bib International College

Threads

Teil 1

Multitasking vs. Multithreading

Multitasking:

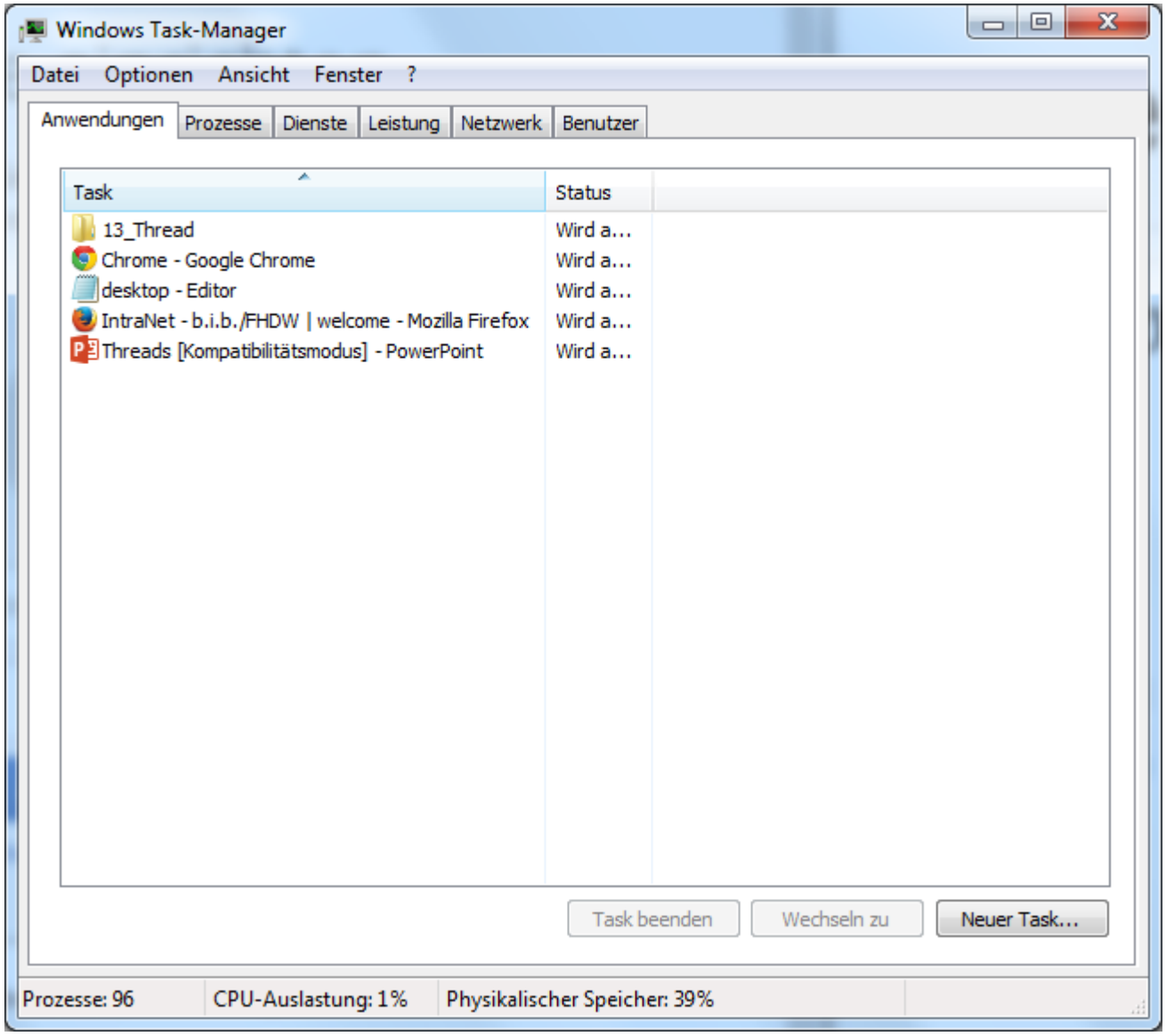
Fähigkeit eines Betriebssystems, mehrere Aufgabe nebenläufig auszuführen. Quasi-Parallelität auf einem einzelnen Prozessorkern. CPU schaltet zwischen den Tasks hin und her.

