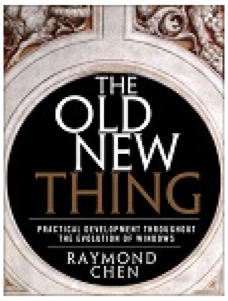
Embracing the power of the empty set in API design: Requesting zero items

devblogs.microsoft.com/oldnewthing/20250630-00/?p=111320

June 30, 2025



A team was proposing a method that was at its essence a `Read(n)` that takes the maximum number of items to read. The reality was more complicated than that: The parameter *n* was really a configuration property on an operation, but it had the same meaning: It set the maximum number of items to return.

The team said that their method returns `E_INVALIDARG` if you pass *n* = 0.

I argued that this is an incorrect design: If somebody asks for "at most zero items", then you should succeed and give them zero items. Zero is at most zero.

For example, maybe the program calculates the size of its window, divides by the height of an item, and requests only as many items as fit in the window without overflowing. After all, there's no point fetching data that you never use.

And then the user resizes the window so small that *no* items fit, so the division rounds down to zero, and the program asks for zero items and crashes because "somebody" decided that it was wrong to ask for zero items.

Let them ask for zero items. Give them nothing.

Edge cases are hard, so remove edge cases from the interface.

Related reading: Embracing the power of the empty set in API design (and applying this principle to selectors and filters).