combit GmbH
Untere Laube 30
78462 Konstanz

Whitepaper

Produkt: List & Label

Hilfe zur Selbsthilfe

Inhalt

Einleitung

Dieses Whitepaper soll Ihnen dabei helfen, Fehler und Probleme selbständig ausfindig zu machen und zu beheben. Es gibt Ihnen Tipps wie Sie sich am besten bei einem auftretenden Problem verhalten, bei der Suche vorgehen und in welcher Reihenfolge was zu tun ist. Wenn Sie dennoch den Support kontaktieren müssen, hilft es Ihnen die ersten Vorbereitungen zu treffen, um dem Support so viele Informationen wie möglich zu liefern.

Ein großes Augenmerk gilt dem sogenannten Debuggen. Debuggen bedeutet in unserem Fall Ihr Programm bei der Ausführung live auf Fehler zu untersuchen. Dabei wird Ihnen das Programm "Debwin3" hilfreich zur Seite stehen. Sie finden es im List & Label Installationsverzeichnis im Unterordner "Tools". Dieses Programm protokolliert alle Vorgänge in einer einzigen Text-Datei, der sogenannten Log-Datei. Im Laufe dieses Whitepapers erhalten Sie einen Einblick über den Aufbau und den Inhalt der Log-Datei. Ebenso wie Sie Fehler in der selbigen schnellstmöglich ausfindig machen können.

Vorgehensweise bei Fehlersuche

Wenn ein Problem auftritt empfiehlt es sich eine bestimmte Reihenfolge bei der Fehlersuche einzuhalten. Nachfolgend eine Auflistung der vorzunehmenden Schritte, eine Erklärung der einzelnen Schritte folgt im Laufe des Whitepapers.

1. Service Pack-Aktualität prüfen
2. Service Pack ReadMe lesen
3. Online Knowledgebase durchsuchen
4. Log-Datei erstellen
5. Log-Datei analysieren
6. Bei Abstürzen, Dump-Datei erstellen
7. Support kontaktieren

Service Pack-Aktualität prüfen

Bitte überprüfen Sie als erstes, ob Sie die aktuellste Version von List & Label nutzen. Viele aufgetretene Probleme, die uns die Benutzer/Entwickler melden, werden nach einer erfolgreichen Lokalisierung und Behebung in den Service Packs gefixt. Daher empfiehlt es sich List & Label immer auf dem neuesten Stand zu halten.
Die Version des genutzten Service Packs lässt sich auf zwei Arten feststellen. Zum einen über den Designer und zum anderen über die Log-Datei. Auf Letzteres wird im weiteren Verlauf eingegangen.
Führen Sie im geöffneten Designer die Tastenkombination <STRG + UMSCHALT + F12> (CTRL + SHIFT + F12) aus. In dem sich öffnenden Dialog können Sie Informationen über List & Label und Ihr System entnehmen. Entscheidend hierbei ist die Versionsnummer. Unter "Version" sehen Sie Ihre momentane List & Label Version mit Unterversion. Außerdem kann man erkennen, welche Module geladen werden.

Eine Übersicht der aktuellen Service Packs zu Ihren registrierten Produkten finden Sie auf unserer Homepage unter folgendem Link:
http://www.combit.net/servicepacks.aspx
Zu jedem Service Pack auf der Homepage finden Sie eine ReadMe-Datei in Form eines PDF. In diesem ReadMe sind alle Änderungen im Vergleich zur Vorgängerversion, nach betroffenen Modulen sortiert, aufgelistet.

Service Pack ReadMe lesen

ReadMes werden grundsätzlich in englischer Sprache verfasst. In ihnen sind neue Features, Änderungen und behobene Probleme im Vergleich zu der vorherigen Version dokumentiert.

Schauen Sie sich also zuerst die ReadMe's genau an. Eventuell ist Ihr Problem durch das Herunterladen und Installieren eines Service Packs bereits schnell und einfach gelöst.

Online Knowledgebase durchsuchen

Wenn Sie bereits erahnen oder gar wissen was die Ursache Ihres Problems ist, können Sie auch unsere kostenlose Online Knowledgebase zur Hilfe nehmen. Hier finden Sie Artikel mit Tipps oder Lösungen zu bekannten und häufig auftretenden Problemen. Artikel wie "List & Label einbinden" oder "Barcodes: Probleme bei der Darstellung" sind nur ein Bruchteil von dem was die Knowledgebase beinhaltet. Sie gelangen entweder über das Suchformular oder die Kategorieauswahl schnell und ohne Umwege zu den Artikeln. Zur Knowledgebase gelangen Sie über folgenden Link:

http://www.combit.net/knowledgebase

Debugging von .NET Anwendungen

Probleme die auf dem Entwicklerrechner auftreten können meist leicht gefunden werden – hier kann direkt mit den üblichen Features der Entwicklungsumgebung gearbeitet werden und ein Problem so recht schnell eingegrenzt werden. Der erste Schritt besteht darin, eventuell auftretende Exceptions abzufangen und deren Ursache zu überprüfen (vgl. Abschnitt "Fehlerhandling mit Exceptions" in der Programmierer-Referenz).

Als Entwicklungskomponente wird List & Label aber natürlich unter einer Vielzahl verschiedener Konstellationen bei den Endanwendern ausgeführt. Um Probleme dort möglichst einfach zu finden, steht ein eigenes Debug-Tool zur Verfügung, das bei selten oder nur auf bestimmten Systemen auftretenden Problemen eine Protokollierungsfunktion bietet, mit deren Hilfe Probleme auch auf Systemen ohne Debugger untersucht werden können.

Natürlich kann die Logging-Funktion auch auf dem Entwicklerrechner genutzt werden und bietet auch dort die Möglichkeit, sämtliche Aufrufe und Rückgabewerte schnell auf einen Blick zu prüfen.

Protokolldatei anfertigen
Tritt ein Problem nur auf einem Kundensystem auf, sollte auf diesem zunächst eine Protokolldatei erstellt werden. Hierzu dient das Tool Debwin, welches im "Verschiedenes"-Verzeichnis der List & Label-Installation installiert wird.

Debwin muss vor der Applikation gestartet werden. Wenn anschließend die Applikation gestartet wird, werden sämtliche Aufrufe an die Komponente mit ihren Rückgabewerten sowie einige Zusatzinformationen zu Modulversionen, Betriebssystem etc. protokolliert.

