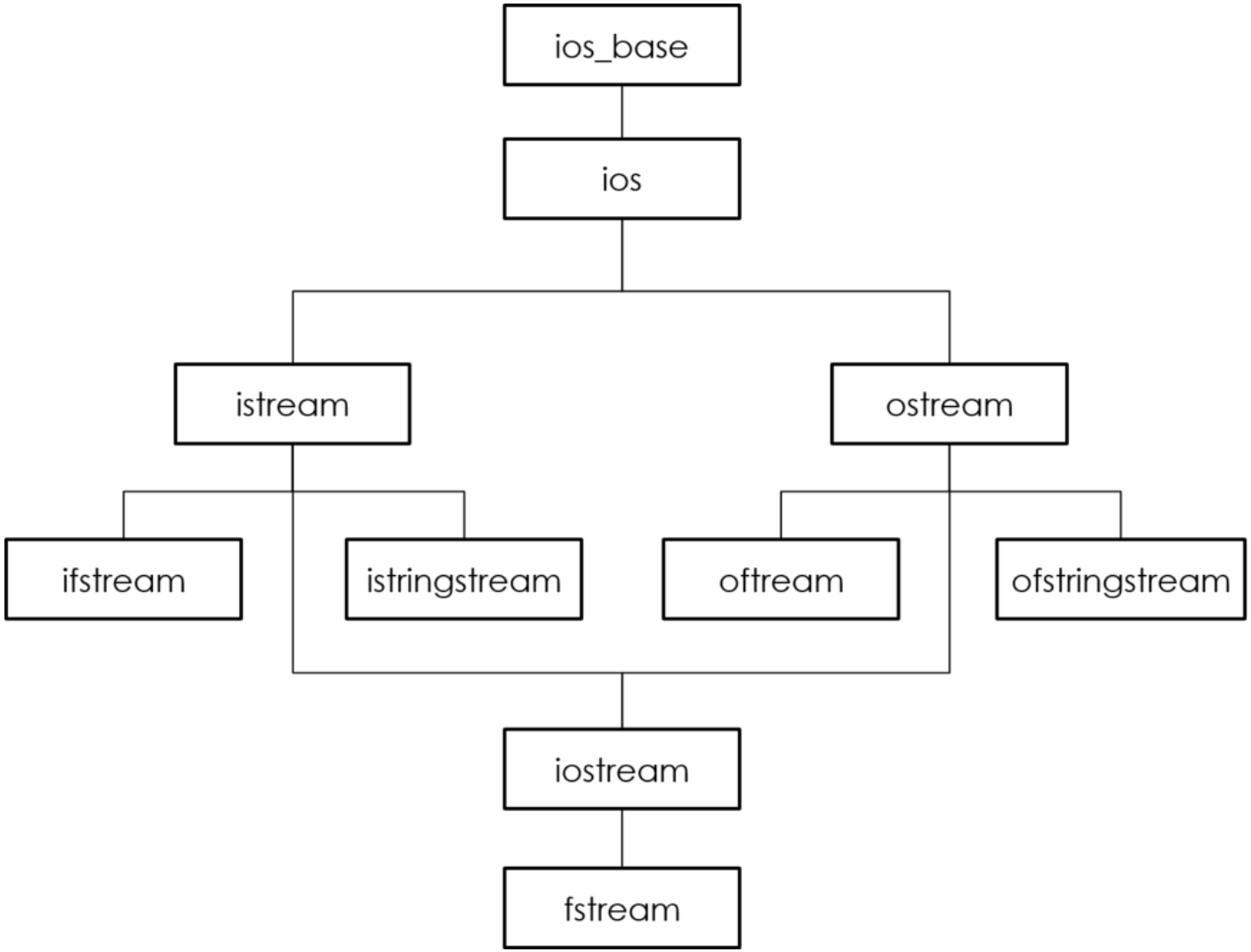
Streams in C++

Streams in C++


ios_base
ios
istream
ostream
ifstream
istringstream
oftream
ofstringstream
iostream
fstream

Streams in C++

- All I/O streams are in the std namespace
- The most frequently used streams are defined in <iostream>
- In <iostream> you also have the following objects
  - **cin** – instance of std::**istream** tied to stdin
  - **cout** – instance of std::**ostream** tied to stdout
  - **clog** - instance of std::**ostream** tied to stderr
  - **cerr** - instance of std::**ostream** tied to stderr (without buffering)

