

Vorgefertigt

Klassen- bibliotheken verwenden

Natürlich ist es eine **schöne Sache**, dass Sie in C++ eigene **Objekte** definieren und später wieder verwenden können. Wesentlich **effizienter** ist es, wenn man **vorgefertigte** Klassen verwenden kann.

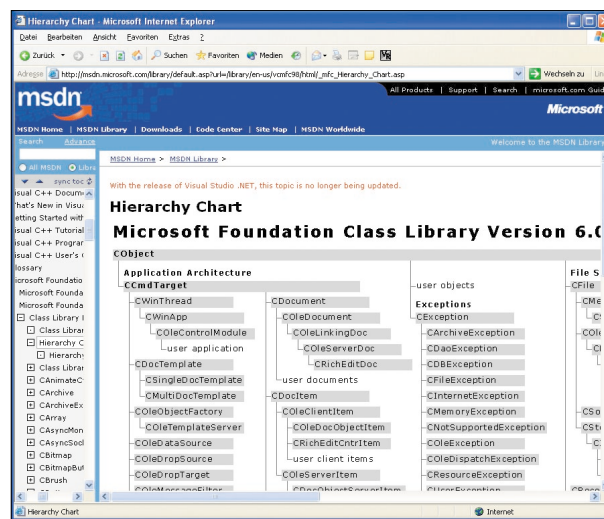
THOMAS WÖLFER

Sie sollten sich nach den ersten Schritten ins Objektorientierte zunächst mit den Bibliotheken beschäftigen, die bei Ihrer VC++-Kopie dabei sind, denn darin befinden sich deutlich mehr hilfreiche Klassen, als man auf den ersten Blick meinen möchte. VC++ kommt von Haus aus mit vier umfangreichen Klassenbibliotheken: der Standard-C++-Bibliothek inklusive der STL (Standard Template Library), der iostreams-Bibliothek, den MFC (Microsoft Foundation Classes) sowie der ATL (Active Template Library). Diese Bibliotheken sind zum Teil sehr umfangreich – aber alle haben ihren sinnvollen Platz im Rahmen der Software-Entwicklung.

■ Die Standard-C++-Bibliothek

Diese Bibliothek setzt sich nicht nur aus Klassen, sondern auch aus Funktionen zusammen. Dabei ist im Besonderen die STL-Bibliothek von Interesse. Diese kapselt zum Beispiel die Funktionalität, die Sie für “Sammlungen“ von Objekten und Daten benötigen. Eine solche Sammlung kann zum Beispiel eine Liste, ein Array oder auch ein Hashtable sein. Mit der STL-Bibliothek stehen Ihnen solche Sammlungen für beliebige Objekttypensicher zur Verfügung – allerdings mit einem kleinen Wermuts-

tropfen: Die Bedienung der STL ist nicht ganz einfach, teilweise sogar recht kryptisch. Der Grund dafür liegt hauptsäch-



DIE MICROSOFT FOUNDATION CLASSES sind bei VC++ immer beige packt. Die Bibliothek dient der Anwendungsentwicklung, enthält aber auch ganz allgemein nützliche Klassen.

lich in der massiven Anwendung von Templates, die das typensichere Verhalten dieser Bibliothek erst möglich machen.

Die iostreams-Library dient der Kapselung von Ein- und Ausgaben. In den Beispielprogrammen im Sonderheft haben Sie immer die C-Funktionen für I/O verwendet – also zum Beispiel *fopen()*,

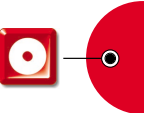
`scanf()`) und so weiter und so fort. Diese Funktionen sind zwar mächtig, schnell und leistungsfähig – doch haben viele davon verschiedene Probleme. So arbeitet beispielsweise `scanf()` mit einer Variable Argument-Liste, in der die Adressen der Ziel-Variablen übergeben werden müssen: Das ist natürlich sehr fehleranfällig und das obendrein noch an einer Stelle, die der Compiler nicht beim Syntaxcheck bemängeln kann. Dies ist nur eines der Probleme, die die `iostreams`-Bibliotheken beseitigen. Dafür ist die bei den `iostreams` zu verwendende Schreibweise teilweise ein wenig umständlich, besonders dann, wenn viele Formatierungsanweisungen bei der Ausgabe von Daten verwendet werden müssen: Die C-Funktionen sind da schon deutlich kompakter.