Jede unter .NET geworfene Exception entspricht im Protokoll einem negativen Rückgabewert einer Funktion. Im Protokoll finden sich meist weitere hilfreiche Informationen.

Soll die Anwendung ohne Hilfe von Debwin Debug-Protokolle erstellen, kann dies z.B. über die Konfigurationsdatei der Anwendung erreicht werden. Eine Protokollierung kann darin wie folgt erzwungen werden:

```
<configuration>
 <appSettings>
  <add key="ListLabel DebugLogFilePath" value="C:\Users\Public\debug.log" />
  <add key="ListLabel EnableDebug" value="1" />
 </appSettings>
</configuration>
```

Benutzerdefiniertes Logging & Protokollierung in Webanwendungen
Für normale Desktopanwendungen sind die Aufzeichnung der Logausgaben von List & Label mit Debwin und das integrierte Schreiben einer Protokolldatei einfache und praktische Lösungen.

Bei Webanwendungen, Windows-Diensten und Mehrbenutzersystemen geraten diese Vorgehensweisen aber an ihre Grenzen: Debwin und die integrierte Protokolldatei verwenden genau eine Logdatei je Prozess, sodass Logausgaben mehrerer, parallellaufender Jobs nicht getrennt werden können.

In diesen Situationen empfiehlt sich die Nutzung eines eigenen Logging-Mechanismus oder eines Logging-Frameworks wie NLog oder log4net. Sie können dazu eine Klasse erstellen, die von LoggerBase abgeleitet ist (oder selbst das ILlLogger-Interface implementiert) und ein Objekt dieser Klasse an den Konstruktor des

ListLabel-Objekts übergeben. Anschließend werden alle Logausgaben von List & Label an dieses Objekt weitergeleitet. Die Logausgaben können anhand verschiedener Prioritäten (Debug-Ausgabe, Information, Warnung und Fehler) und Kategorien (z.B. Datenprovider, .NET-Komponente, Druckerinformation, etc.) gefiltert werden.

Beispiel: Logausgaben an NLog weiterleiten:

```
ILogger   nlogLogger = NLog.LogManager.GetLogger("MyApp.Reporting");
ILlLogger llLogger = new ListLabel2NLogAdapter(nlogLogger);
ListLabel LL = new ListLabel(llLogger);
```

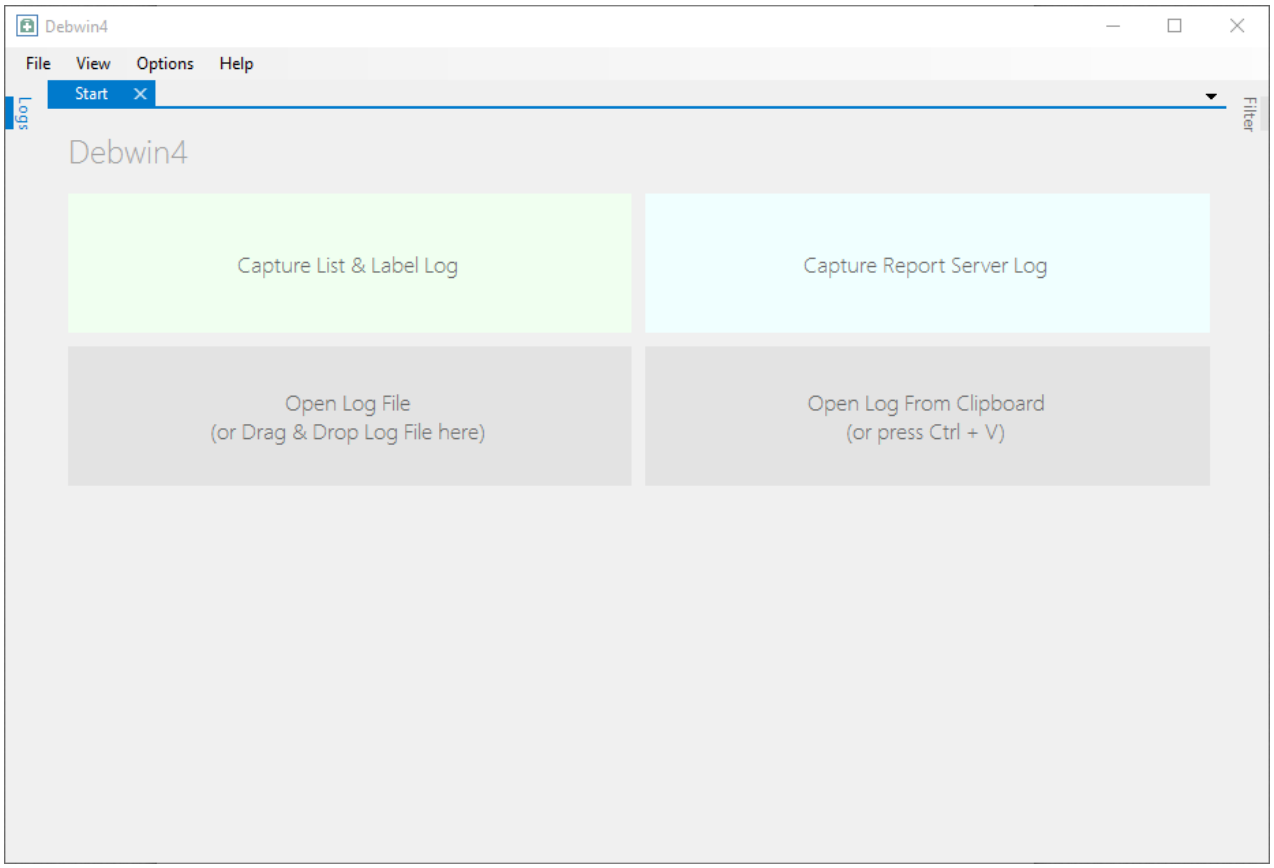
Bitte beachten Sie:

- Die Eigenschaften "Debug" und "DebugLogFilePath" der ListLabel-Klasse werden ignoriert, wenn Sie einen eigenen Logger übergeben.
- Beschränken Sie die Logausgaben in Ihrer ILlLogger-Implementierung mit Hilfe der WantOutput()-Funktion auf ein Minimum, um die Performance nicht zu stark zu beeinträchtigen.
- Die meisten der mitgelieferten Datenprovider unterstützen (optional) ebenfalls ein externes Logger-Objekt. Diese Datenprovider implementieren die Schnittstelle ISupportsLogger und verfügen über eine SetLogger()-Funktion. Falls ein Datenprovider kein eigenes Logger-Objekt hat, wird das der ListLabel-Instanz übernommen.

