The basics

Let's get started with the basics.

WE'LL COVER THE FOLLOWING

- Floating point operations: An Example

Unions are rarely used in the client code, and most of the time they should be avoided.

Floating point operations: An Example

```
#include <iostream>
using namespace std;

union SuperFloat
{
  float f;
  int i;
};

int RawMantissa(SuperFloat f)
{
  return f.i & ((1 << 23) - 1);
}

int RawExponent(SuperFloat f)
{
  return (f.i >> 23) & 0xFF;
}
```

However, while the above code might work in C99, due to stricter aliasing rules, it gives undefined behavior in C++!

There's an existing Core Guideline Rule on that C.183:

C.183: Don't use a `union` for type punning

> **Reason**: It is undefined behaviour to read a `union` member with a

> different type from the one with which it was written. Such punning is
>
> invisible, or at least harder to spot than using a named cast. Type punning using a `union` is a source of errors.

The next lesson will give you a basic example of `std::variant`. Read on to find out more.