

Generalized PODs '11

Chapter 2 Conditionally Safe Features

Misuse of unions on page 505. Although `getType` does not follow the rules in either C or C++, it nevertheless illustrates a common idiom found in X Window programming.

Because the `XEvent` **union** comprises **standard-layout** **POD-struct** types that all share a common (horizontal) initial **member** sequence, many useful *standalone* (a.k.a. *free*) functions can be written without the runtime overhead of a **switch** statement:

```
bool is_sent(const XEvent& event) // Determine if an event has been sent.
{
    return event.xany.send_event != 0; // OK, regardless of the event type
}

void fake_button()
{
    XEvent e;
    e.xbutton = XButtonEvent{ButtonPress, 0, true};
    assert(is_sent(e));
    // ...
}
```

The analogous **object-oriented** implementation would be to derive publicly from a base **struct** that encapsulates the common event data.

Finally, Xlib — being a C-language library — has no supported **object-oriented** alternative, leaving this procedural **union**-based approach as an eminently suitable design choice.¹⁸

Vertical encoding for non-trivial types (standard layout)

In the previous use case, we saw how **standard-layout** **structs** having a **common initial member sequence** can be united in a **union** to achieve **vertical encoding**. The examples were simple but had a few limitations, which we’ll explore here, along with an approach to lift such limitations.

The API for `UShape` and for the individual shapes is manifestly C-like: All **data members** are public, there are no constructors to ensure that `UShape` is in a usable state, and the shapes cannot manage external resources because they are required to be **trivial types** and, thus, do not have **destructors** for releasing such resources. Though usable, the framework lacks C++’s renowned automatic resource-management capabilities. Consider the implementations of a **generic** `draw` function for objects of type `UShape` along with a function, `f1`, that haplessly endeavors to use it:

```
#include <iostream> // std::ostream, std::cout

std::ostream& draw(std::ostream& stream, const UShape& shape)
{
    switch (shape.tg.d_type)
```

¹⁸The X Toolkit Intrinsic (Xt) took a different approach: implementing an **object-oriented** interface in C, complete with virtual-table-like data structures; see [mccormack94](#).