Tipp für NLog: Oft werden viele Logmeldungen in kurzer Zeit ausgegeben. Verwenden Sie das AsyncWrapper-Target für eine asynchrone Verarbeitung der Logausgaben, sodass List & Label nicht auf diese warten muss.

Log-Datei mit Debwin4 erstellen

Mit der Version 22 von List & Label wurde das Debugging mit Debwin komplett überarbeitet und deutlich vereinfacht.



Der Debug-Modus muss nicht mehr über LlSetDebug() eingeschaltet werden. Um alle Ausgaben zu erhalten, muss Debwin vor der zu debuggenden Applikation gestartet werden. Sobald Ihre Applikation gestartet ist, werden dann die Debug-Ausgaben von Debwin verwertet und ausgegeben.

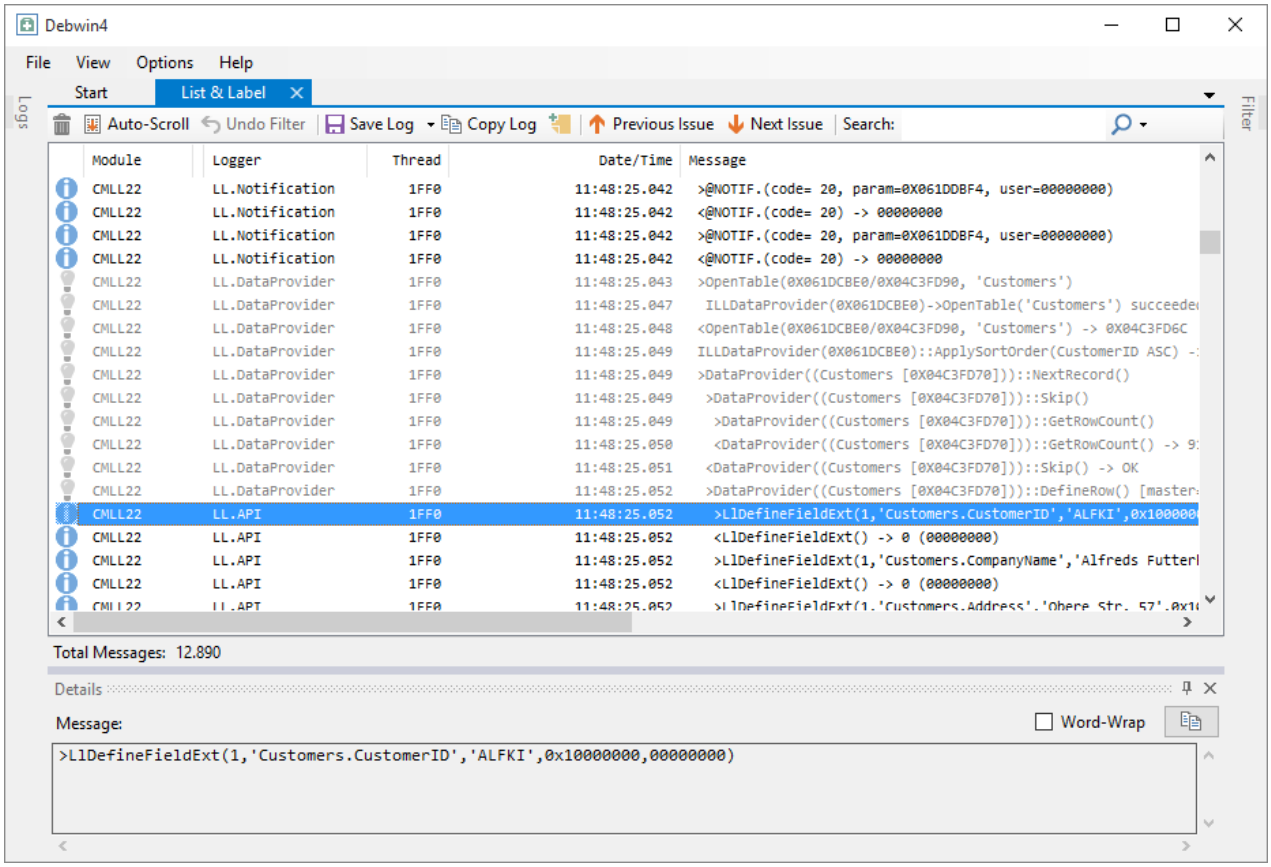
Folgende Möglichkeiten stehen Ihnen zur Verfügung:

Capture List & Label Log	Log-Datei für List & Label erstellen
Capture Report Server Log	Log-Datei für den combit Report Server erstellen
Open Log File (or Drag & Drop Log File here)	Öffnen Sie eine vorhandene Log-Datei
Open Log From Clipboard (or press Ctrl + V)	Öffnen Sie in die Zwischenablage kopierte Log-Ausgaben

Bei der Verwendung von Debwin4 muss die bekannte Vorgehensweise von Debwin3 eingehalten werden:

1. Debwin4 starten.
2. Capture List & Label Log wählen.
3. Die Applikation starten und das Fehlverhalten erneut provozieren.

Neben den Fehlercodes erhalten Sie weitere Informationen, so dass Sie häufig auf diese Weise die Ursache für ein unerwünschtes Verhalten finden können.