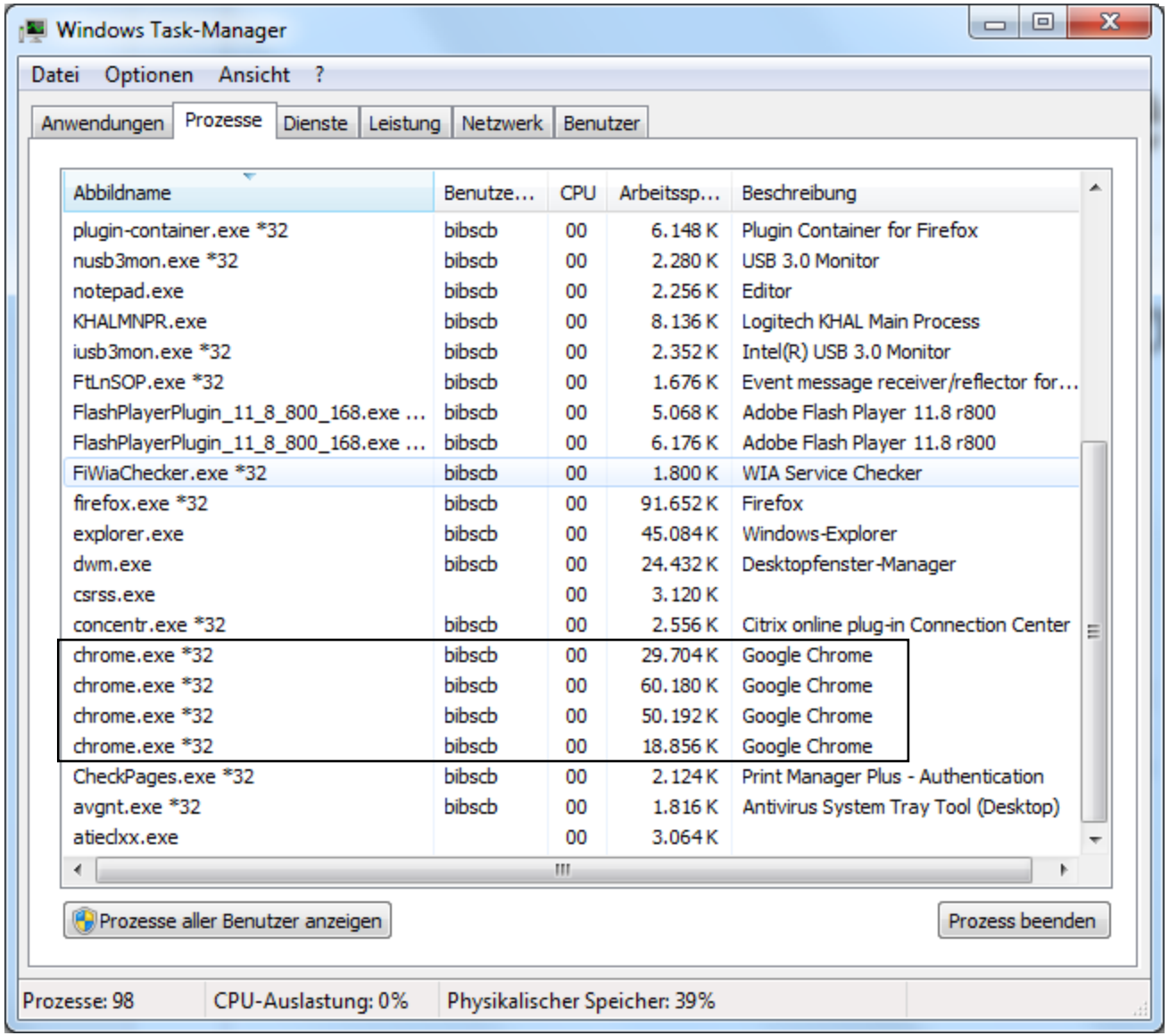
Multithreading:

Gleichzeitiges Abarbeiten mehrerer Ausführungsstränge (Threads) innerhalb eines einzelnen Prozesses/Task. Skalierbar, d.h. bei Bedarf Hinzufügen neuer Threads

