

# **unabhängiges storage-system**

**Vortrag von**

**Thomas Weise**

**11.06.2002 13.45**

**1/336**

# Gliederung

- ★ Interfaces, COM und OLE
  - ★ Interfaces
  - ★ COM
  - ★ OLE
- ★ Structured Storage Model (SSM)
- ★ independent implementation
  - ★ Interface Hierarchie Diagramm
  - ★ Objekt Hierarchie Diagramm
- ★ einige Konzepte
- ★ Ausblick
- ★ Fragen

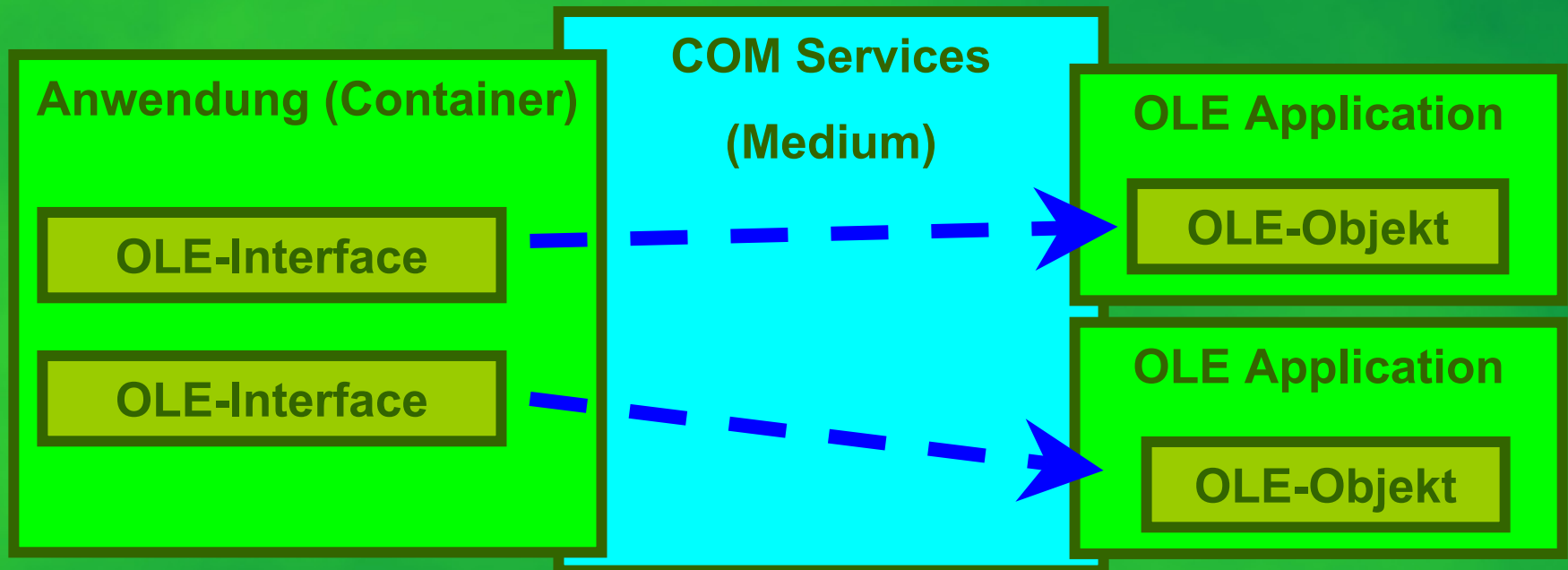
*eigene Erweiterungen eines Standards*

# COM, Interfaces and OLE

Interface ≡ Export von Routinen/Methoden eines Objekts

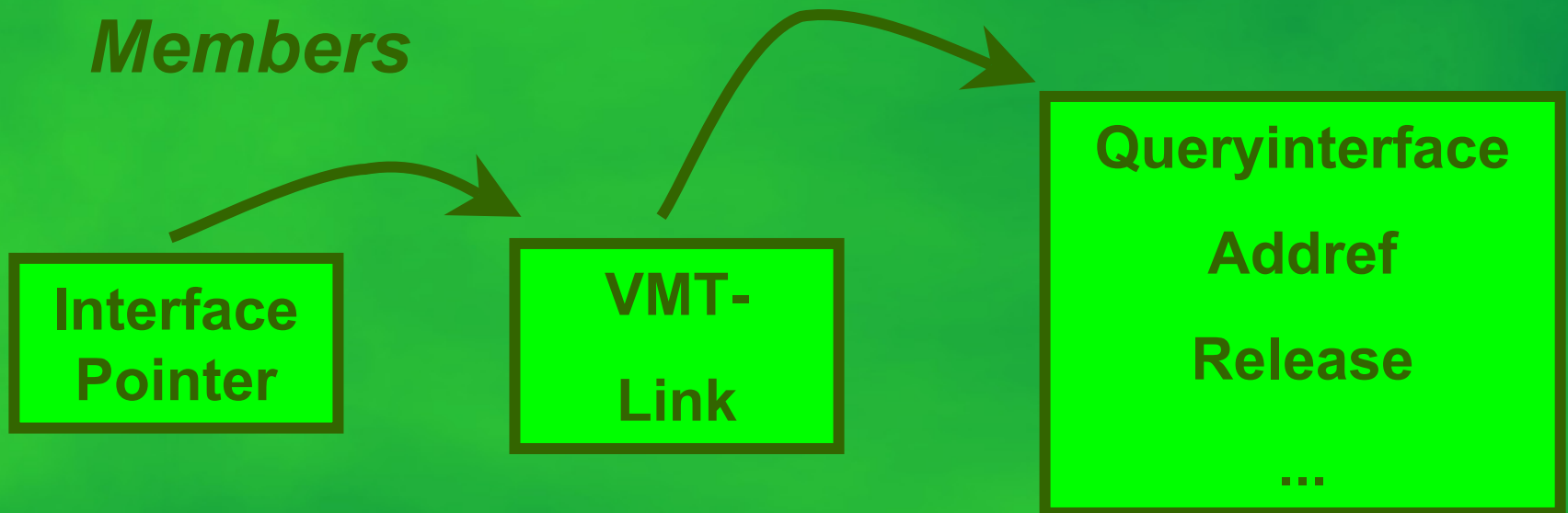
COM ≡ Compound Object Model

OLE ≡ Object Linking and Embedding



# Interfaces

- ★ Objekt kann beliebig viele Interface einbinden
- ★ alle Interface stammen von IUnknown ab
- ★ VMT  $\equiv$  Virtual Method Table
- ★ *Dlls exportieren Routinen als Interface-Members*