Man erkennt das aufgerufene Modul (CMLL22), den Logger, die Thread-ID, Timing-Informationen und die aufgerufene Funktion mit Parametern sowie – in der Folgezeile – den Rückgabewert der Funktion:

CMLL22 LL.API 1FF0 11:48:25.052 >LlDefineFieldExt(1, 'Customers.CustomerID', 'ALFKI', 0x10000000, 0000000)

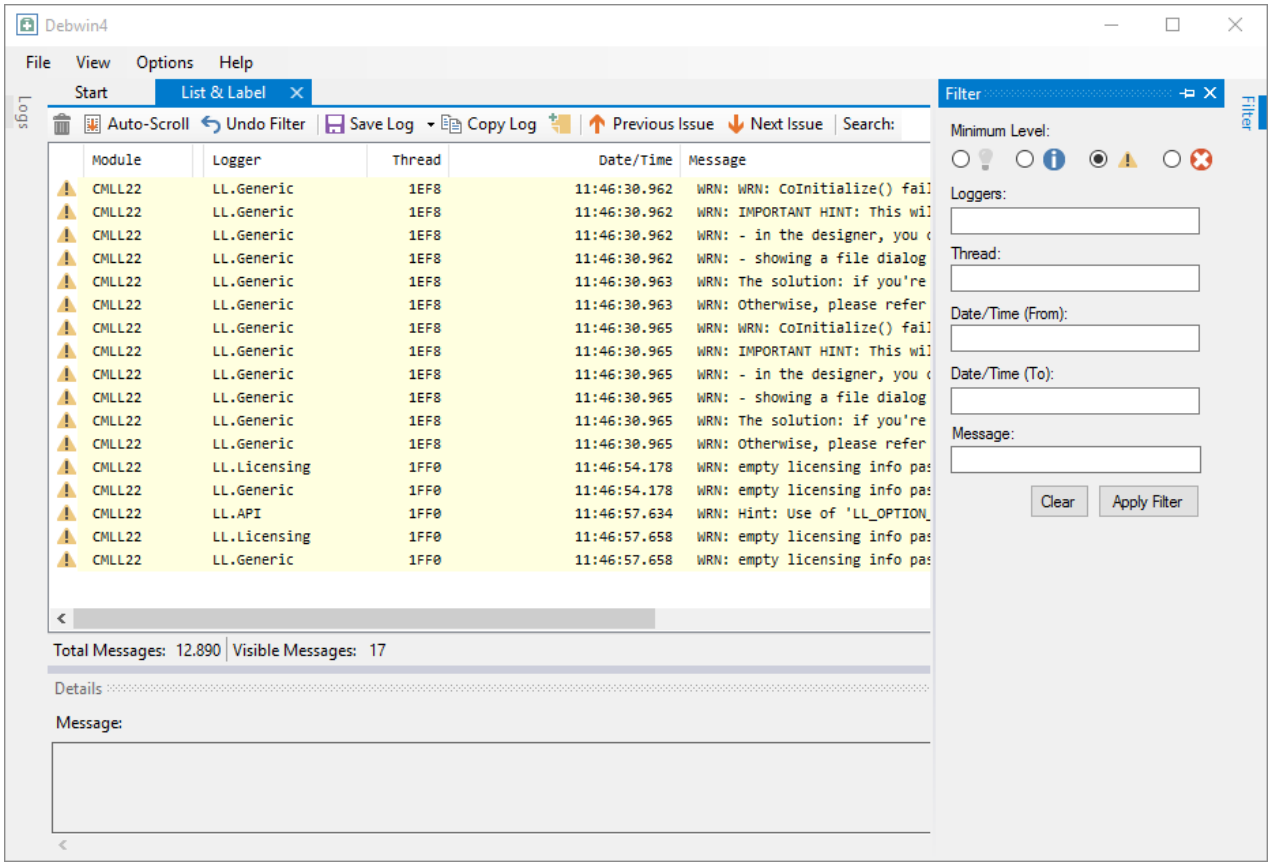
Während des Loggings stehen folgende Funktionen in Debwin4 zur Verfügung:

Clear All Messages	Hiermit werden alle Ausgaben in der aktuellen Ansicht gelöscht.
Auto-Scroll	Ist diese Option aktiviert, dann liegt der Fokus des Ausgabe-Fensters auf den letzten und somit aktuellen Debug-Ausgaben.
Undo Filter	Eventuell gesetzte Filter werden zurückgesetzt.
Save Log	Speichert die Log-Datei im neuen log4-Format.
Save and Open in Editor	Speichert die Log-Datei im log-Format und öffnet diese im Editor.
Copy Log	Kopiert die Debug-Ausgaben in die Zwischenablage.
Add Custom Message	Fügt eine Notiz/Nachricht in die Log-Datei ein.
Previous Issue	Zeigt den vorherigen Fehler in der Log-Datei an.
Next Issue	Zeigt den nächsten Fehler in der Log-Datei an.
Search	Suche in der Log-Datei.