Prozesse

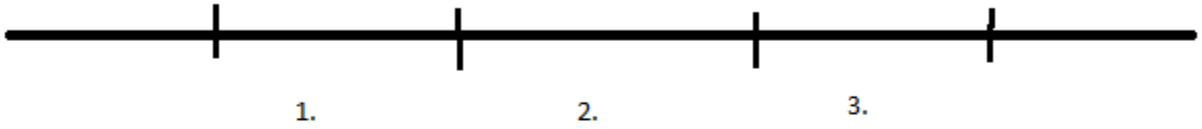
- Laufende Instanz einer Anwendung
- Setzt sich aus Codabschnitten(aus Programm und Bibliotheken) im Speicher zusammen
- Eigener Adressraum, eigene Ressourcen(z.B. Threads, Dateien, dynamisch reservierter Speicher)
- Der Code im Adressraum eines Prozesses wird durch einen Thread (MainThread) abgearbeitet.
- Wenn der letzte Thread beendet ist → Beenden
- Anwendung und Prozess ist nicht dasselbe.

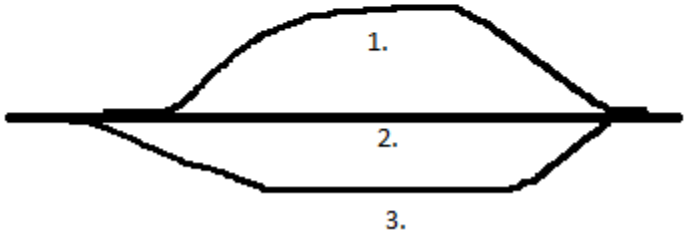

Windows Task-Manager
Datei
Optionen
Ansicht
Fenster
?
Anwendungen
Prozesse
Dienste
Leistung
Netzwerk
Benutzer
Task
Status
13_Thread
Wird a...
Chrome - Google Chrome
Wird a...
desktop - Editor
Wird a...
IntraNet - b.i.b./FHDW | welcome - Mozilla Firefox
Wird a...
Threads [Kompatibilitätsmodus] - PowerPoint
Wird a...
Task beenden
Wechseln zu
Neuer Task...
Prozesse: 96
CPU-Auslastung: 1%
Physikalischer Speicher: 39%


Windows Task-Manager
Datei Optionen Ansicht ?
Anwendungen Prozesse Dienste Leistung Netzwerk Benutzer
Abbildname Benutze... CPU Arbeitssp... Beschreibung
plugin-container.exe *32 bibscb 00 6.148 K Plugin Container for Firefox
nusb3mon.exe *32 bibscb 00 2.280 K USB 3.0 Monitor
notepad.exe bibscb 00 2.256 K Editor
KHALMNPR.exe bibscb 00 8.136 K Logitech KHAL Main Process
iusb3mon.exe *32 bibscb 00 2.352 K Intel(R) USB 3.0 Monitor
FtLnSOP.exe *32 bibscb 00 1.676 K Event message receiver/reflector for...
FlashPlayerPlugin_11_8_800_168.exe ... bibscb 00 5.068 K Adobe Flash Player 11.8 r800
FlashPlayerPlugin_11_8_800_168.exe ... bibscb 00 6.176 K Adobe Flash Player 11.8 r800
FiWiaChecker.exe *32 bibscb 00 1.800 K WIA Service Checker
firefox.exe *32 bibscb 00 91.652 K Firefox
explorer.exe bibscb 00 45.084 K Windows-Explorer
dwm.exe bibscb 00 24.432 K Desktopfenster-Manager
csrss.exe 00 3.120 K
concentr.exe *32 bibscb 00 2.556 K Citrix online plug-in Connection Center
chrome.exe *32 bibscb 00 29.704 K Google Chrome
chrome.exe *32 bibscb 00 60.180 K Google Chrome
chrome.exe *32 bibscb 00 50.192 K Google Chrome
chrome.exe *32 bibscb 00 18.856 K Google Chrome
CheckPages.exe *32 bibscb 00 2.124 K Print Manager Plus - Authentication
avgnt.exe *32 bibscb 00 1.816 K Antivirus System Tray Tool (Desktop)
atieclxx.exe 00 3.064 K
Prozesse aller Benutzer anzeigen
Prozess beenden
Prozesse: 98 CPU-Auslastung: 0% Physikalischer Speicher: 39%

