Erläuterungen zu den neuen Funktionen zur Ausgabe von Postscript

PDF-marks etc.

1.3

17. Mai 1999

von

Norbert Karl Hanz

Allgemeines

Damit der pdfmark-Operator nur im Distiller interpretiert wird und von einem Postscript-Interpeter nicht beachtet wird muß folgender Code eingefügt werden:

```
/pdfmark where {pop} {userdict /pdfmark /cleartomark load put} ifelse
/languagelevel where {pop languagelevel}{1} ifelse
2 lt {
    userdict (<<) cvn ([) cvn load put
    userdict (>>) cvn (]) cvn load put
} if


% Einige Definitionen zur Ausrichtung
/Left 0 def
/Center 1 def
/Right 2 def

% Bild ist kein Link
/Image false def

/NImage 0 def     % Kein Link hinter dem Bild
/WImage 1 def     % WebLink hinter dem Bild
/FImage 2 def     % FileLink hinter dem Bild
/LImage 3 def     % Link hinter dem Bild

% Verhindert Fehlermeldungen beim Einbinden von EPS-Dateien
/showpage { } def
```

Wenn `!docinfo [programimage]` benutzt wird muß der Wert auf `true` gesetzt werden, sonst auf `false`.

```
/programmimage true def
```

Wenn `!docinfo [authorimage]` benutzt wird muß der Wert auf `true` gesetzt werden, sonst auf `false`.

```
/authorimage true def
```

Schaltet die Description-Umgebung ab.

```
/descript false def
```

Setzt die Seitenzahl auf 0 und reserviert Platz für die Seitennummer.

```
/pagenumber 0 def
/Seite (Seite    ) def
```

Definitionen für die Fußnoten.

```
/footnumbers 0 def
/footlines 1 def
/strlentest 0 def
/footnotetext 10 array def
/foot false def
```

setBaseColor

Die Funktion `setBaseColor` setzt die Basis-Textfarbe. Da `showpage` die Textfarbe zurücksetzt, muß die Funktion `newpage` entsprechend angepasst werden (s.u.). Wird die Funktion nicht aufgerufen, ist die Textfarbe schwarz.
Die Funktion muß angewandt werden, da die Funktionen für Links darauf zugreifen.
Verwendungszweck, wenn `!doclayout [ps] [textcolor]` gesetzt ist.

```
/setBaseColor          % R G B setBaseColor -
{
  /bcb exch def
  /bcg exch def
  /bcr exch def

  bcr bcg bcb setrgbcolor
} bind def
```

Dokument-Info

Die Dokumentinformation kann in diese Struktur eingetragen werden. der Einfachheit halber schlage ich vor, diese direkt in den pdfmark einzutragen. Die Einträge sind hoffentlich selbsterklärend.

```
[ /Title (Projekt Udo2Postscript2PDF)
    /Author (Norbert Hanz)
    /Subject (Erläuterungen zu Postscript)
    /Keywords (Begriff1 Begriff2 ... BegriffN)
    /Creator (UDO 6 Patchlevel 12 for Win32)
    /ModDate (D:19990421130500)
  /DOCINFO pdfmark
```

HowToOpen

Die Funktion `HowToOpen` legt fest, wie die PDF geöffnet werden soll.
Der Wert **Modus** gibt an, wie die Datei geöffnet werden soll. Folgende Möglichkeiten bestehen:

- `/UseNone` Das Dokument wird ohne Lesezeichen und Thumbnails dargestellt
- `/UseOutlines` Das Dokument wird mit Lesezeichen angezeigt.
- `/UseThumbs` Das Dokument wird mit Thumbnails angezeigt (für UDO wohl derzeit nicht brauchbar)
- `/FullScreen` Das Dokument wird im Vollbildmodus angezeigt.