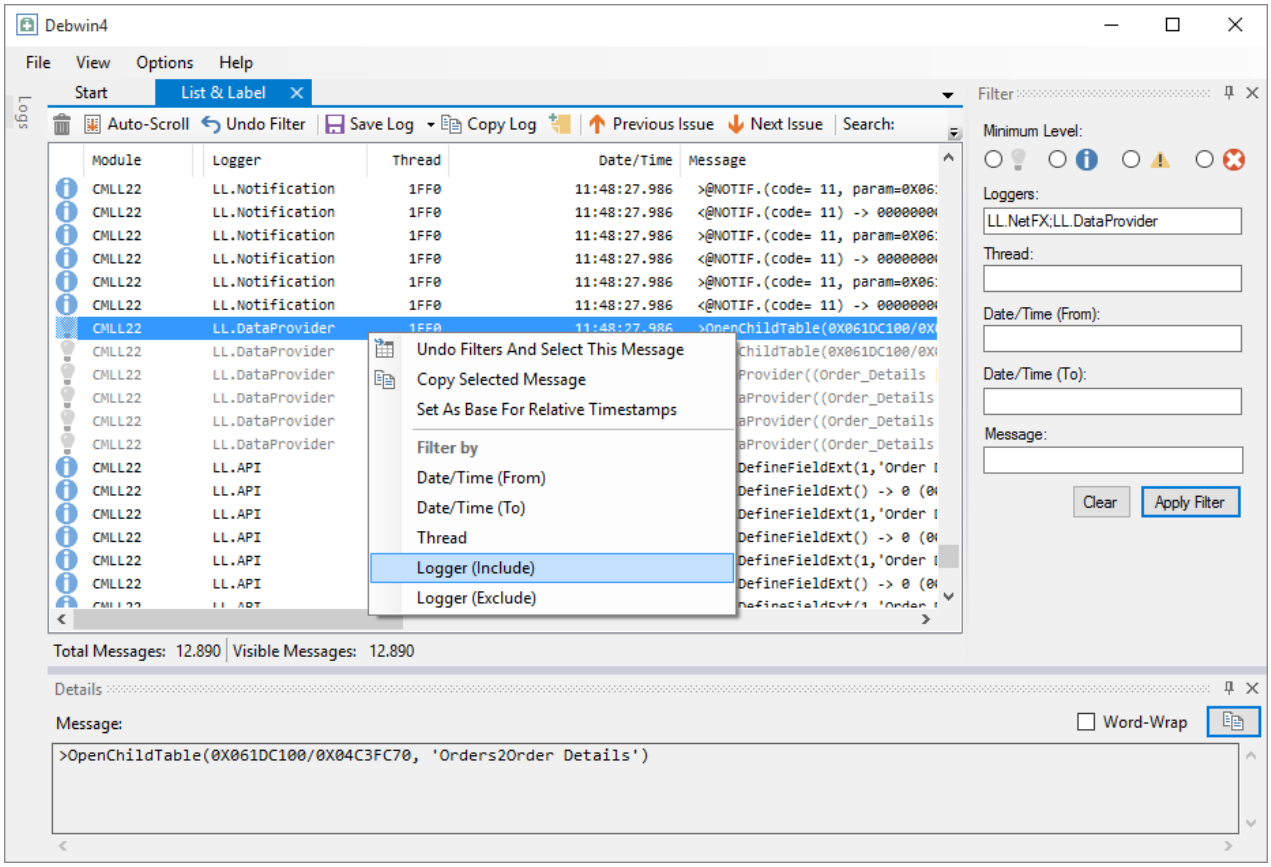
Log-Datei filtern

Über das Filter-Fenster haben Sie die Möglichkeit die Log-Datei schnell und unkompliziert nach bestimmten Informationen zu durchsuchen bzw. die Debug-Ausgaben einzugrenzen.

Minimum Level	Bestimmt die Priorität der Ausgabe (Debug-Ausgabe, Information, Warnung und Fehler).
Loggers	Eingrenzung nach Logger (LL.API, LL.DataProvider, LL.Export, LL.Generic, LL.Licensing, LL.NetFX, LL.Notification).
Thread	Eingrenzung nach einem bestimmten Thread.
Date/Time (From)	Debug-Ausgaben ab einem bestimmten Datum/Zeit.
Date/Time (To)	Debug-Ausgaben bis einem bestimmten Datum/Zeit.
Message	Bestimmte Debug-Ausgabe.



Die Filter erlauben die Verwendung mehrerer semikolon-separierter Werte. Zusätzlich lassen sich die Filterkriterien, z.B. nach einem bestimmten Thread oder verschiedene Logger bequem über das Kontextmenü des Ausgabefensters selbst zusammenstellen. Klicken Sie einfach auf die jeweilige Zeile und wählen Sie den einzuschließenden Thread oder Logger aus:



Tipp: Die Suche (Search) und der Filter "Message" erlauben die Verwendung von Regular Expressions.

Erstellen einer Log-Datei bei Server/Webapplikationen

Zur Erstellung von Log-Dateien für Server/Webapplikationen ist es notwendig Debwin4 als Administrator auszuführen (Kontextmenü "Als Administrator ausführen") und "Capture List & Label Log" zu wählen. Das Logging in Diensten und Webapplikationen wird dann automatisch gestartet.

Log-Datei mit Debwin3 und älter erstellen

Der nächste Schritt in Sachen Fehlersuche wäre die Erstellung einer Log-Datei. Fertigen Sie diese Log-Datei mit dem Hilfsprogramm "Debwin3" an. Das Programm protokolliert alle Vorgänge vom Programmstart bis hin zur Fertigstellung eines Druckjobs. Wenn während dieses Druckjobs ein Problem auftreten sollte, so wird dieses von Debwin3 protokolliert. Debwin3 besitzt einen Zeilen-Ringpuffer, in dem nach Belieben vor- und zurückgeblättert werden kann. Dies geschieht durch die üblichen Cursor-Tasten oder den Scrollbalken.

Menübeschreibung

LOGGING

Force Debug Mode: Diese Option forciert den Debug-Modus von Applikationen, d.h. der Debug-Modus muss **nicht** zu Programmbeginn im Code über "LlSetDebug()" aktiviert werden. Alle DLLs ab Version 15 werden dies unterstützen. Der Default für diese Option ist "False" und der Status der Option wird auch nicht gespeichert. Dieses Verhalten ist so gewünscht, d.h. man muss diese Option jedes Mal bewusst neu aktivieren. Bitte beachten Sie bei der Arbeit mit "Force Debug Mode" folgende Vorgehensweise:

1. Debwin3 starten.
2. Die Option "Force Debug Mode" aktivieren.
3. Die Applikation starten und das Fehlverhalten erneut provozieren.

Logging active: (De-)aktiviert die Anzeige der Meldungen auf dem Bildschirm und sorgt dafür, dass alle zukünftigen Ausgaben an die Log-Datei angehängt werden. In älteren Debwin-Versionen war diese Option auf die Menüpunkte **Log to Window** und **Log to File** aufgeteilt.

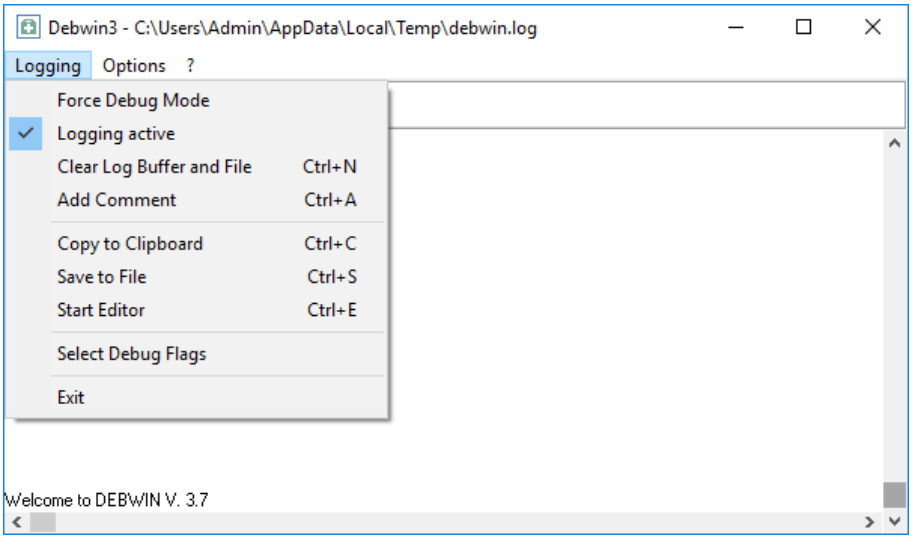
Clear Log Buffer and File: Löscht das Log und den Ringpuffer für die Bildschirmausgabe.

