction, and let technical sophistication emerge naturally from elegant design.
