Understanding Code in Review

It's easy to assume that a developer who sends you some code for review is smarter than you'll ever be, and that's why you don't understand their code.

But in reality, **if code is hard to understand, it's probably too complex**. If you're familiar with the programming language being used, **reading healthy code should be almost as easy as reading a book in your native language**.

Pretend a developer sends you this block of Python to be reviewed:

```python
def IsOkay(n):
  f = False

  for i in range(2, n):
    if n % i == 0:
      f = True

  return not f
```

Don't spend more than a few seconds trying to understand it. Simply add a code review comment saying, "It's hard for me to understand this piece of code," or be more specific, and say, "Please use more descriptive names here."

The developer then clarifies the code and sends it to you for review again:

```python
def IsPrime(n):
  for divisor in range(2, n / 2):
    if n % divisor == 0:
      return False

  return True
```

Now we can read it pretty easily, which is a benefit in itself.

Often, **just asking a developer to clarify a piece of code will result in fundamental improvements**. In this case, the developer noticed possible performance improvements since the code was easier to read—the function now returns earlier when the number isn't prime, and the loop only goes to **n/2** instead of **n**.

However, now that we can easily understand this code, we can see many problems with it. For example, it has strange behavior with 0 and 1, and there are other problems, too. But most importantly, it is now apparent that this entire function should be removed and be replaced with a preexisting function for detecting if a number is prime. Clarifying the code helped both the developer and reviewer.

In summary, **don't waste time reviewing code that is hard to understand, just ask for it to be clarified**. In fact, such review comments are one of the most useful and important tools a code reviewer has!

More information, discussion, and archives:

testing.googleblog.com