Add Comment: Hierüber kann ein Kommentar an die aktuelle Position eingefügt werden.

Copy to Clipboard: Kopiert den gesamten Puffer in die Zwischenablage.

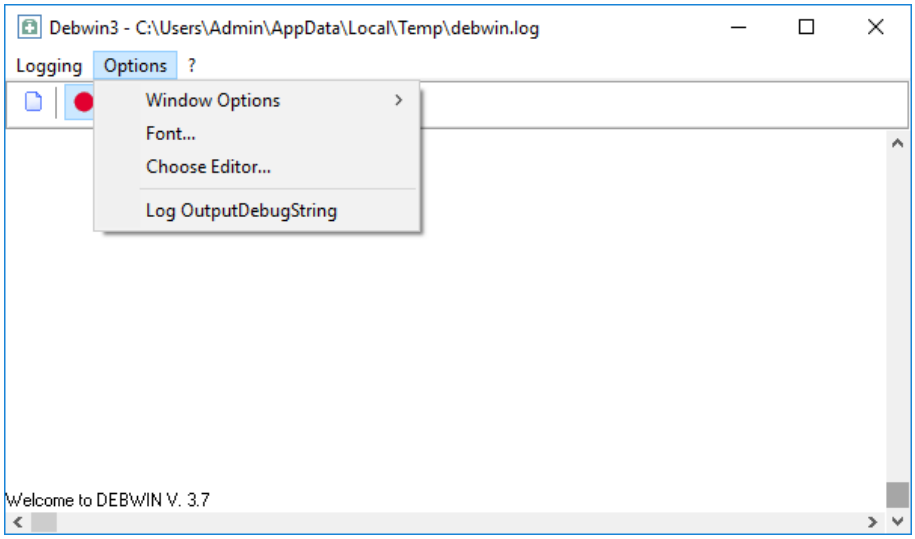
Save to File: Öffnet einen Dateiauswahldialog, in dem Sie einen Dateinamen wählen können.

Start Editor: Startet den konfigurierten Editor.

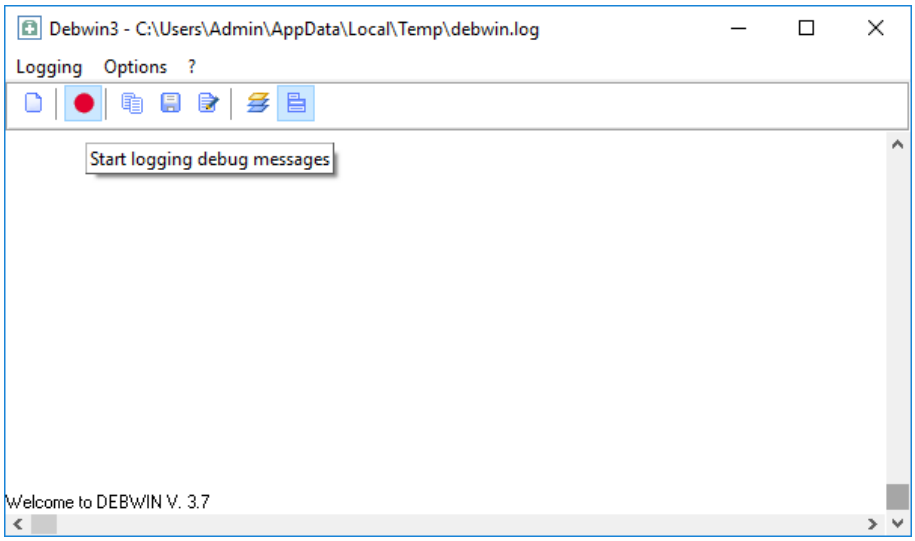


OPTIONS
Window Options:

Debug Output Brings Debwin to Front:	Debwin wird immer in den Vordergrund geholt, wenn neue Ausgaben erfolgen.
Stay on Top:	Das Debwin-Fenster bleibt immer im Vordergrund.
Font:	Erlaubt die Wahl eines alternativen Fonts.
Choose Editor:	Erlaubt die Wahl eines alternativen Editors.
Log OutputDebugString:	Zeichnet Ausgaben auf, die über die Windows API "OutputDebugString" ausgegeben werden.



TOOLBAR und TITELLEISTE
Die Buttons verfügen über Tooltipps, die die jeweilige Funktion erklären. In der Titelleiste finden Sie eine Pfadangabe, welche auf den Speicherort der Log-Datei verweist.



Erstellen einer Log-Datei auf einem lokalen Entwicklungs-PC

- Schalten Sie den Debug-Modus **zu Programmbeginn** mit dem Aufruf der Funktion "LlSetDebug()" ein (oder nutzen Sie die Option "Force Debug Mode").

bei .NET: "LL.Debug = LlDebug.Enabled;"
bei VCL: "LL.Debug = 1;"

- Starten Sie Debwin3 und stellen Sie sicher, dass das Logging in Debwin aktiviert ist (Optionen "Log to Window" und "Log to File". Bitte achten Sie hier auf die Hinweise aus der Programmierer-Referenz).
- Starten Sie nun Ihre Applikation und führen Sie diese bis zum Auftreten des Problems aus
- Die Debug-Ausgaben werden nun von Debwin3 verwertet und ausgegeben
- Speichern Sie anschließend die Ausgaben mit "Save to File" ab
- Die Erstellung der Log-Datei ist somit abgeschlossen

Erstellen einer Log-Datei bei Server-/Webapplikation
Sofern List & Label innerhalb eines Services verwendet wird, ist eine Ausgabe von Debug-Meldungen in Debwin3 nicht mehr möglich, da dieser Service in einem anderen Benutzer-Kontext als die angezeigte Workstation läuft.