Prozess

Sequentielle Ausführung





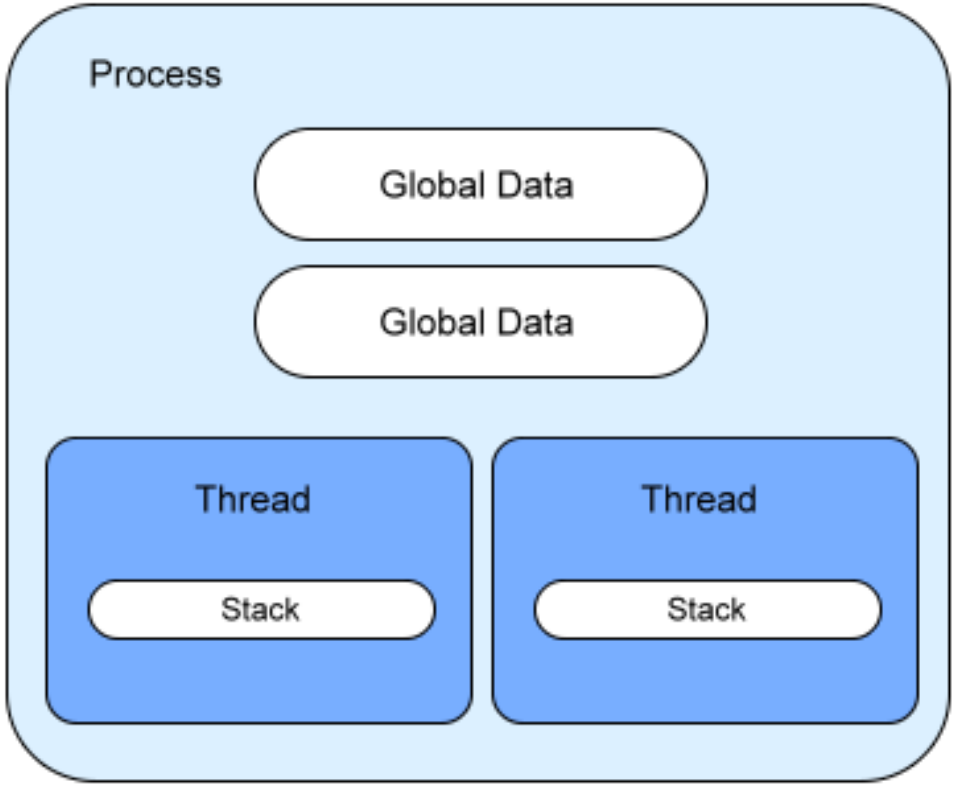
Parallele Ausführung mehrerer Threads

Threads

- Ein Prozess besitzt mind. einen Thread, der Code ausführt.
- Ausführungsstrang/ sequentielle Ausführungsreihenfolge in der Abarbeitung eines Programms
- Threads des gleichen Prozesses teilen sich Betriebsmittel (Codesegment, Datensegment, verwendete Dateideskriptoren)
- Threads des gleichen Prozesses verwenden voneinander unabhängige Stacks (unterschiedliche Abschnitte des Adressraums)
- Der Heap wird von allen Threads des gleichen Prozesses gemeinsam verwendet → Konflikte → Synchronisation
- Zwei Arten: Kernelthread, User Thread

bib International College

Threads



Free Threading

Threads

Threads = leichtgewichtige Prozesse, kleinste ausführbare Einheit

Präemptives Multitasking:

