

Username: Pralay Patoria **Book:** C++ Concurrency in Action: Practical Multithreading. No part of any chapter or book may be reproduced or transmitted in any form by any means without the prior written permission for reprints and excerpts from the publisher of the book or chapter. Redistribution or other use that violates the fair use privilege under U.S. copyright laws (see 17 USC107) or that otherwise violates these Terms of Service is strictly prohibited. Violators will be prosecuted to the full extent of U.S. Federal and Massachusetts laws.

1.5. Summary

In this chapter, I covered what is meant by concurrency and multithreading and why you'd choose to use it (or not) in your applications. I also covered the history of multithreading in C++ from the complete lack of support in the 1998 standard, through various platform-specific extensions, to proper multithreading support in the new C++ Standard, C++11. This support is coming just in time to allow programmers to take advantage of the greater hardware concurrency becoming available with newer CPUs, as chip manufacturers choose to add more processing power in the form of multiple cores that allow more tasks to be executed concurrently, rather than increasing the execution speed of a single core.

I also showed how simple using the classes and functions from the C++ Standard Library can be, in the examples in [section 1.4](#). In C++, using multiple threads isn't complicated in and of itself; the complexity lies in designing the code so that it behaves as intended.

After the taster examples of [section 1.4](#), it's time for something with a bit more substance. In [chapter 2](#) we'll look at the classes and functions available for managing threads.