■ MFC – die Microsoft Foundation Classes

Mit dieser Klassenbibliothek bekommen Sie es zu tun, wenn Sie echte Windows-Anwendungen schreiben möchten. Die MFC und VC++ sind stark miteinander verflochten: So arbeiten eine ganze Menge der in VC++ eingebauten Assistenten direkt mit MFC-Quell-

code zusammen und sind auch speziell fürs Programmieren von Windows-Anwendungen mit MFC gedacht. Dabei enthält die MFC-Klassen für Fenster, Menüs, Icons, Bitmaps und andere Elemente, die beim Programmieren von Windows-Anwendungen anfallen. Darüber hinaus gibt es aber auch Klassen, die mit Windows-Anwendungen per se zunächst nichts zu tun haben. Dazu gehören zum Beispiel die String- und Dateiklassen, die in MFC enthalten sind. Oben-

drein enthalten die MFC-Klassen auch Klassen für die Arbeit mit Sammlungen – also ähnliche Klassen, wie die STL zur Verfügung stellt. Dabei kann man die einzelnen Bibliotheken bzw. der Nutzung natürlich vermischen: Es spricht nichts dagegen innerhalb eines Windows-Programms, das grundsätzlich mit MFC entwickelt wird, auch die io-



streams-Bibliothek oder die STL-Klassen zu verwenden.

Bei der MFC-Bibliothek wurde hauptsächlich darauf Wert gelegt, die Entwicklung von Windows Programmen mit C++ möglichst einfach zu gestalten: Typensicherheit und ein Maximum an Effizienz, wie das bei der STL der Fall ist, sollten Sie bei MFC nicht erwarten: Dafür können Sie mit dieser Bibliothek und den zugehörigen Assistenten ein voll funktionsfähiges Windows-Programm erzeugen, ohne eine einzige Zeile Quellcode zu schreiben.

Nachdem Sie über den Quellcode der MFC verfügen, können Sie auch eigene MFC-Varianten bauen. Dabei sind eine ganze Reihe an Möglichkeiten denkbar – vor allem deshalb, weil viele Spielarten der MFC-Bibliothek von Haus aus vorgesehen sind.

Welche dieser Varianten vom Compiler erzeugt werden, ist von den Präprozessor-Direktiven abhängig, die Sie beim Übersetzen verwenden. Diese sind im Quellcode-Verzeichnis von MFC erläutert.

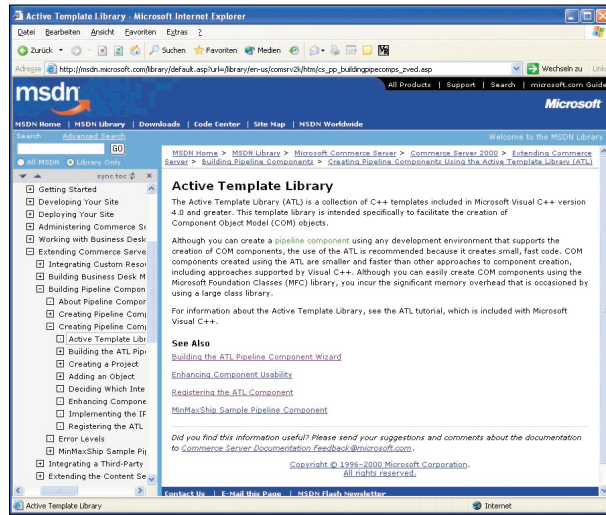
■ ATL

Die ATL (Active Template Library) ist eine "kleine" Library für die Entwicklung von ausführbaren Programmen, wobei sich die ATL hauptsächlich um die Belange von Entwicklern kümmert, die Active-X oder COM-Objekte programmieren wollen. Nachdem Active-X-Objekte auf Webseiten benutzt werden, bedeutet das natürlich auch, dass diese Objekte vom Client heruntergeladen werden müssen.