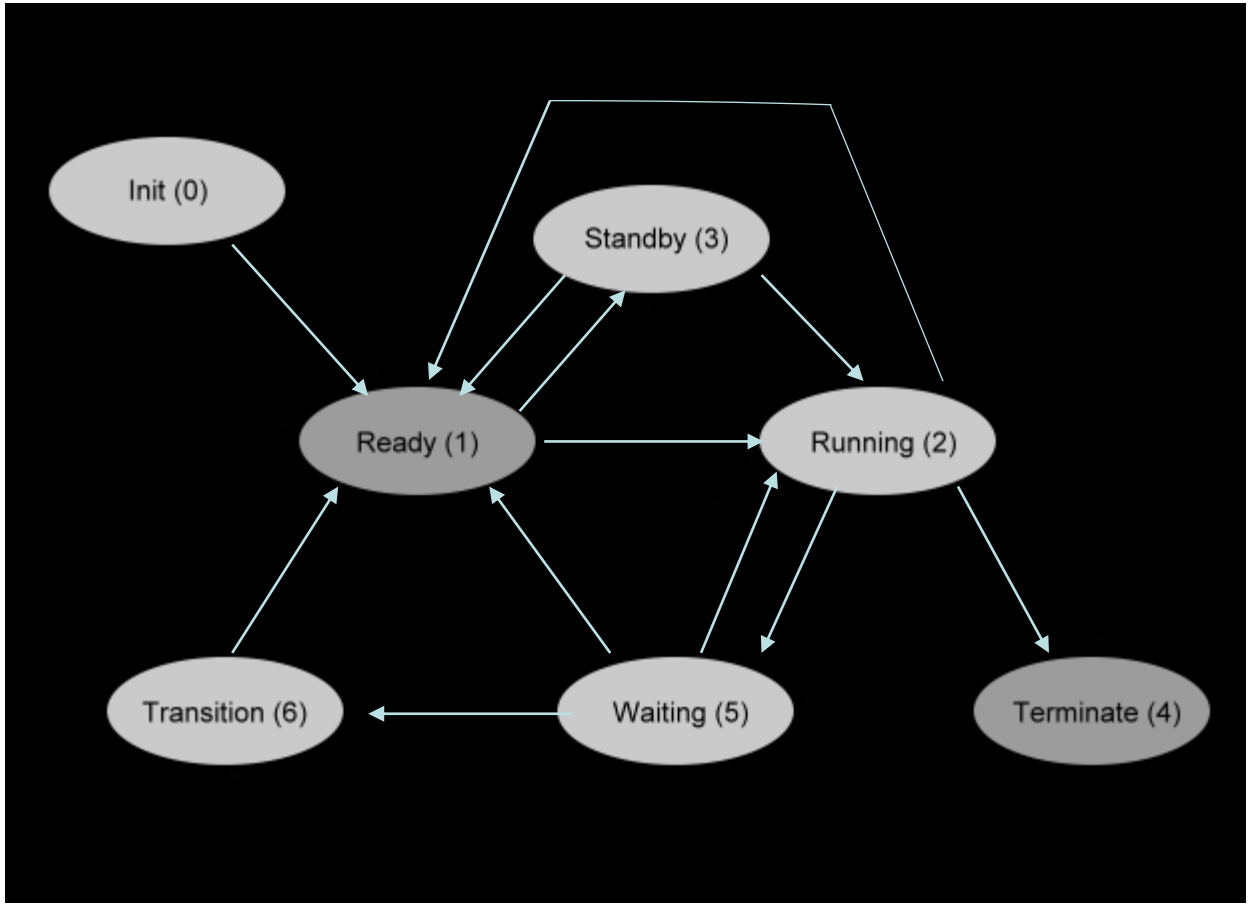
- der Betriebssystemkern steuert die Abarbeitung der einzelnen Prozesse
- Verwendung der Vorrangwarteschlange in Verbindung mit der Round-Robin-Scheduling-Strategie (Zeitscheibenstrategie, time slicing)
- Bei Mehrkern-Prozessor-Systemen werden die Threads auf die Kerne verteilt.

Kooperatives Multitasking:

- Jedem Thread ist selbst überlassen, wann er die Kontrolle an den Kern übergibt.

bib International College

Thread Status



Ready:
Thread ist einsatz- bzw. ausführungsbereit und wird vom Scheduler beachtet.

Standby:
Thread, der als nächstes auf einem bestimmten Prozessor "am Zug" ist. Dispatcher führt einen Kontextwechsel zu diesem Thread durch. Auf dem System kann sich nur ein Thread pro CPU-Kern im Status *Standby* befinden. Der Thread kann umgangen werden, wenn ein Thread mit höherer Priorität einsatzbereit wird oder ein Interrupt auftritt.