Um dennoch Debug-Informationen zu erhalten, rufen Sie "LlSetDebug" bitte wie folgt auf:

LlSetDebug(LL_DEBUG_CMBTLL | LL_DEBUG_CMBTLL_LOGTOFILE)

Alle Debug-Ausgaben werden nun in die Datei "combit.log" im %APPDATA% (bis Version 13: Windows)-Verzeichnis umgeleitet. Dieses Verzeichnis können Sie leicht anzeigen, indem Sie in einem Explorer-Fenster %APPDATA% als Pfad eintippen. Beachten Sie, dass Sie zur Verwendung von List & Label in einem Service entsprechende Serverlizenzen benötigen.
Das Pipe-Zeichen (|) bezeichnet den OR-Operator. Bitte setzen Sie den OR-Operator Ihrer Programmiersprache ein.
Für die .NET-Assembly beachten Sie bitte die Hinweise im Debugging-Kapitel Ihrer Programmierer-Referenz!

Log-Datei analysieren

Allgemeiner Aufbau einer Log-Datei

Die mit Debwin3 erstellten Log-Dateien haben generell den gleichen Aufbau. Als erstes wird der sogenannte "Header" erstellt. Diesem folgen die eigentlichen Debug-Ausgaben, welche die Funktionsaufrufe, Rückgabewerte und eventuelle Fehlercodes enthalten.

Header

```
SysInfo of                  : 24.07.2011 11:10:39
Application                 : ???\COMBIT\LL??\LLDEMO32.EXE [??.000]
List & Label                : ???\COMBIT\LL??\REDIST...\CMLL??.DLL [??,0,0,0 (11-08-06 08:13)]
Serial number               : *********
LL flags                    : USE_IDP,,DD(0),MCBS(disabled)
User and system name        : ***** on *****
Open jobs                   : 1 (in this task)
ACP, OEMCP, LIBCP           : 1252, 850, 1252
Keyboard                    : 00000407
Max. RTF version on system  : 0401
OS version                  : Windows NT 5.1, build 2600, Service Pack 3, Uniprocessor Free
                            :Emulated OS: Windows XP (Professional), (Uniprocessor Free),
                            (x86-32 Processor) v5.1 Build:2600 Service Pack:3
                            : Underlying OS: Windows XP (Professional), (Terminal Services in
                            Remote Admin Mode), (Uniprocessor Free), (x86-32 Processor) v5.1
                            Build:2600 Service Pack:3
Temporaries                 :
GetTempPath()               : C:\Dokumente und Einstellungen\????\Lokale Einstellungen\Temp\
                            (hard disk, 5970456 KB free), R/W check OK
        env(TEMP)           : C:\DOKUME~1\???\LOKALE~1\Temp (hard disk, 5970456 KB free)
        env(TMP)            : C:\DOKUME~1\???\LOKALE~1\Temp (hard disk, 5970456 KB free)
        env(PATH)           : C:\WINDOWS\system32; C:\WINDOWS; C:\WINDOWS\System32\Wbem;
                            C:\Programme\Microsoft SQL Server\90\Tools\binn\
Printers                    : PDFCreator: PDFCreator on PDFCreator:, Status 00000000
                            : Microsoft XPS Document Writer: Microsoft XPS Document Writer on XPSPort:,
                            Status 00000000
                            : An OneNote 2007 senden: Send To Microsoft OneNote Driver on Send To
                            Microsoft OneNote Port:, Status 00000000
Default printer             : PDFCreator
```

System-Informationen

Pfadangaben TEMP-Dateien

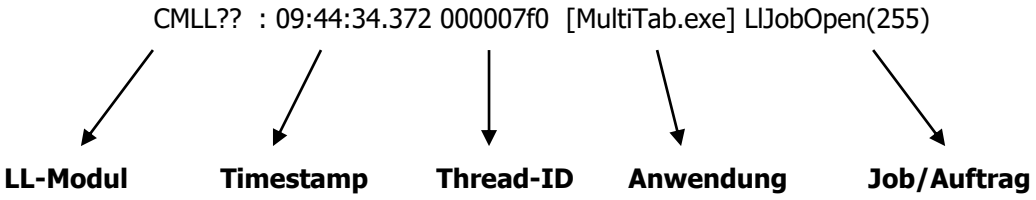
Vorhandene Drucker

➔ eingerichteter Standard-Drucker

System-Informationen:	Angaben zu Ihrem System Angaben zur List & Label-Version, Unterversion, Service Pack
Pfadangaben TEMP-Dateien:	Speicherort für temporäre Dateien beim Druck Verfügbarer Speicherplatz auf Speicherpfad Lese-/Schreibrechte Überprüfung (R/W check OK)
Vorhandene Drucker:	Auflistung aller auf dem System eingerichteter Drucker
Standard-Drucker:	Vom System verwendeter Standard-Drucker

Anhand dieser Header-Informationen können Sie nun auch Ihre aktuelle List & Label-Version in Erfahrung bringen. **Dieser Header sollte bei Log-Dateien für den Support immer enthalten sein.**

Debug-Ausgaben
Wie bereits erwähnt folgen dem Header die eigentlichen Ausgaben. Eine solche Zeile könnte zum Beispiel so aussehen:



LL-Modul
- Kennzeichnet das verantwortliche List & Label Modul

Timestamp
- Gibt die Uhrzeit der Ausführung des Druckjobs zurück
- Ist von Bedeutung, wenn bei der Ausführung des Druckjobs zum Beispiel eine unerklärliche Pause von ca. 30 Sekunden gemacht wird, in denen nichts geschieht
- D.h. um die Ursache zu finden, müssten Sie zuerst die Zeile ausfindig machen, bei der ein Zeitsprung von 30 Sekunden ersichtlich ist

Thread-ID
- Kennzeichnet den Thread und hilft bei der Unterscheidung von verschiedenen Threads

Anwendung
- Benennt die ausgeführte Anwendung, hier ist es das List & Label Beispiel "MultiTab"

Job/Auftrag
- Kennzeichnet den aktuell ausgeführten Auftrag, hier ist es das Öffnen des Projektes

Fehlererkennung
Aufgetretene Fehler sind in der Log-Datei relativ einfach aufzuspüren. Jeder Fehler liefert einen negativen Fehlercode zurück. Das heißt jeden Fehlercode ist ein "=-" vorangestellt. Sie gelangen am schnellsten zu dem Fehler, in dem Sie die Log-Datei nach "=-" durchsuchen. Wichtig ist es darauf zu achten, den zuerst aufgeführten Fehler ausfindig zu machen. Dieser kann auch Auslöser für andere, später auftretende Fehler sein. Eine Übersicht aller Fehler finden Sie im Anhang der Programmierer-Referenz. Eine solche Fehlerzeile könnte zum Beispiel so aussehen:

CMLL?? : 15:33:32.343 000009e8 =-12 (Während des Druckens ist ein Fehler aufgetreten. Möglicherweise ist der Druckspooler überlastet, oder es ist kein Speicherplatz mehr vorhanden. Abhilfe schafft meist auch die Einstellung des Direktdrucks ohne Spooler. Mögliche Ursache bei Direktdruck: allgemeiner Druckerfehler, Papierstau.)