Der Wert **Seite** gibt an, welche Seite beim Öffnen angezeigt werden soll, wenn die Datei z.B. über den Desktop und nicht über einen Link mit Anker aufgerufen wird.
Der Wert **Toolbar** gibt an ob die Iconleiste angezeigt wird (`true`) oder nicht (`false`).
Der Wert **Menubar** gibt an ob die Menüleiste angezeigt wird (`true`) oder nicht (`false`).
Der Wert **Anordung** gibt an, wie die Seiten angeordnet werden. Folgende Möglichkeiten bestehen:

- `/SinglePage` Seiten einzeln anzeigen
- `/OneColumn` Seiten spaltenweise anordnen
- `/TwoColumnleft` Seiten zweispaltig anordnen, ungerade Seiten links
- `/TwoColumnRight` Seiten zweispaltig anordnen, ungerade Seiten rechts

```
/HowToOpen       % Modus Seite Toolbar Menubar Anordnung HowToOpen -
{
  /Anordnung exch def
  /Menubar exch def
  /Toolbar exch def
  /Seite exch def
  /Modus exch def
  [ /PageMode Modus /Page Seite
  /DOCVIEW pdfmark

  [{Catalog} << /ViewerPreferences << /HideToolbar Toolbar
  /HideMenubar Menubar >> >>
  /PUT pdfmark

  [{Catalog} << /PageLayout Anordnung >>
  /PUT pdfmark
} bind def
```

Beispiel:

```
/UseOutlines 4 true false /SinglePage HowToOpen
```

Bookmarks

Die Funktion `Bookmarks` dient der Erzeugung der Lesezeichen, die links neben dem Dokument dargestellt werden. Als Ziele werden, wie im Inhaltsverzeichnis die Anker der Named Destinations benutzt.
Der Funktion muß der auszugebende Titel, das Ziel (Anker) und die Anzahl der Unterkapitel übergeben werden. Wenn keine unterkapitel existieren muß `0` übergeben werden. Ist die übergebene Anzahl positiv, werden die Lesezeichen aufgeklappt, ist sie negativ, dann werden sie zugeklappt. Die zugeklappte Version ist m.M.n. zu bevorzugen.
Zuerst müssen jeweils die Nodes, dann die Subnodes etc. definiert werden.

```
/Bookmarks     % Title Ziel Anzahl Bookmarks -
{
  /Anzahl exch def
  /Ziel exch def
  /Title exch def
  [ /Dest Ziel
    /Title Title
    /Count Anzahl
  /OUT pdfmark
} bind def
```

Beispiel:

```
(Aufgaben) /Aufgaben -2 Bookmarks
(Allgemeines) /Allgemeines 0 Bookmarks
(Emissionen) /Emissionen 0 Bookmarks
```

WebLink

Die Funktion `WebLink` dient der Erzeugung von Verknüpfungen, die ins WWW gehen. Der Funktion muß der auszugebende Text, die entsprechende Adresse im Web (z.B. http://, mailto: o.ä.) und die Farbe des Links, entsprechend `!doclayout [ps] [linkcolor]`, übergeben werden.
Für die Farbwerte `R G B` können Werte zwischen `0` und `255` übergeben werden.
Die Funktion stellt selbst fest, ob es sich bei dem Link um einen Text oder ein Bild handelt. Ein Bild hat dann einen Link, wenn bei `imagePost` kein Leerstring als Adresse übergeben wird.
Die Farbe des Textes wird wieder auf die vor dem `WebLink` gültige Farbe eingestellt.

```
/WebLink        % Text Adresse R G B WebLink -
{
  /B exch def
  /G exch def
  /R exch def
  /Adresse exch def
  /Text exch def

  Image          % Ist der Link ein Bild?
  {
    /Tx Text 0 get def
    /Ty Text 1 get def
    /Tw Text 2 get def
    /Th Text 3 get def
    [ /Rect [Tx Ty Tw Th]
      /Border [0 0 0]
      /Action << /Subtype /URI /URI Adresse >>
      /Subtype /Link
    /ANN pdfmark
  }
  {             % Link ist kein Bild
    Text stringwidth pop
    /wx exch def

    /localvar strlen wx add def
    localvar linelen gt
    {
      newline
    } if
    localvar def

    currentpoint
    /y exch def
    /x exch def
    /y y fontsize add def
    /x x wx add def
    [ /Rect [currentpoint x y]
      /Border [0 0 0]
      /Action << /Subtype /URI /URI Adresse >>
      /Subtype /Link
    /ANN pdfmark
    currentrgbcolor
    R G B setrgbcolor
    Uon Text udoshow Uoff
    setrgbcolor
  } ifelse
} bind def
```