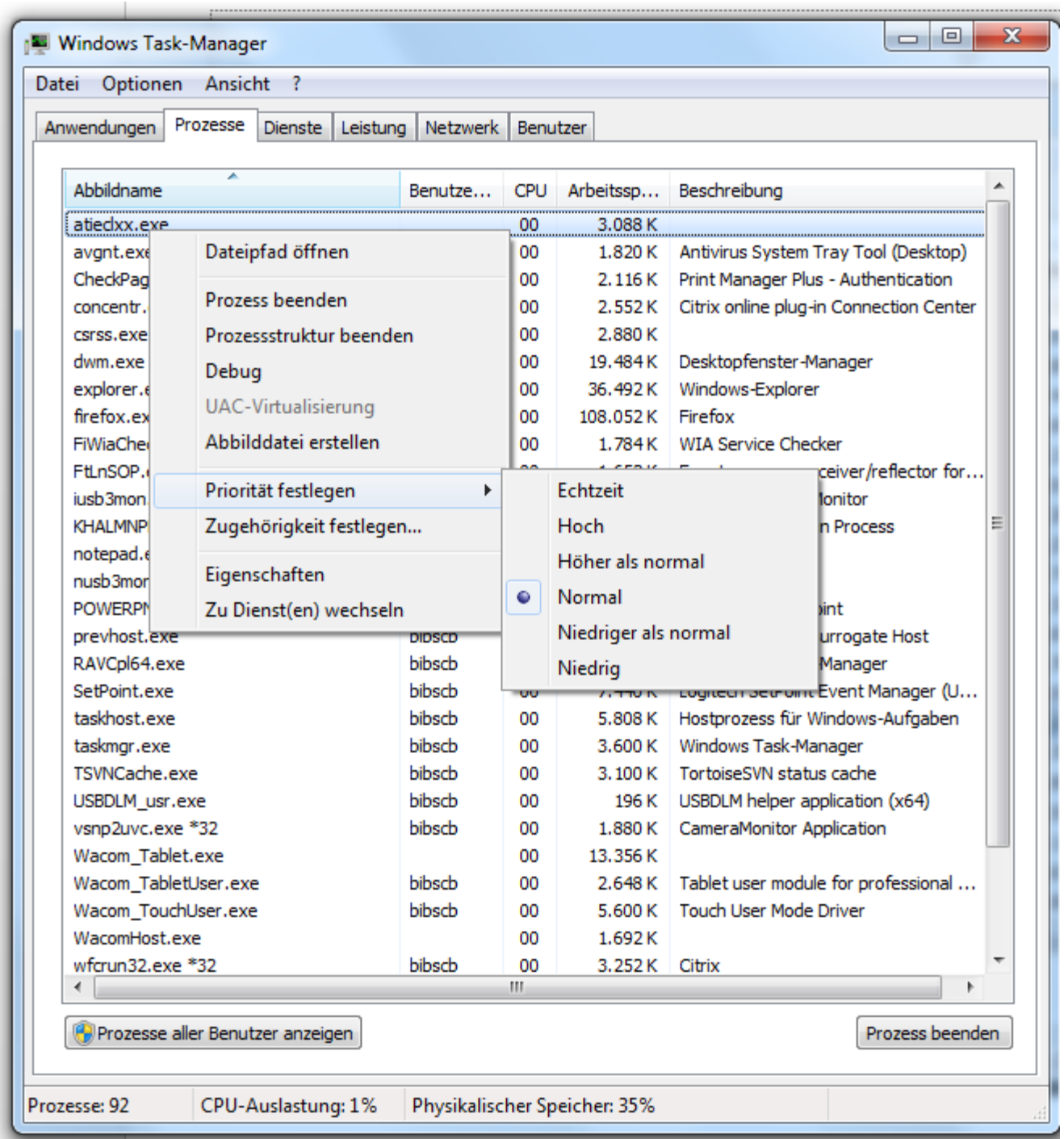
Running:
Thread wird solange ausgeführt, bis sein Quantum abgelaufen ist, er freiwillig in den Wartezustand geht, er unterbrochen oder terminiert wird.

Waiting:
Thread in den Wartezustand, wenn er auf I/O-Informationen wartet, die er zum Weiterarbeiten benötigt. Sobald diese Ressource verfügbar ist, geht er in den "Ready"-Zustand über.

Transition:
ablaufbereiter Thread, dessen Kernel-Stack nicht im Speicher verfügbar ist. Sobald der Kernel-Stack zurück in den Speicher geladen wurde, tritt der Thread in den "Ready"-Zustand ein.

Terminated:
Thread hat seine Arbeit beendet. Wird ein Thread terminiert, wird das Objekt des Threads manchmal nicht sofort gelöscht; es kann noch länger verfügbar bleiben, um es zu reinitalisieren und wieder zu verwenden.

