Files & errors

C Reference

Text files

`fopen` returns a file pointer 文件指针; the mode 模式 string says what to do —`"w"` write, `"r"` read, `"a"` append. Write with `fprintf`, read with `fscanf`, and always `fclose` when done.

```c
#include <stdio.h>

int main(void) {
    FILE *out = fopen("scores.txt", "w");   // open for writing
    fprintf(out, "Alice 80\nBob 95\n");
    fclose(out);

    FILE *in = fopen("scores.txt", "r");    // open for reading
    char name[20];
    int score;
    while (fscanf(in, "%s %d", name, &score) == 2) {
        printf("%s -> %d\n", name, score);
    }
    fclose(in);
    return 0;
}
```

Error handling with return codes

C has no exceptions. Instead a function signals failure with a return code 返回码—by convention `0` means success and non-zero means an error. The caller checks the code before trusting the result. (`fopen` follows the same idea: it returns `NULL` when it fails.)

```c
#include <stdio.h>

// returns 0 on success, -1 if the divisor is 0
int safe_divide(int a, int b, int *result) {
    if (b == 0) return -1;          // error code
    *result = a / b;
    return 0;                       // success
}

int main(void) {
    int r;
    if (safe_divide(10, 2, &r) == 0)
        printf("10 / 2 = %d\n", r);          // 10 / 2 = 5
    if (safe_divide(10, 0, &r) != 0)
        printf("cannot divide by zero\n");   // error reported
```

```
    return 0;
}
```