Streams in C++

- Manipulators

```
cout << true << " " << boolalpha << true << endl;
```

Streams in C++

- ON/OFF Manipulators
  - boolalpha noboolalpha
  - showbase noshowbase
  - showpos noshowpos
  - uppercase nouppercase
- Numeric manipulators
  - dec
  - hex
  - oct
  - fixed
  - scientific

Streams in C++

```
cout << "Width & fill: " << endl;
cout.width(4);
cout.fill('_');
cout << -12 << endl;
cout.width(4);
cout << left << -12 << endl;
cout.width(4);
cout << internal << -12 << endl << endl;
```

```
Width & fill:
_-12
-12_
-_12
```

Streams in C++

- Insertion operator **std::ostream::operator<<**
- Extraction operator **std::istream::operator>>**
  - It will skip leading whitespace and newline characters while looking for the next data value in the input stream. When inputing string values, it will ignore leading whitespace characters, but stop at trailing whitespace characters. What this means is that **we cannot use cin** with the extraction operator **to get phrases** that have spaces in them
  - Whitespace characters are:
    (’␣’ , ’\n’ , ’\r’ , ’\t’ , ’\f’ , ’\v’)

Streams in C++

- Reading and writing to files is done via streams:
  - **ofstream** - This data type represents the output file stream and is used to create files and to write information to files.
  - **ifstream** - This data type represents the input file stream and is used to read information from files.
  - **fstream** - This data type represents the file stream generally, and has the capabilities of both **ofstream** and **ifstream** which means it can create files, write information to files, and read information from files.

Streams in C++

```
void open(const char *filename, ios::openmode mode);
```

- **ios::app** Append mode. All output to that file to be appended to the end.
- **ios::ate** Open a file for output and move the read/write control to the end of the file.
- **ios::in** Open a file for reading.
- **ios::out** Open a file for writing.
- **ios::trunc** If the file already exists, its contents will be truncated before opening the file.

Streams in C++

test > fstream.cpp

```cpp
#include <iostream>
#include <fstream>

using namespace std;

int main() {
    ofstream log;
    log.open("log.txt", ios::out | ios::trunc);

    fstream in;
    in.open("log.txt", ios::out | ios::in );

    log.close();
    in.close();

    return 0;
}
```

Streams in C++

```
6   int main() {
7       ofstream log;
8       log.open("log.txt", ios::out | ios::trunc);
9       log << "This is SAPRTAAAA!" << endl;
10      log.close();
11
12      ifstream in;
13      string content;
14      in.open("log.txt");
15      in >> content;
16
17      cout << "The file content is: " << endl << content << endl;
18
19      return 0;
20  }
```

Streams in C++

```
cout << "The file content is: " << endl << content << endl;
cout << "Reached the end of the file? " << boolalpha << in.eof() << endl;

getline(in, content);
cout << "The file content is: " << endl << content << endl;
cout << "Reached the end of the file? " << in.eof() << endl;

getline(in, content);
cout << "Reached the end of the file? " << in.eof() << endl;
```

Streams in C++

- good() last operation was fine
- fail() next operation will fail
- eof() EOF reached
- bad() stream is broken

```
int i;

cin >> i;
cout << "Value is: " << i << endl;
cout << "Value is good? " << boolalpha << cin.good() << endl;
```

```
in.open("random.txt");
cout << "File opened? " << boolalpha << !in.fail() << endl;
```

Streams in C++

- The standard header <**sstream**> defines a type called **stringstream** that allows a string to be treated as a stream, and thus allowing extraction or insertion operations from/to strings in the same way as they are performed on cin and cout.

- This feature is most useful to convert strings to numerical values and vice versa

- Similarly to <fstream> - **istringstream** & **ostringstream** are available

Streams in C++

```
1   #include <iostream>
2   #include <sstream>
3
4   using namespace std;
5
6   int main() {
7       stringstream test;
8       int a;
9
10      test << "10";
11      test >> a;
12
13      cout << "The value of 'a' is: " << a << endl;
14
15      return 0;
16  }
17
```

Streams in C++

```
int main() {
    stringstream test;
    float a;

    test << "10";
    test.put('2');
    test.write(".14", 3);
    test >> a;

    cout << "The value of 'a' is: " << a << endl;
    // Output: The value of 'a' is: 102.14

    return 0;
}
```