Daraufhin ist die ATL auch optimiert: Sie können damit möglichst kleine Programme für die Verwendung auf Webseiten programmieren. Das führt aber dazu, dass die ATL-Bibliothek doch sehr unübersichtlich ist. Zwar gibt es innerhalb von VC++ auch Assistenten mit denen Sie ATL-Elemente bearbeiten können, der erzeugte Quellcode ist aber nicht gerade leicht verständlich.

■ Die Klassen: DLLs, Quellcode und dergleichen

Für viele der Klassen liegt Ihnen bei VC++ der komplette Quellcode vor. So finden Sie den Quellcode zu MFC zum Beispiel im MFC-Unterverzeichnis des



AUCH DIE ACTIVE TEMPLATE LIBRARY ist bei VC++ beigelegt. Diese Bibliothek dient hauptsächlich der Programmierung von COM-Objekten.

von Fall zu Fall verschieden, kann aber der Online-Hilfe zu den einzelnen Bibliotheken entnommen werden.

■ Freie Klassenbibliotheken

Klassen kann man für alles Mögliche gebrauchen und implementieren, und genau das haben auch schon eine ganze Menge Leute getan. Im Folgenden werden Ihnen einige interessante Klassenbibliotheken vorgestellt, von denen Sie auch jeweils eine Kopie auf der CD zu diesem Sonderheft finden.

WFC – Win32 Foundation Classes

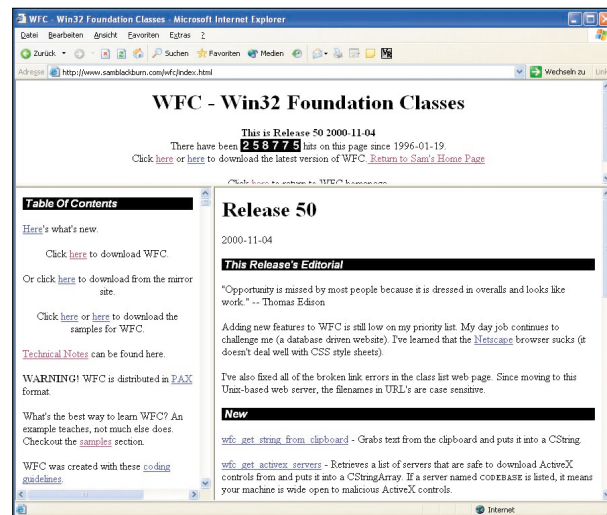
Diese Klassenbibliothek von Sam Blackburn ist eine sehr interessante Erweiterung für die MFC-Bibliothek. Die Microsoft Foundation Classes kümmern sich zwar um eine einfache Entwicklung von Windows-Anwendungen, aber das Programmieren im Allgemeinen und ein spezieller Support für die Win32 API ist in MFC nicht enthalten.

Genau das finden Sie bei WFC: Sam Blackburn hat so hilfreiche Klassen wie den "CDesktop", der die Desktop API

kapselt, und auch den CRandomNumber-Generator zur Erzeugung von Zufallszahlen geschaffen. Die WFC ist eine sehr umfangreiche Klassenbibliothek, die als Erweiterung von MFC eigentlich in keinem Werkzeugkasten eines Windows-Programmierers fehlen sollte. Die vollständige Dokumentation sowie Beispiele und Anmerkungen zur WFC finden Sie im Internet unter <http://www.samblackburn.com/wfc/index.html>.

