**Fazit**

Ich wurde gebeten ein Fazit über die Bachelorarbeit abzugeben und zu beschreiben was ich alles im Laufe der Zeit dabei gelernt habe.  
Im Grunde hab ich schon eine ganze Menge aufgezählt. Die Scintilla Nachrichten, das Arbeiten mit CWEB, das Bauen einer Win32 Anwendung, …

Aber zuerst eine kleine Vorgeschichte was ich vorher getan und gewusst habe, um zu sehen was ich tatsächlich alles gelernt habe.

**Was habe ich vorher gekonnt:**

Durch das Studium habe ich das Programmieren mit Sprachen wie Java, C/C++, Scala, LISP, html, JavaScript und MMIX gelernt. Dabei gingen die Programme lediglich bei Java über die normalen einfachen Anwendungen hinaus.  
Dennoch waren sämtliche Programme immer Konsolenanwendungen und es war nie verlangt eine Oberfläche zu programmieren.  
Durch meine Werkstudententätigkeit und das Praktikum bei einer Firma für Bankensoftware in Aschheim habe ich Erfahrung in vielen verschiedenen Bereichen der Anwendungsentwicklung und Systemintegration gesammelt.  
Ich habe Java Anwendungen mit Oberfläche, Server auf Java, Android und IOS Apps und später auch Win8 Apps entwickelt.

Erfahrung mit Win32 Anwendungen hatte ich daher nicht. Und ich wusste auch nur wie man Oberflächen mit Java, Objective C, JavaScript, XAML und C# baut.  
Die Entwicklung einer Win32 Anwendung mit Oberfläche war deswegen für mich totales Neuland.

Schluss mit der Erzählstunde.

**Was habe ich gelernt:**

Zu allererst das Programmieren einer Win32 Anwendung.  
Zu meinem Erstaunen gibt es dort sehr starke Abweichung zu z.B. Java oder C#. So ist der Messageloop etwas komplett Neues für mich gewesen. Da in Java sowas durch die VM und die Listener geregelt wird, und in C# einfach eine Methode mitgeben wird.  
Ich war bisher nur Sprachen gewohnt die mir die meiste Arbeit abgenommen haben. In Win32 aus C++ ist dies meistens nicht so. Der Vorteil dadurch ist natürlich die volle Konfigurierbarkeit.  
Die Ressourcen haben mich überrascht da ich es bisher nur von Win8 Apps gewohnt war dass die Oberfläche logik-unabhängig in z.B. XAML geschrieben werden konnte. Ehrlich gesagt war ich von einem Java-Swing ähnlichen Ansatz ausgegangen.

Im Bereich von Scintilla war ich einfach erstaunt was man alles mit der Komponente anstellen kann. Die Fülle an Funktionen war einfach überwältigend. Als ich mich durch die Doku von Scintilla gelesen habe fand ich lauter Funktionen die ich gerne im Programm gehabt hätte, war dann aber ganz enttäuscht als ich festgestellt hab dass für das Meiste die Zeit nicht ausreichen würde. Aus diesem Grunde werde ich weiter am Programm arbeiten und neue Teile in dieses integrieren.

In Bezug auf MMIX war ich über die CWEB Technologie sehr erfreut. Ich finde es sehr praktisch einen Hauptcode zu haben den man, ohne Zwischenspeichern oder Versionskontrolle, zu seiner Zufriedenheit ändern kann. Ich wünschte es gäbe diese Technologie auch für andere Sprachen. Besonders wenn man verschiedene Versionen ein und desselben Programms bauen muss ist CWEB sehr nützlich. Ohne das muss man mit Vererbung oder Kopieren arbeiten.

**Meine Probleme mit dem Code:**

Während der Entwicklung bin ich auf ein paar Probleme gestoßen die mich meistens sehr viel Zeit gekostet haben.

So mussten die C Dateien des MMIX Codes im Visual Studio speziell als C Datei kompiliert werden um Linker und Debugger Fehler zu vermeiden. Dazu musste in Visual Studio in den Eigenschaften des Compilers ein Flag gesetzt werden.

Variablendeklarationen in C Methoden müssen am Anfang dieser stehen da sonst der Visual Studio Compiler meckert. Diesen Stil war ich von den bisherigen Compilern nicht gewohnt.

Die ListBox in der Anwendung als Ressource zu schreiben hat sich als wenig effektiv erwiesen. Zumal es sowieso total Overkill ist, eine Ressource wegen einer einzelnen ListBox zu bauen, hat dies Probleme mit dem Weiterleiten des Doppelklicks gebracht. Der Doppelklick wird nur an das Parent ListBox gesandt. Da das Parent nicht das Hauptfenster ist, sondern die Ressource, funktioniert das nicht so wie gewollt. Aus diesem Grund ist die Fehlerliste mit CreateWindow erschaffen.

Das Ermitteln der Breite eines Strings durch SCI\_TEXTWIDTH gibt bei z.B. dem String „999“ nicht die volle Breite aus. Die Breite die dabei ausgegeben wird ist komischerweise etwas zu klein für „999“. Dies kann man beheben wenn man dem Text zwei „\_“ mitgibt, also zwei weitere Zeichen.

Ein weiteres Problem trat auf beim Setzen und Abfragen des versteckten Werts in der ListBox. Ich hatte alles richtig gebaut nur wollte es einfach nicht funktionieren. Manchmal liegt sowas daran dass man einmal einen ordentlichen Clean ausführen muss. Daran lag es aber nicht. Was geholfen hat war ein Cut-Compile-Paste. Also entfernen des Programmteils, kompilieren, einfügen des vorher entfernten Programmteils und wieder kompilieren. Ich rätsele noch heute daran warum das nicht funktionieren wollte.

**Anmerkungen und die Zukunft:**

Es war überraschend wie stark einem Rückmeldungen von Personen helfen können oder einem Ideen bringen.

So sollte ich im Rahmen dieser Bachelorarbeit auch ein Referat halten. Dies geschah in einer Vorlesung in der mit MMIX programmiert wurde.  
Es war erstaunlich wie viele Rückmeldungen ich erhalten hatte und wie viele Leute Ideen für die Zukunft des Programms hatten.

Da mir die Arbeit an dem Programm und mit Scintilla gefallen hat werde ich gerne daran weiter arbeiten um das Programm mit mehr nützlichen Funktionen auszustatten.  
Dabei werde ich die Ideen der Teilnehmer der Lehrveranstaltung mit einbeziehen so gut es geht.

**Danksagung:**

Ich möchte zwei Personen, die mir sehr geholfen haben, Danke sagen.

Als aller erstes möchte ich meinem Professor danken, der mir immer mit Hilfe zur Seite stand. Ich weiß nicht ob ich es ohne seine Unterstützung geschafft hätte. Ich weiß dass es nicht besonders ist mit mir zu arbeiten aber ich möchte ihm dafür danken dass er mich ausgehalten und überlebt hat.

Christoph Hösler (Uni Tübingen) ist zu danken dass er den Lexer geschrieben hat den ich in MMIX-Edit verwende. Er hat mir damit einiges an Arbeit abgenommen.