LU03e - Bitmap

Nachdem wir im Kapitel bildformate die wichtigsten Begriffe definiert haben, werden wir uns das Bitmap-Format genauer anschauen. Dieses Format verzichtet fast vollständig auf Tricks zur Reduktion des Speicherbedarfs. Dadurch ist es recht gut nachvollziehbar.

Siehe auch http://de.wikipedia.org/wiki/Windows_Bitmap

Im Bitmap-Format wird ein Bild als zweidimensionales Raster von Pixeln gespeichert. Dieses Raster mit den einzelnen Pixeln wird „Bitmap“ genannt, woher das Format auch seinen Namen hat. Zu jedem einzelnen Pixel wird die entsprechende Farbinformation gespeichert.

Alle Informationen sind als binär codierte Ganzzahlen gespeichert. Je nach Länge entspricht dies den Datentypen **byte**, **short** oder **int**.

Die Datei besteht aus drei Teilen:

Dateikopf

Die ersten 14 Byte der Datei sind der Dateikopf. Im Dateikopf werden einige allgemeine Angaben zum Bild gespeichert:

Position (Hex)	**Länge (Byte)**	**Name**	**Beschreibung**
00	2	bfType	Dateityp „BM“, ASCII-Zeichenkette 42 $4D_{16}$
02	4	bfSize	Grösse der Datei in Byte (unzuverlässig)
06	4	bfReserved	Reservierter Bereich, standardmässig **0**
0A	4	bfOffBits	Startposition der Bilddaten.

Leider wird der Dateikopf nicht von allen Programmen korrekt verarbeitet. Dies führt unter anderem dazu, dass die Dateigrösse falsch abgefüllt wird.

Informationsblock

Direkt nach dem Dateikopf folgt der Informationsblock. Der Informationsblock besteht aus den Bitmap-Eigenschaften und einer eventuellen Farbmaske und Farbtabelle.

Bitmap-Eigenschaften

Dieser Block enthält unter anderem Angaben zur Grösse des Bildes und der Farbtiefe der einzelnen Pixel.

Position (Hex)	Länge (Byte)	Name	Beschreibung
0E	4	biSize	Grösse des Informationsblocks in Byte; Wert=40_{10} / 28_{16}
12	4	biWidth	Breite der Bitmap in Pixeln
16	4	biHeight	Die Höhe der Bitmap in Pixeln Ein positiver Wert (Normalfall) bedeutet, dass die einzelnen Zeilen von unten nach oben beschrieben sind. Ein negativer Wert bedeutet, dass die Zeilen von oben nach unten beschrieben sind.
1A	2	biPlane	Wurde in älteren Formaten verwendet. In Bitmap-Dateien immer **1**.
1C	2	biBitCount	Gibt die Farbtiefe der Bitmap an. Gültige Werte: **1**, **4**, **8**, **16**, **24** oder **32**. (Siehe http://msdn.microsoft.com/en-us/library/windows/desktop/dd183376%28v=vs.85%29.aspx)
1E	4	biCompression	Gibt die Art der Komprimierung an (siehe http://msdn.microsoft.com/en-us/library/windows/desktop/dd183376%28v=vs.85%29.aspx).
22	4	biSizeImage	Grösse der Bilddaten in Byte.
26	4	biXPelsPerMeter	Horizontale Auflösung des Ziel-Ausgabegeräts, in der Regel **0**.
2A	4	biYPelsPerMeter	Vertikale Auflösung des Ziel-Ausgabegeräts, in der Regel **0**.
2E	4	biClrUsed	Anzahl der Einträge in der Farbtabelle.
32	4	biClrImportant	Anzahl der verwendeten Einträge in der Farbtabelle.

Vorsicht: Verkehrte Welt

Bei den einzelnen Werte sind die Zahlen in der verkehrten Reihenfolge gespeichert.

Beispiel:

- Breite des Bildes: 2000 Bit
- Hexadezimale Darstellung: 00 00 07 D0
- Gespeicherte Information: D0 07 00 00

Es werden also die einzelnen Zahlenblöcke von rechts nach links gespeichert.

Bilddaten

Die eigentlichen Bilddaten stellen den letzten und grössten Block dar. Der Startpunkt innerhalb der Datei steht im Feld `bfOffBits` im Dateikopf.

Die Bilddaten enthalten die Farbinformation der Pixel Zeile für Zeile von links nach rechts. Wie die Farbinformation der einzelnen Pixel gespeichert wird, hängt von der Farbtiefe (`bfBitCount`) und der Komprimierung (`bfCompression`) ab. Es würde zu weit führen, sämtliche Varianten zu beschreiben. Daher beschränke ich mich an dieser Stelle auf zwei exemplarische Varianten.

24 Bit pro Pixel / keine Kompression

Die Farbinformation besteht aus je 8 Bit (1 Byte) für die Farben Blau, Grün und Rot. Somit können 16'777'216 verschiedene Farben gespeichert werden.

Diese Art der Codierung verwenden wir auch für die Angabe einer Farbe in Stylesheets (CSS). Zum Beispiel `background-color: #cc9933;`. Allerdings ist die Reihenfolge der Farben umgekehrt:

- Stylesheet: Rot Grün Blau
- Bitmap: Blau Grün Rot

1,4 oder 8 Bit pro Pixel / keine Kompression

Bei dieser Art der Codierung steht die Farbinformation in einer Farbtabelle. Die Daten des Pixels enthalten den Index der jeweiligen Farbe in der Farbtabelle.

Beispiel Auszug aus einer Bitmap-Datei:

```
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
0  42 4d 2a 75 03 00 00 00 00 00 36 00 00 00 28 00
1  00 00 4c 02 00 00 c6 02 00 00 01 00 08 00 00 00
2  00 00 f4 74 03 00 00 00 00 00 00 00 00 00 00 00
3  00 00 00 00 00 00 ff ff ff ff ff ff ff ff ff ff
```

Informationen aus einer Bitmap lesen

1. Welche Information suche ich?

Annahme: Ich möchte die Breite des Bildes (Bitmap) kennen. Dazu suche ich im Dateikopf oder Informationsblock wo die gewünschte Information gespeichert ist:

- Name: biWidth
- (Start)Position: 12_{16}
- Länge: 4 Byte

2. Wert ablesen

Aus der Bitmap-Datei suche ich die Startposition. Dann übernehme so viele Bytes, wie die Länge angibt.

Im oben stehenden Auszug wäre das. **$\mathbf{4c\ 02\ 00\ 00_{16}}$**

3. Umrechnen

In der Bitmapdatei sind die Bytes in der falschen Reihenfolge gespeichert. Bevor ich nun ins Dezimalsystem umrechne, korrigiere ich die Reihenfolge:

$\mathbf{4c\ 02\ 00\ 00_{16} \Rightarrow 00\ 00\ 02\ 4c_{16}}$

Nun fehlt noch die Umrechnung ins Dezimalsystem und schon weiss ich:

Die Breite des Bildes beträgt 588 Pixel.

m114-A0G

Marcel Suter

From:
https://wiki.bzz.ch/ - **BZZ - Modulwiki**

Permanent link:
https://wiki.bzz.ch/de/modul/m114/learningunits/lu03/bitmap?rev=1769631167

Last update: **2026/01/28 21:12**