Die WFC wird

ständig, wenn auch langsam, weiterentwickelt. So fanden in letzter Zeit zum Beispiel auch Klassen für die Behandlung von Internet-Problematiken Einzug in die Bibliothek. Für alle Programmierer, die zwar die Win32 API benutzen möchten, aber lieber ein objektorientiertes Interface statt dem normalen "C"-Interface der API verwenden wollen, sind die WFC eigentlich Pflicht.



SAM BLACKBURNS WFC BIBLIOTHEK kapselt viele Elemente der Win32 API, die in Microsofts MFC-Bibliothek nicht zur Verfügung stehen.

VC++-Installationsverzeichnis. Beigepackt sind auch Makefiles fürs Bauen einer eigenen MFC-Variante – Sie können also den MFC-Quellcode nach belieben untersuchen. Zur Laufzeit liegen die verschiedenen Klassenbibliotheken entweder in Form von DLLs vor, oder aber Sie linksen statisch mit statischen Varianten der Bibliotheken. Welche dieser Möglichkeiten angeboten werden, ist

Boost – Portable C++ Libraries

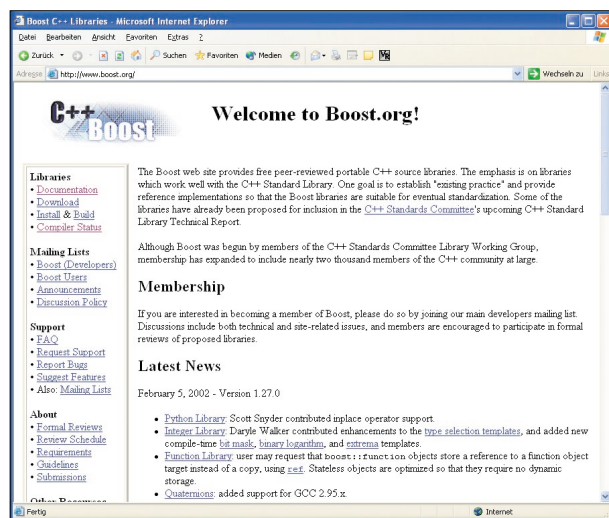
Bei Boost handelt es sich um ein Projekt, das ursprünglich von Mitgliedern der Standard C++ Library Working Group ins Leben gerufen wurde. Ziel der Sache ist es, eine ganze Reihe von Libraries herzustellen, die nicht nur frei verfügbar, sondern auch standardisiert vorliegen sollen. Dabei sind bereits Teile der Boost-Bibliotheken für die Aufnahme in der C++ Standard Library vorgeschlagen worden. Zurzeit stehen circa 30 Bibliotheken zur Verfügung, wobei diese Themen wie die Graphentheorie, mathematische Probleme aber auch Speicherverwaltung und CRC behandeln.

Boost ist besonders deshalb interessant, weil die zur Verfügung stehenden Libraries durchweg von sehr hoher Qualität sind und Probleme lösen, um die sich jeder Programmierer immer wieder kümmern muss. Angesichts der Herkunft der Boost-Projekte ist es doch sehr wahrscheinlich, dass im Laufe der Zeit immer mehr Teile von Boost ohnehin in die Standard-C++-Bibliothek einfließen – es kann also ganz sicher nicht schaden, sich schon heute mit den dadurch gebotenen Möglichkeiten auseinander zu setzen. Die Boost Libraries können von www.boost.org heruntergeladen werden.

