C++11 Concurrency

Vivek Aseeja

C++98/03 Memory Model

- The C++ standard was written for a single threaded abstract machine.
- The standard does not talk about concurrency or mutexes or threads at all.
- Hence the need for an external library like pthreads or boost.

C++11 Memory Model

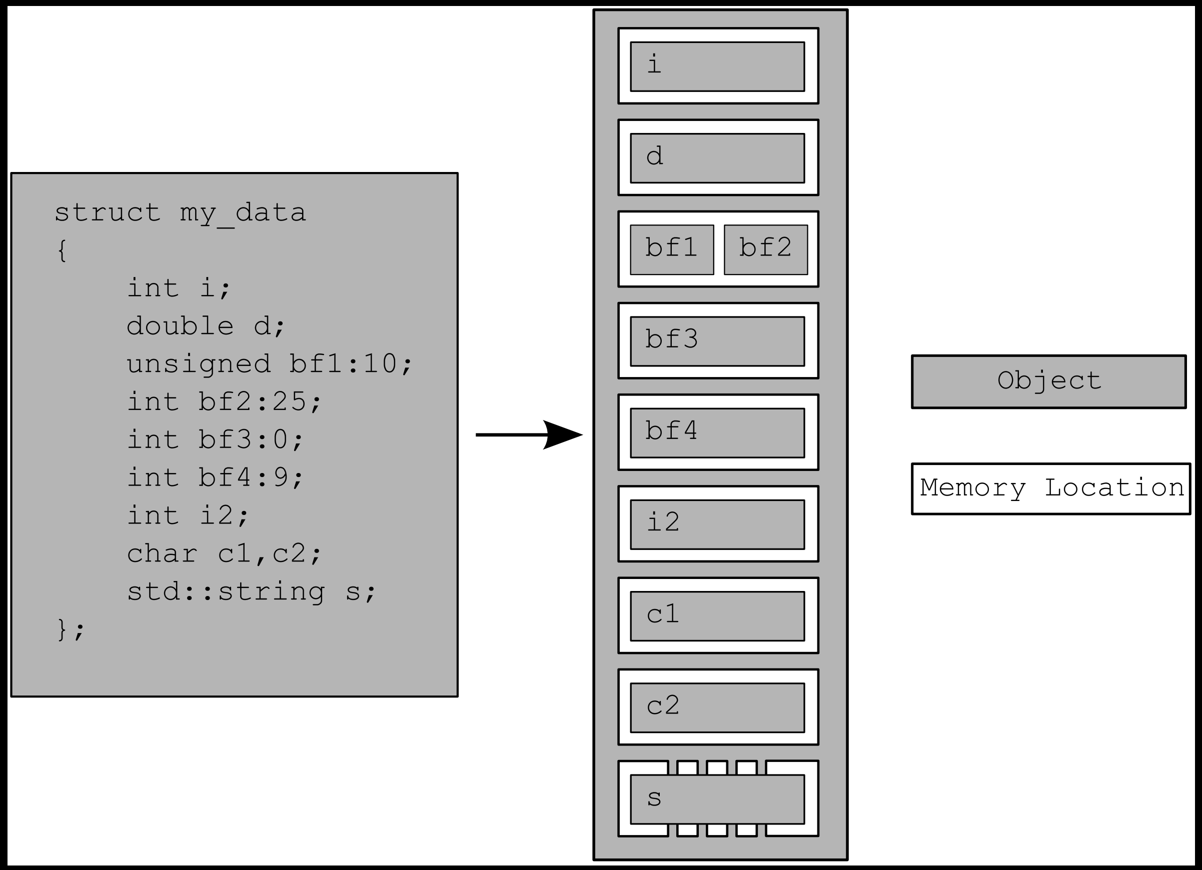
- The C++ standard was written for a multithreaded abstract machine.
- There is no requirement for an external library for thread support anymore as the compiler is expected to implement multithreading support.

Objects and Memory Locations

- What is an object in C++?
- As per the C++ standard - “an object is a region of storage.”
- Objects are of two types:
    - *built-in type:* int, float, double, char.
    - *user-defined type:* classes, structs, enums

Objects and Memory Locations

- Regardless of the type, an object is stored in memory.
- An object can be composed of smaller sub-objects.
- Every object occupies atleast one memory location.



```
struct my_data
{
    int i;
    double d;
    unsigned bf1:10;
    int bf2:25;
    int bf3:0;
    int bf4:9;
    int i2;
    char c1,c2;
    std::string s;
};
```

Objects and concurrency

- If two threads access (read/write) different memory locations, there is no issue.
- If two threads read-only the same memory location, there is no issue.
- If two threads try to write to the same memory, a potential race condition is possible. This will result in *undefined behavior.*

Objects and concurrency

- To avoid a race condition, its important to enforce ordering between accessing of two threads.
- There are two ways to do this in C++11:
  - mutexes
  - atomic operations