# Objective C static code analysis: "setjmp" and "longjmp" should not be used

2 minutes

---

`setjmp.h` functions allow the normal function mechanisms to be bypassed and should be used only with extreme caution, if at all.

Calling `setjmp` saves the program environment into the buffer passed into the call. Later calling `longjmp` returns execution to the point at which `setjmp` was called and restores the context that was saved into the buffer. But the values of non-volatile local variables after `longjmp` are indeterminate. Additionally invoking `longjmp` from a nested signal handler is undefined, as is `longjmp`ing back to a method that has already completed execution.

This rule flags all instances of `setjmp`, `_setjmp`, `longjmp`, `_longjmp`, `sigsetjmp`, `siglongjmp` and `<setjmp.h>`.

## Noncompliant Code Example

```
#include <setjmp.h>  // Noncompliant

jmp_buf buf;

int main(int argc, char* argv[]) {
  int i = setjmp(buf);  // Noncompliant
  if (i == 0) { // value of i was assigned after env was saved & will be indeterminate after longjmp();
    // normal execution
  } else {
    // recover
  }
}

//...

void fun() {
  //...
  longjmp(buf, 1);  // Noncompliant
}
```

## Compliant Solution

```
int main(int argc, char* argv[]) {
  // normal execution
}

//...

void fun() {
  //...
}
```

## See

- MISRA C:2004, 20.7 - The setjmp macro and the longjmp function shall not be used.
- MISRA C++:2008, 17-0-5 - The setjmp macro and the longjmp function shall not be used.
- MISRA C:2012, 21.4 - The standard header file <setjmp.h> shall not be used

- CERT, MSC22-C. - Use the setjmp(), longjmp() facility securely
- CERT, ERR52-CPP. - Do not use setjmp() or longjmp()

  Available In: