BILL WONG | Embedded/Systems/Software Editor
bill.wong@penton.com

Bjarne Stroustrup: C++ Creator Keeps Developing The Language

Mention Bjarne Stroustrup's name to programmers, and they think of C++. That's not surprising since he came up with the object-oriented programming language and continues to be involved in the C++ standard, the latest being C++11.

Stroustrup has master's degrees in mathematics and computer science from Aarhus University in Denmark. His PhD in computer science is from the University of Cambridge. He has also taught and written about C++ with books like *Programming: Principles and Practice Using C++*. So how did he get started designing C++?

"I needed a tool to help me on a project where I needed hardware access, high performance for systems programming tasks, and help with handling complexity. The project was to 'split' a UNIX kernel into parts that could run on a multi-processor or a high-performance local network," Stroustrup said.

"At the time (1979/80), no language could meet all three requirements, so I added Simula-like classes to C. The earliest designs added function argument checking and conversion (what later became C function prototypes), constructors and destructors, and simple inheritance," he said.

Bjarne Stroustrup created C++ and made programming history. He is still working on the latest revisions of the C++ standard.

"My earliest paper on 'C with Classes,' as it was called in the early years, used macros to implement a simple form of generic programming. Later I found that didn't scale and I had to add templates," he added *(see "Interview: Bjarne Stroustrup Discusses C++" at electronicdesign.com)*.

C VERSUS C++

C++ started in 1979 when Bjarne was working on his PhD thesis. "The C++ Programming Language" was published in 1985. In 1998, the C++ standards committee published the first standard for C++ ISO/IEC 14882:1998. Now known as C++98, most C++ compilers support it. C++03 and C++11 then followed. The next revision will be C++14.

C++ shares quite a bit with C, but it was not a proper superset. C11 and C++11 are closer and share most of C's enhancements. Lambda, or anonymous functions, are part of C++11.

C++'S AWFUL TEXTBOOKS—IN STROUSTRUP'S OWN WORDS

When I first was going to teach programming, I looked at the textbooks using C++, and I was furious! There were (and are) books teaching every little obscure detail of C before getting to the far easier to use C++ alternatives and deeming those alternatives "advanced" to scare off all but the most determined student.

Seriously, how could a standard-library vector be as hard to use well as a built-in array? How could using qsort() be simpler than using the more general and efficient sort()? C++ provides better notational support and stronger type checking than C does. This can lead to faster object code.

Other books presented (and present) C++ as a somewhat failed attempt to be a "pure object oriented programming language" and force most every operation into class hierarchies (a la Java) with lots of inheritance and virtual functions. The result is verbose code with unnatural couplings and lots of casting. To add insult to injury, such code also tends to be slow.

As I said, if that's C++, I don't like it either! I responded by writing ***Programming: Principles and Practice Using C++***. It does not assume previous programming experience, though it has been popular with programmers wanting to know what C++ is about.

C++ also supports namespaces for grouping of entities like classes, objects, and functions.

Namespaces can be mixed together without conflicts that could be encountered with C when mixing libraries. The syntax of C++ is based on C. However, it adds quite a bit more including features like operators, operator overloading, templates, and, of course, object classes.

C++ supports static and dynamic polymorphism. It can also handle single-object and multiple-object inheritance. Java provides single inheritance support but allows interfaces per class. Virtual functions enable C++ to provide dynamic polymorphism. Objects can have constructor and destructor functions. C++ memory management includes static, automatic, and dynamic memory allocation. Libraries can support garbage collection.

Templates provide generic function support via parameterized types. Class and function templates are supported. Templates allow classes and functions to be instantiated at compile time. Compilers generate code as necessary when templates are utilized based on the defined types.

Part of the C++ standard is the C++ Standard Library. It provides features that C++ programmers have come to expect including smart pointers. It includes multithreading support, although C++11 includes native thread support within the language.

C++ is significantly more complex than C but there are significant advantages to using it. You don't have to learn all of those features to effectively use C++, though, so if you're looking to learn C or C++, I recommend C++.

C still dominates the embedded programming space, but C++ is overtaking it as more programmers learn C++ and more compilers support it. C++ has had a major impact in the consumer and enterprise space.

BEYOND THE CODE

Stroustrup headed up the Large-scale Programming Research department at AT&T Labs, formerly Bell Labs, until 2002. He is both an IEEE and ACM Fellow. These days he has little time to write a lot of C++ programs because he is teaching. He is now a Distinguished Professor at Texas A&M University, where he holds the College of Engineering Endowed Chair in Computer Science.

He still finds programming to be great fun. His advice to new programmers is to learn to communicate well verbally and in print. Of course, they also should learn programming fundamentals as well as several programming languages.