Initialized:
Zustand wird intern verwendet, während ein Thread erzeugt wird



bib International College

bibscb/2014

Dynamische Prioritätsanpassung

Thread-Prioritäten werden vom Betriebssystem in bestimmten Situationen angehoben:

- Vervollständigung einer I/O-Operation
- Nach dem Warten auf auszuführende Ereignisse oder Semaphores
- Nachdem Threads im Vordergrund-Prozess eine Warteoperation abgeschlossen haben
- Wenn GUI-Threads aufgewacht sind, weil das Fenster aktiv wurde
- Wenn ein Thread im "Ready"-Zustand lange Zeit nicht ausgeführt wurde

Quantum/Zeitscheibe

- Anhebung der Thread-Priorität
- der Thread für ein zusätzliches Quantum mit der gesteigerten Priorität
- Priorität wird um 1 verringert und der Thread läuft für ein weiteres Quantum/ Zeitscheibe
- dieser Zyklus setzt sich fort, bis der Thread seine Basispriorität erreicht hat
- Taktinterrupt bei

  - Einkern-Prozessor-Systemen: 10 msec

  - Mehrkern-Prozessor-Systemen: 15 msec
- ClockRes zur Ermittlung der exakten Zeitspanne

Dispatcher Database

- Kernel verwaltet **Dispatcher Database** um Entscheidungen über das Scheduling zu treffen
- Speichert Informationen über wartende Threads und die Zuteilung von Threads an Prozessoren
- *Ready queues* zeigen Threads im "Ready"-Zustand an
- Pro Priorität gibt es eine Warteschlange. Die Auswahl wird durch eine 32-Bit Maske (*Ready summary)* beschleunigt
- Jedes Bit zeigt die Anzahl der wartenden Threads in der betreffenden Warteschlange an

bib International College

Multithreading in .NET

- .NET Framework hat viele Werkzeuge für das verwaltete Threading
- Grundlage für das Threading in .NET ist der Namensraum System.Threading mit einer Fülle von Klassen und Interfaces zur Unterstützung von Multithread-Programmen
- .NET Framework, Version 4 vereinfacht die Multithread-Programmierung durch die Klassen System.Threading.Tasks.Parallel und System.Threading.Tasks.Task, Paralleles LINQ (PLINQ) und ein neues Programmierungsmodell erheblich, das auf dem Konzept von Aufgaben und nicht auf Threads basiert. http://msdn.microsoft.com/library/dd460693.aspx

Klasse Thread

- .NET-Threads sind die Managed-Code Variante der unterliegenden Threads des Betriebssystems
- In .NET auf Windows, werden die Threads eins zu eins auf die nativen Win32- bzw. Win64-Threads abgebildet
- Alle .NET-Threads werden durch die Klasse Thread im Namensraum System.Threading repräsentiert. Sie bietet zahlreiche Methoden und Eigenschaften um einen verwalteten Thread zu kontrollieren
- Die ThreadState-Enumeration definiert eine Gruppe aller möglichen Ausführungszustände für Threads im .NET Framework

Ausführungszustände eines Thread

Zustand	Beschreibung
Running	Der Thread wurde gestartet, er wird nicht blockiert, und es ist keine ausstehende ThreadAbortException vorhanden.
StopRequested	Es besteht eine Anforderung für die Beendigung des Threads. Dies ist ausschließlich für die interne Verwendung vorgesehen.
SuspendRequested	Es besteht eine Anforderung für die Unterbrechung des Threads.
Background	Der Thread wird nicht als Vordergrundthread, sondern als Hintergrundthread ausgeführt. Dieser Zustand wird durch Festlegen der Thread.IsBackground-Eigenschaft gesteuert.
Unstarted	Die Thread.Start-Methode wurde für den Thread nicht aufgerufen.
Stopped	Der Thread wurde beendet.
WaitSleepJoin	Der Thread ist blockiert. Die Ursache hierfür könnte sein, dass Thread.Sleep oder Thread.Join aufgerufen wurde, dass eine Sperre angefordert wurde, z. B. durch Aufrufen von Monitor.Enter oder Monitor.Wait, oder dass auf ein Threadsynchronisierungsobjekt wie ManualResetEvent gewartet wird.
Suspended	Der Thread wurde unterbrochen.
AbortRequested	Die Thread.Abort-Methode wurde für den Thread aufgerufen, doch der Thread hat noch nicht die ausstehende System.Threading.ThreadAbortException empfangen, die ihn zu beenden versucht.
Aborted	Der Threadzustand schließt **AbortRequested** ein, und der Thread ist jetzt deaktiviert. Der Zustand hat sich jedoch noch nicht in **Stopped** geändert.