Beispiel:

```
(Adresse von Dirk) (http://www.dirk-hagedorn.de) 0 0 255 WebLink
```

FileLink

Die Funktion `FileLink` dient zur Erzeugung von Verweisen zu anderen PDF-Dateien. Der Funktion muß der auszugebende Text, die Zieldatei, die Position innerhalb der Zieldatei (optional) und die Farbe des Verweises, entsprechend `!doclayout [ps] [linkcolor]`, übergeben werden.
Für die Farbwerte `R G B` können Werte zwischen `0` und `255` übergeben werden.
Soll keine bestimmte Position in der Datei angesprungen werden muß für den Wert Anker `/Null` übergeben werden.
Die Farbe des Textes wird wieder auf die vor dem `FileLink` gültige Farbe eingestellt.

```
/FileLink       % Text Datei Anker R G B FileLink -
{
  /B exch def
  /G exch def
  /R exch def
  /Anker exch def
  /Datei exch def
  /Text exch def

  Image          % Ist der Link ein Bild?
  {
    /Tx Text 0 get def
    /Ty Text 1 get def
    /Tw Text 2 get def
    /Th Text 3 get def
    [ /Rect [Tx Ty Tw Th]
      /Border [0 0 0]
      /Dest Anker
      /File Datei
      /Action /GoToR
      /Subtype /Link
    /ANN pdfmark
  }
  {             % Link ist kein Bild
    Text stringwidth pop
    /wx exch def
    /localvar strlen wx add def
    localvar linelen gt
    {
      newline
    } if
    localvar 0 def

    currentpoint
    /y exch def
    /x exch def
    /y y fontsize add def
    /x x wx add def
    [ /Rect [currentpoint x y]
      /Border [0 0 0]
      /Dest Anker
      /File Datei
      /Action /GoToR
      /Subtype /Link
    /ANN pdfmark
    currentrgbcolor
    R G B setrgbcolor
    Uon Text udoshow Uoff
    setrgbcolor
  } ifelse
} bind def
```

Beispiel:

```
(Sprung zu Dest in anderer Datei) (Test.pdf) /Hier 0 0 255 FileLink
```

Link

Die Funktion `Link` dient zur Erzeugung von Verweisen innerhalb einer PDF-Dateie. Der Funktion muß der auszugebende Text, die Position innerhalb der Datei und die Farbe des Verweises, entsprechend `!doclayout [ps] [linkcolor]`, übergeben werden.
Für die Farbwerte `R G B` können Werte zwischen `0` und `255` übergeben werden.
Mit dieser Funktion kann auch gut das Inhaltsverzeichnis erstellt werden. Dafür sollte als Farbe `0 0 0` (schwarz) übergeben werden.
Die Farbe des Textes wird wieder auf die vor dem `Link` gültige Farbe eingestellt.

```
/Link        % Text Anker R G B Link -
{
  /B exch def
  /G exch def
  /R exch def
  /Anker exch def
  /Text exch def

  Image          % Ist der Link ein Bild?
  {
    /Tx Text 0 get def
    /Ty Text 1 get def
    /Tw Text 2 get def
    /Th Text 3 get def
    [ /Rect [Tx Ty Tw Th]
      /Border [0 0 0]
      /Action /GoTo
      /Dest Anker
      /Subtype /Link
    /ANN pdfmark
  }
  {             % Link ist kein Bild
    Text stringwidth pop
    /wx exch def

    /localvar strlen wx add def
    localvar linelen gt
    {
      newline
    } if
    localvar 0 def

    currentpoint
    /y exch def
    /x exch def
    /y y fontsize add def
    /x x wx add def
    [ /Rect [currentpoint x y]
      /Border [0 0 0]
      /Action /GoTo
      /Dest Anker
      /Subtype /Link
    /ANN pdfmark
    currentrgbcolor
    R G B setrgbcolor
    Uon Text udoshow Uoff
    setrgbcolor
  } ifelse
} bind def
```

