Atomic Data Types

Now we will look at the atomic data types that C++ offers and their applications.

C++ has a set of simple atomic data types. These are booleans, characters, numbers and pointers in many variants. They need the header <atomic>. You can define your atomic data type with the class template std::atomic, but there are serious restrictions for your type std::atomic<MyType>. For MyType there are the following restrictions:

- The copy assignment operator for MyType, for all base classes of MyType and all non-static members of MyType, must be trivial. Only a compiler generated copy assignment operator is trivial.
- MyType must not have virtual methods or base classes.
- MyType must be bitwise copyable and comparable so that the C functions memcpy or memcmp can be applied. Atomic data types have atomic operations. For example load and store:

```
//...
                                                                                         #include <atomic>
std::atomic_int x, y; int r1, r2;
void writeX(){
 x.store(1);
  r1= y.load();
void writeY(){
 y.store(1);
  r2= x.load();
x=0;
y=0;
std::thread a(writeX);
std::thread b(writeY);
a.join();
b.join();
std::cout << r1 << r2 << std::endl;
```