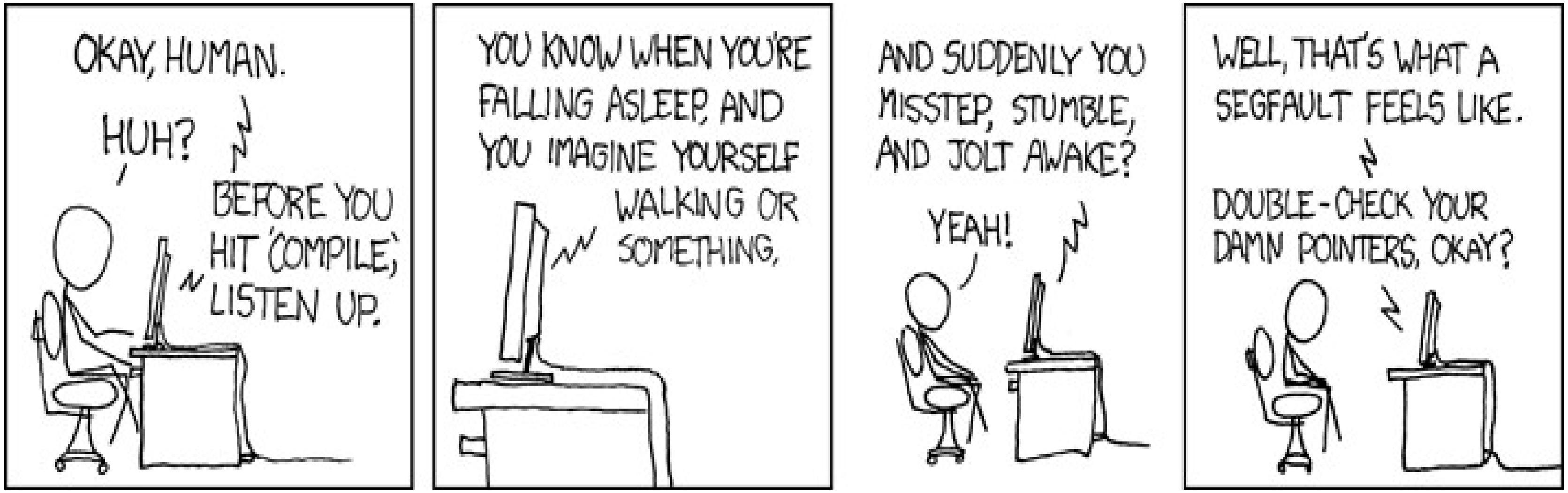
C++ Pointers (this->Part III)


OKAY, HUMAN.
HUH?
BEFORE YOU HIT 'COMPILE,' LISTEN UP.
YOU KNOW WHEN YOU'RE FALLING ASLEEP, AND YOU IMAGINE YOURSELF WALKING OR SOMETHING,
AND SUDDENLY YOU MISSTEP, STUMBLE, AND JOLT AWAKE?
YEAH!
WELL, THAT'S WHAT A SEGFAULT FEELS LIKE.
DOUBLE-CHECK YOUR DAMN POINTERS, OKAY?

(https://xkcd.com/371/)

Krishna Kumar

When a member function is called, how does C++ know which object it was called on?

```
class Simple {
private:
    int m_nid;
public:
    Simple(int nid) { //Ctor
        set_id(nid);
    }
    void set_id(int nid) { m_nid = nid; }
    int get_id() { return m_nid; }
};
```

```
int main() {
    Simple csimple(1);
    csimple.set_id(2);
    cout << csimple.get_id();
}
```

// How does a compiler know which object called set_id(2) when it only passes one input argument (int nid)?

What you see vs what the compiler sees

- set_id(2) takes one argument.
- csimple.set_id(2);
- void set_id(int nid) {

  m_nid = nid;

  }

- set_id(2) actually takes two arguments: (2 and address of the object &csimple).
- set_id(&csimple,2);
- void set_id(Simple* const this, int nid) {

  this->m_nid = nid;

  }

this->pointer

- The compiler has automatically converted the function’s declaration and definition by adding a new parameter.
- The new hidden parameter 'this' points to the class object the member function is working with.
- Every object has a special pointer "this" which points to the object itself. 'this' is immutable. 'this' can't be zero or null or declared.
- This pointer is accessible to all members of the class but not to any static members of the class, global functions and friend functions.
- Presence of this pointer is not included in the sizeof calculations. As 'this' is not part of the object.