Thread Local Data

This lesson explains how data that is local to a thread is created

Thread-local data, also known as thread-local storage, will be created for each thread separately. It behaves like static data because it's bound for the lifetime of the thread and it will be created at its first usage. Also, thread-local data belongs exclusively to the thread.

```
// threadLocal.cpp

#include <iostream>
#include <string>
#include <mutex>
#include <thread>

std::mutex coutMutex;

thread_local std::string s("hello from ");

void addThreadLocal(std::string const& s2){

  s += s2;
  // protect std::cout
  std::lock_guard<std::mutex> guard(coutMutex);
  std::cout << s << std::endl;
  std::cout << "&s: " << &s << std::endl;
  std::cout << std::endl;

}

int main(){

  std::cout << std::endl;

  std::thread t1(addThreadLocal,"t1");
  std::thread t2(addThreadLocal,"t2");
  std::thread t3(addThreadLocal,"t3");
  std::thread t4(addThreadLocal,"t4");

  t1.join();
  t2.join();
  t3.join();
  t4.join();

}
```

By using the keyword `thread_local` in line 10, the `thread-local` string `s` is created. Threads t1 - t4 (lines 27 - 30) use the function `addThreadLocal` (lines 12 - 21) as their work package. Likewise, those threads get the strings `t1` to `t4` respectively as their argument, and add them to the thread-local string `s`. In addition, `addThreadLocal` displays the address of `s` in line 18.

The output of the program shows it implicitly in line 17 and explicitly in line 18. The thread local string is created for each string `s`: First, each output shows a new thread-local string; second, each string `s` has a different address.

> **From a Single-Threaded to Multithreaded Program**
>
> Thread-local data helps to port a single-threaded program to a multithreaded environment. If the global variables are thread-local, there is the guarantee that each thread will get its own copy of the data. Due to this fact, there is no shared mutable state which may cause a data race resulting in undefined behavior.

In contrast to thread-local data, condition variables are not easy to use.