# COM

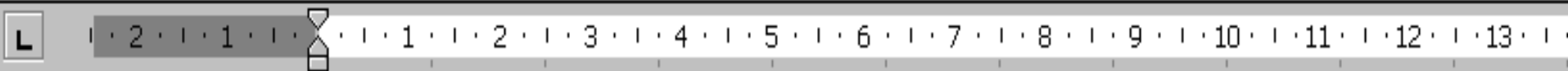
- ★ Stufe zwischen OLE und Interfaces
- ★ regelt Lebensdauer (reference counting) und z.B. Lizenzierung von Objekten
- ★ Beschreibung von Objekten in z.B. TLBs mit IDL
- ★ DCOM, COM+: Objekte im Netzwerk via Marshalling
- ★ *DLLs werden als COM-Objekte implementiert*  
*(Ausblick: Marshalling von DLLs)*

# OLE

- ★ Automation von Aufgaben
- ★ OLE Container: ein Host-Programm kann Objekte von völlig anderen Applikationen einbinden, ohne über deren innere Beschaffenheit, Ort der Applikation, Datenformat usw. bescheid zu wissen.
- ★ z.B. Word, Excel usw.
- ★ *interne Threadsicherheit (normalerweise vom Host gewährleistet) -> multiple Hosts*



A16 = Und im Excel eine Grafik



Eine eingebettete Musikdatei:   
 und eine Powerpoint Präsentation:

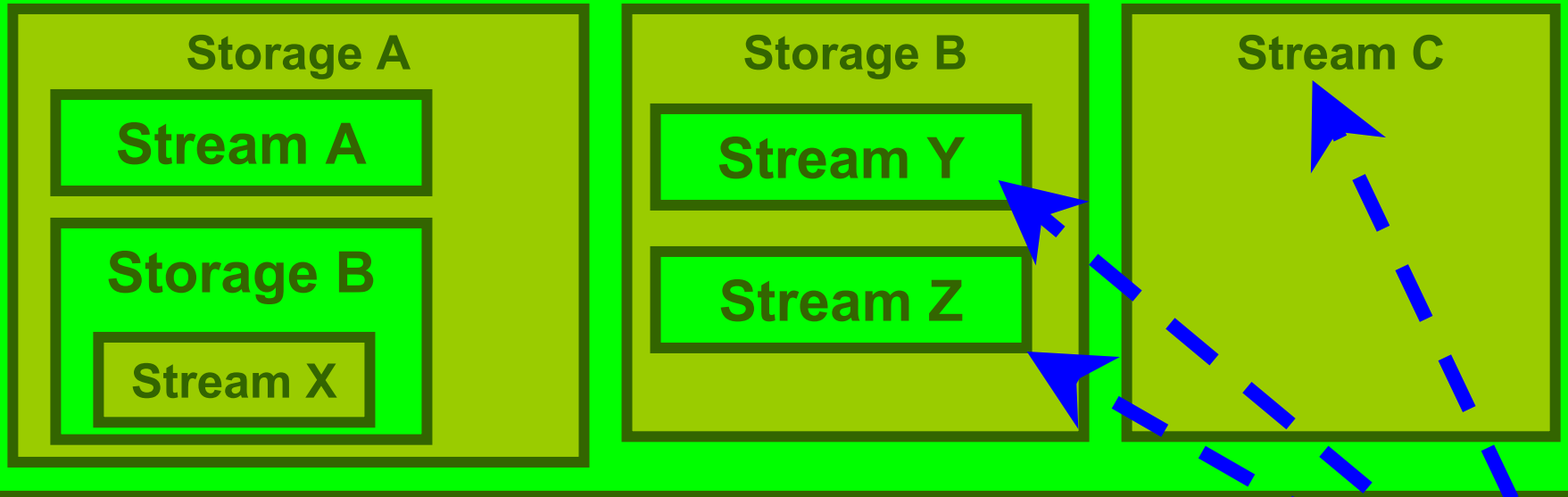


	A	B	C	D	E
1	A1+A2=34	Hallo Welt von			
2		Excel in Word			
3					
4					
5					
6					
7					
8					
9					
10					
11					
12					
13					
14					
15					
16	Und im Excel eine Grafik				
17					
18					
19					
20					



# Structured Storage Model

Datei auf Datenträger (= root storage)