Ausschlaggebend ist hier die Ausgabe "=-12". Dieser Fehler wird von der "cmll??.dll" gemeldet. Des Weiteren erkennt man den genauen Zeitpunkt des Auftretens und die zugehörige Thread-ID. In den runden Klammern folgt eine Beschreibung des Fehlers.

Die Fehlercodes (-996), (-997) und (-998) sind nicht unbedingt als Fehler anzusehen. Diese dienen als Hinweise.

CMLL?? : 09:44:38.370 000007f0 =-996 (Der Tabellenname hat sich geändert.)
Diese Meldung deutet darauf hin, dass sich der Tabellenname der zu druckenden Tabelle geändert hat.

CULL?? : 11:11:47.263 00000340 =-998 (Momentaner Datensatz passte nicht mehr auf die Seite.)
Diese Meldung weist darauf hin, dass der momentane Datensatz nicht mehr auf die Seite passte. Dieser Rückgabewert wird benötigt, um zu melden, dass eine neue Seite initialisiert werden muss und der momentane Datensatz erneut geschickt werden muss.

CMLL?? : 09:48:23.389 00001c21 [MultiTab.exe] =-17 (Keine Vorschaudateien gefunden, Datei ist beschädigt oder leer.)
Parameterfehler, Erklärung in der Klammer ist sehr aussagekräftig. Es ist ein Fehler beim Öffnen der Vorschau aufgetreten.

Überprüfen von Rückgabewerten
Nach jedem Aufruf einer Funktion empfiehlt es sich die Rückgabewerte zu überprüfen, um eine korrekte Ausführung sicherstellen zu können.
Korrekte Funktionsausführungen liefern einen Wert von "0" zurück. Alle Rückgabewerte kleiner Null "-??" deuten auf Fehler bzw. Hinweise hin.
Unter .NET werden statt der negativen Rückgabewerte Exceptions geworfen. Dies ermöglicht es Ihnen, alle möglichen Fehler bequem in einem catch-Block abzufangen.

Dump-Datei erstellen

Sie können die Dump-Datei mit zwei verschiedenen Tools erstellen, entweder ProcDump oder WinDbg:

1) ProcDump
Mit diesem Hilfsprogramm von Microsoft können Sie ganz einfach Process Dumps (Prozesssicherungen) von CPU-Spitzen in Applikationen überwachen. Das Tool hilft Ihnen außerdem dabei, Prozessabbilder zu erzeugen, wenn sich Prozesse aufhängen oder unbehandelte Ausnahmen auftreten.
http://technet.microsoft.com/en-us/sysinternals/dd996900.aspx

So erstellen Sie Prozessabbilder:
- Laden und extrahieren Sie das Tool auf Ihren Desktop
- Starten Sie Ihre Anwendung
- Führen Sie ProcDump mit folgenden Befehl in der Kommandozeile aus:
procdump -accepteula –e –ma <Anwendung>.exe –o %TEMP%\dump.dmp
--> Jetzt werden verschiedene Informationen über den Prozess in der Kommandozeile angezeigt
- Führen Sie bitte die Schritte bis zu dem Fehlverhalten erneut aus
--> ProcDump erzeugt automatisch eine Dump-Datei, sobald das Problem auftritt

2) WinDbg
Unter folgendem Link stehen diverse "Standalone Debugging Tools" für die verschiedenen Windows Versionen zum Download bereit:
http://msdn.microsoft.com/en-us/windows/hardware/gg463009

Für Windows XP und Windows Vista nutzen Sie bitte den Link "Get the standalone debugging tools for Windows XP as part of Windows 7 SDK".
Für Windows 7 und neuer nutzen Sie bitte den Link "Get Debugging Tools for Windows (WinDbg) (from the SDK)".

So erstellen Sie Ihre Dump-Datei:
- Installieren Sie das Tool (ab Windows 7: Setzen Sie bei der Installation lediglich den Haken für "Debugging Tools for Windows".
- Starten Sie nun WinDbg über den erstellten Startmenüeintrag
- Starten Sie den betroffenen Prozess unter dem Debugger über "File > Open Executable"
- Starten Sie das Debugging über "Debug > Go" oder F5
- Falls das Problem auftritt, geben Sie bitte ".dump /ma c:\combit.dmp" in die Debugger-Kommandozeile ein
--> WinDbg erzeugt jetzt eine Dump-Datei

Support kontaktieren

Wenn Sie den Fehler selbst nicht ausfindig machen können und den Support kontaktieren müssen, ist es im Sinne einer raschen Problemlösung ratsam, dem Support-Team so viele Informationen wie nur möglich zur Verfügung zu stellen. Im Normalfall werden folgende Informationen (Dateien) benötigt:

1. Genaue *Fehlerbeschreibung* (Wo tritt der Fehler auf? Bei welchem Arbeitsschritt?). Unser Online-Formular unterstützt Sie bei der Bereitstellung der wichtigsten Informationen. Zusätzlich kann noch die *Projektdatei* (*.lst, *.lsr etc.) + Vorschau-Datei (*.ll) mitgeschickt werden, sofern das Problem dort sichtbar ist.

und

2. Wenn Sie das Problem mit unserer "*Beispielanwendung*" (DemoApplication22.exe) reproduzieren können, genügt die dort verwendete Druckvorlage (Projektdatei), damit der Support das Problem auch nachvollziehen kann. Ansonsten ist ein *lauffähiges und minimiertes Source Code Beispiel* notwendig. Sie können hierzu eines unserer vielen mitgelieferten Beispiele nutzen oder aber Sie reduzieren Ihre Anwendung soweit, dass diese problemlos vom Support ausführbar ist.

Sollte dies nicht möglich sein, kann in manchen Fällen auch eine Log-Datei helfen:

3. Vollständige *Log-Datei* (mit **vollständigen Header** die den Druckverlauf bis hin zum eigentlichen Problem zeigt)

Bei Abstürzen oder Exceptions kann auch Folgendes helfen.

4. Dump