Blitz – numerische Klassen

Bei der Blitz Library handelt es sich um eine Klassenbibliothek, die sich ausschließlich um numerische Probleme in C++ kümmert. Ziel von Blitz ist es, C++-Klassen zur Verfügung zu stellen, mit denen mathematische Probleme, ähnlich effizient wie in Fortran, gelöst werden können. Blitz steht unter der GPL zur Verfügung und implementiert momentan hauptsächlich mehrdimensionale Arrays. Klassen für Matrizen und die Erzeugung von Zufallszahlen sind in Arbeit. Allerdings gibt es von Blitz keine Variante, die für den VC++-Compiler von Microsoft gedacht ist. Der Quellcode von Blitz ist fürs Studi-

um trotzdem auf der Heft-CD enthalten. Der Grund dafür ist der, dass der von Blitz gewählte Ansatz zwar ein sehr "wilder", aber nichts desto trotz faszinierender ist. Im Zuge der Formelauswertung verwendet Blitz den Compiler dazu, verschachtelte Templates zu generieren, die ihrerseits die einzelnen Rechenschritte bei der Berechnung der Formeln in Schleifen auflösen, so dass keine temporären Objekte benötigt werden. Das führt dazu, dass am Ende der Auswertung einer Formel, aber bevor der Operator "=" zum Zuge kommt, eine verschachtelte Struktur aus Templates entstanden ist, die die komplette Gleichung mehr oder minder so abbildet, wie man sie auch von Hand be-



DAS BOOST-Projekt wurde ursprünglich von Mitgliedern der Standard C++ Library Working Group ins Leben gerufen.

rechnen würde. Der Operator "=" emittiert dann den Code, der die Evaluierung der zuvor generierten Struktur anstößt. Wer mehr über Templates lernen will, findet bei Blitz einen sehr kreativen, aber auch sehr komplexen Ansatz zur Nutzung dieses C++-Sprachfeatures. Die Homepage der Library findet sich unter <http://www.oonumerics.org/blitz/>.

AV3D – Spiele programmieren

Die AV3D-Library von Galacticasoftware ist eine Bibliothek, die sich vor allem an Spieleprogrammierer wendet. Die Bibliothek unterstützt OpenGL sowie SFX, Sounds und Musikwiedergabe. Zusätzlich stehen Klassen für die Berechnungen von 3D-Operationen zur Verfügung. Dabei ist die Library sogar mehr oder weniger portabel und kann sowohl für Linux als auch für Windows eingesetzt werden. Dabei wird die Win-

dows API und X Windows unterstützt. Unter Windows werden dabei unter anderem auch die DirectX-Funktionen verwendet, die natürlich auch nativ benutzt werden können: Der Vorteil von AV3D ist dabei der, dass die Library diese Funktionalität so weit abstrahiert, dass der gleiche Quellcode, der für ein Win32-Programm verwendet werden kann, auch für ein X-Windows-Programm genutzt werden kann. Die Homepage der AV3D Library finden Sie unter www.galacticasoftware.com/products/av/.

Common C++ – Systemdienste

Die Common C++ Library bietet einen Satz an Klassen, die allgemeine Systemdienste kapseln. Dazu zählt zum Beispiel der Support für Threading, Dateizugriffe, Hintergrundprozesse sowie Sockets. Auch die Common C++ Library finden Sie auf der CD zum Sonderheft. Die Homepage von Common C++ finden Sie unter <http://cplus-plus.sourceforge.net/> dort befindet sich auch die recht umfangreiche Dokumentation.

Verschlüsselung mit Crypto++

Die Crypto++-Bibliothek ist eine freie Klassenbibliothek, die kryptographische Hilfsmittel implementiert. Dazu zählt zum Beispiel eine Klasse, die verschiedene public-Key-Verschlüsselungsmethoden implementiert. Unterstützt werden dabei zum Beispiel RSA, DAS und ElGamal. Außerdem finden Sie in Crypto++ unter anderem MD2, MD4 und MD5 Algorithmen in Klassen verpackt, Komprimierungs- und Dekomprimierungsmethoden (GZIP kompatibel) sowie auch eine Klasse, die Primzahlen ermittelt und verifiziert. Zusätzlich enthält Crypto++-Support für eine ganze Reihe an Verschlüsselungsalgorithmen: Dazu gehören IDEA, DES, Triple-DES, RC5, Blowfish und viele andere mehr. Jeder, der sich für Kryptographie interessiert, wird an Crypto++ seine helle Freude haben, die dazugehörige Homepage findet sich unter <http://www.eskimo.com/~weidai/cryptlib.html>.