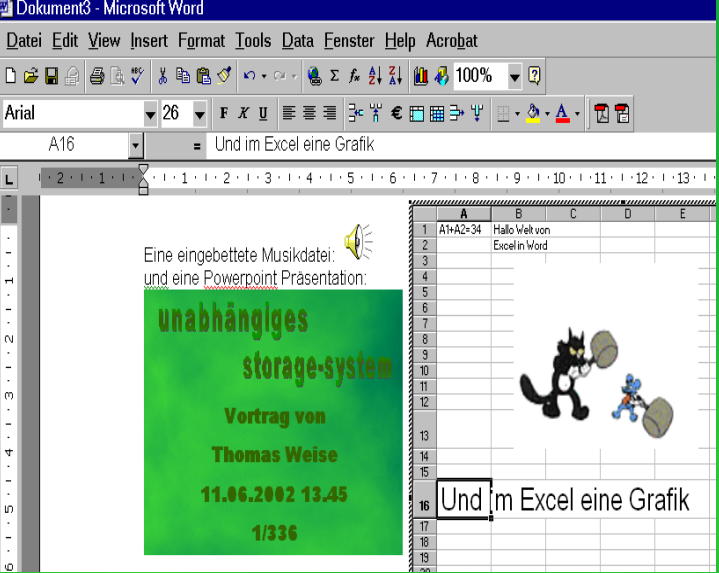


Storage ≡ Verzeichnis

Stream ≡ sequenzielle Datei + Seek

Lockbytes ≡ Array of Byte

*File* ≡ *Stream + Lockbytes + vieles andere*



**Word-Dokument**



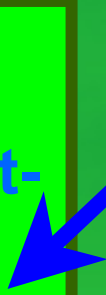
**IPersistStorage**

**Storage**  
**Word-Objekt**



**IPersistStorage**

**Storage**  
**Powerpoint-Objekt**



**IPersistStream**

**Stream**  
**Powerpoint-Daten**

**IPersistStream**

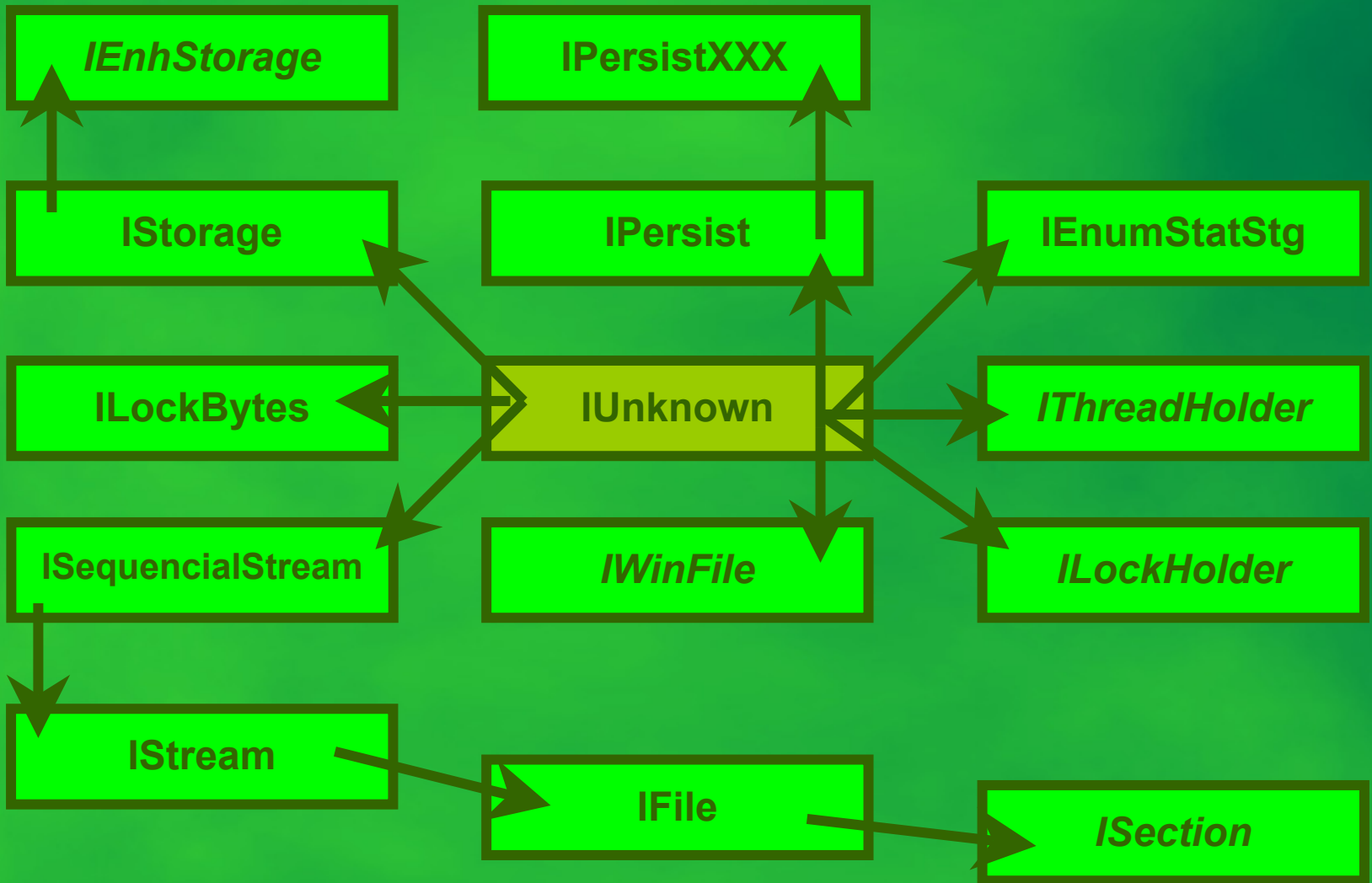


**Stream**  
**Texte**

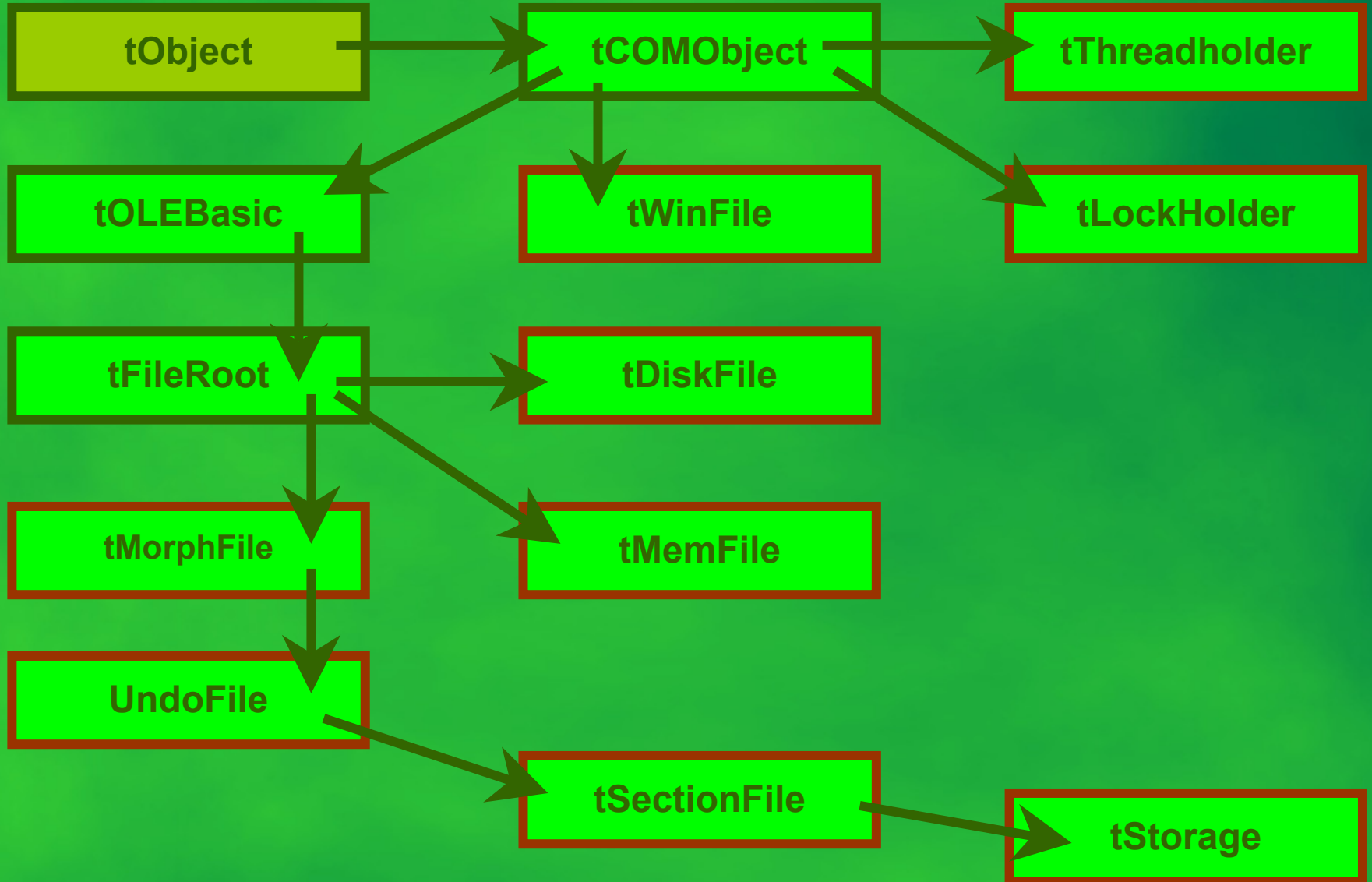
# independent Implementation

- ★ 2 DLLs            `resources.dll`, `files.dll`
- ★ `resources`    shared reference counted memory objects, unicode string methods, fast list drivers, utility routines
- ★ `files`            Routinen, um SSM-Zugriff Com-Objekte zu erzeugen
- ★ beide DLLs können sowohl als shared-COM-Object als auch „normal“ importiert werden (`files` lädt `resources` intern als shared-COM-Object)

# Interface Hierarchy



# Objekt Hierarchie



# einige Konzepte

- ★ keine vordefinierten Objekte, Routinen verwendet
- ★ Grundroutinen über Assemblercode implementiert
- ★ absolute Threadsicherheit, beginnend bei tComObject durch speziellen Assemblercode
- ★ DLLs als COM-Objekte
- ★ verschmelzen von ILockbytes und IStream
- ★ asynchrone Lese- und Schreiboperationen
- ★ Datenbewegung innerhalb und zwischen Dateien
- ★ typisierte Lese- und Schreiboperationen
- ★ Unterstützung von IPersistXXX

# Ausblick

im Moment keine Zeit zur Weiterentwicklung,  
aber geplant:

- ★ Erweiterung des rechteorientierten Arbeitens
- ★ Erhöhung der Performance, evtl. Öffnen-Algorithmus verbessern
- ★ Enhanced Storage als Basis für schnelle Datenbanken – Streams als Tabellen würden in sich selbst verschachtelte Datenbanken zulassen
- ★ Implementierung von Marshalling – Enhanced Storages im Distributed Environment

# Fragen

evtl. Zeigen des Codes, Beispiele aus Code

[tweise@gmx.de](mailto:tweise@gmx.de)

[www.tu-chemnitz.de/~weist/it/ssm/index.htm](http://www.tu-chemnitz.de/~weist/it/ssm/index.htm)