GNU Nana – Fehler schneller finden

GNU Nana nimmt sich der Problematik von Fehlerprotokollen an. Die Library versorgt Sie mit Klassen und Funktionen, um Bedingungen zu prüfen und Informationen mitzuprotokollieren. Dokumentationen zu weiteren Informationen zu Gnu Nana finden sich



auf der dazugehörigen Homepage unter <http://www.cs.ntu.edu.au/home-pages/pjm/nana-home/>.

LFC – die freie MFC-Variante

Die LFC Library ist eine allgemeine C++-Klassenbibliothek sowie ein Framework für die Anwendungsprogrammierung. Die Bibliothek enthält Klassen für allgemeine Datenstrukturen und Algorithmen in Form von STL-Erweiterungen. Außerdem kapselt die Bibliothek Betriebssystemdienste, wie Sockets, Dateien und Threads, enthält aber auch GUI-Funktionalitäten. Außerdem enthält die Bibliothek Klassen für den Zugriff auf Datenbanken, Multimedia-Klassen und Klassen, die die Internet-Protokolle abstrahieren. Die Homepage der LFC befindet sich bei SourceForge unter <http://lfc.sourceforge.net/>, aber die aktuelle Version der Library finden Sie auch auf der Heft-CD zum Sonderheft. Allerdings ist die LFC nicht speziell für VC++ entworfen worden – stattdessen steht der bevorzugte Compiler unter Windows Borlands C++ 5.5

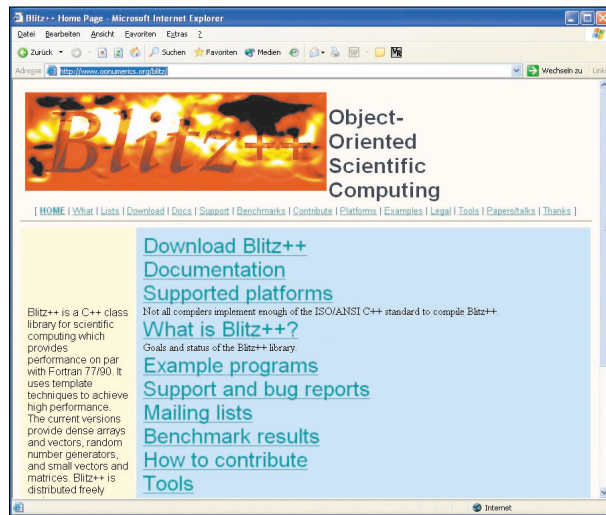
MSS – das Memory Supervision System

Die MSS-Bibliothek hat eine zentrale Aufgabe: Es kümmert sich darum, dass Fehler bei der Speicherverwaltung vermieden werden. Beim MSS handelt es sich um eine Library, die per GPL zur Verfügung gestellt wird und die die folgenden Probleme bei der dynamischen Speicherverwaltung – bzw., diese immer wieder auftretenden Fehler – zu vermeiden hilft:

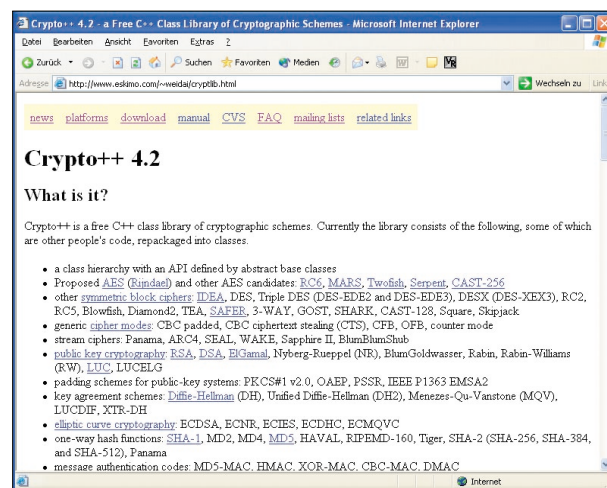
- Speicherlecks,
- Benutzung von nicht initialisiertem Speicher,
- Vermeidung von 0-Byte-großen Speicheranforderungen,
- Zugriff auf Speicher, der zuvor gar nicht angefordert wurde,
- Vermeidung der mehrfachen Freigabe von Speicherblöcken,
- Aufspüren von Speicheranforderungen, die nicht funktioniert haben,

- Vermeidung von Zeigern, die auf nicht initialisierten Speicher zeigen.

Außerdem führt die MSS eine ausführliche Statistik über die Speicherverwendung mit. So erhält man mit MSS die folgenden Informationen:



DIE BLITZ LIBRARY will möglichst effiziente Berechnungen mit C++ ermöglichen. Leider ist die Bibliothek aber mit VC++ nicht kompatibel.



VERSCHLÜSSELUNG MIT C++: Das hat sich die Crypto Library zum Ziel gemacht.

- Die Gesamtsumme des verwendeten Speichers,
- die Anzahl der Speicheranforderungen (und Freigabe) seit dem Start des Programms,
- die Anzahl der angeforderten Blöcke,
- eine Liste der Speicherblöcke, zusammen mit den Dateien und Funktionen, in denen die Blöcke angefordert wurden. Im Großen und Ganzen entspricht das in etwa dem, was man bei MFC mit den CMemory*-Klassen zur Verfügung

hat – nur ist die MSS auch in Programmen verwendbar, die nicht die MFC-Bibliothek benutzen. (Dazu ist allerdings anzumerken, dass auch die C-Laufzeitbibliothek des VC++ 6.0 einen ähnlichen Mechanismus enthält – nur eben nicht auf Basis von Klassen, sondern in Form ganze normaler “C“-Funktionen.) Die Homepage der MSS finden Sie unter <http://hem.pas-sagen.se/blitzar/mss/>.

CppIma – Bildverarbeitung mit C++

Die CppIma-Bibliothek bietet Klassen für die Bildbearbeitung. Bilder können damit erzeugt, weiterverarbeitet und konvertiert werden. Die komplette Dokumentation zu CppIma steht online zur Verfügung. Die Homepage dieser Library findet sich unter <http://www.ph.tn.tudelft.nl/~klamer/cppima.html>.

Soviel zu einer kurzen Übersicht über einige der interessantesten C++-Klassenbibliotheken. Die vorgestellten Bibliotheken sind aber längst nicht alle, die auch verfügbar sind. Eine einfache Suche mit einem normalen Search-Engine im Internet – zum Beispiel Google – bringt schnell eine ganze Reihe weiterer Klassenbibliotheken zum Vorschein: Praktisch für jeden Anwendungsfall sind fertige Klassen im Internet zu haben.

Gar nicht erwähnt wurden die “kommerziellen” Klassenbibliotheken, denn auch davon gibt es eine ganze Reihe: Viele davon stellen entweder eigene Frameworks zur Anwendungsentwicklung zur Verfügung oder sind zumindest als Erweiterung für solche

Frameworks, wie zum Beispiel MFC, zu verstehen. Wieder andere Bibliotheken sollen die Betriebssystemabhängigen API-Aufrufe abstrahieren und somit Plattformen für die Entwicklung von portablem Code darstellen: Entwickelt man Anwendungen mit solchen Bibliotheken, kann die resultierende Anwendung meist auf mehreren Plattformen – also zum Beispiel Win32, Unix mit X Windows und OS/2 oder Solaris, betrieben werden.

UR