Beispiel:

```
(Sprung zu den Aufgaben) /Aufgaben 0 0 255 Link
```

NameDest

Mit der Funktion `NameDest` können die sogenannten Named Destinations erzeugt werden. Der Funktion muß nur der Name des Ankers übergeben werden.
Die Funktion sollte für alle Nodes, Subnodes etc. und Alias benutzt werden.

Hinweis: Dafür ist ein eindeutiger Name notwendig, also bietet es sich an auf die gleiche Weise vorzugehen wie bei HTML-Dateien, entweder die automatische Bezeichnung von UDO oder die Auswertung von `!html_name`.

```
/NameDest       % Name NameDest -
{
  /Name exch def
  [ /Dest Name
    /View [ /XYZ null null null ]
  /DEST pdfmark
} bind def
```

Beispiel:

```
!begin_raw
/Aufgaben NameDest
!end_raw
!node Aufgaben
!begin_raw
/Allgemeines NameDest
!end_raw
!subnode Allgemeines
```

Comment

Die Funktion `Comment` dient zur Erzeugung von Notizzetteln auf einer PDF-Seite. Der Funktion muß der auszugebende Text, ein Titel und die Farbe des Icons übergeben werden.
Die Farbwerte können als `1` oder `0` übergeben werden. Es werden offensichtlich nur die Hauptfarben unterstützt.
Das Verhalten ist fest eingestellt, so daß die Notiz immer als Icon erscheint und auf Doppelclick geöffnet wird (kann aber noch geändert werden). Der Kommentar wird am besten vor dem Absatz ausgegeben, neben dem er stehen soll.

```
/Comment        % Text Title R G B Comment -
{
  /B exch def
  /G exch def
  /R exch def
  /Title exch def
  /Text exch def
  currentpoint
  /y exch def
  /x exch def
  /wy y 150 sub def
  /y y fontsize add def
  [ /Rect [ 50 y 200 wy ]
    /Open false
    /Title Title
    /Contents Text
    /Color [R G B]
  /ANN pdfmark
} bind def
```

Beispiel:

```
(Dies ist ein Text, welcher zum Testen eines Kommentars dient!) _
                                          (Hinweis:!) 1 1 0 Comment
```

breite

```
%% Zentrierter Text
%% Es wird nur der auszugebende Text übergeben

/breite      % Text breite -
{
  /Text exch def
  /Mitte rightmargin leftmargin sub 2 div def
  Text stringwidth pop
  /Breite exch def
  /Halb Breite 2 div def
  currentpoint exch pop
  /y exch def
  /x Mitte Halb sub leftmargin add def
  x y moveto
} bind def
```

setAlign

Die Funktion `setAlign` dient zur Ausgabe von ausgerichtetem Text. Der Funktion muß der auszugebende Text und die Ausrichtung (`Left, Center, Right`) übergeben werden.

```
/setAlign      % Text tAlign setAlign -
{
  /tAlign exch def
  /Text exch def
  /Mitte rightmargin leftmargin sub 2 div def
  Text stringwidth pop
  /Width exch def
  /Halb Width 2 div def
  currentpoint exch pop
  /y exch def

  tAlign 2 eq
  {
    /x rightmargin Width sub def
  } if
  tAlign 1 eq
  {
    /x Mitte Halb sub leftmargin add def
  } if
  tAlign 0 eq
  {
    /x x def
  } if

  x y moveto
  Text udoshow
} bind def
```

imageAboutUdo

Die Funktion imageAboutUdo gibt das Bild **udo_mw** aus, inklusive eines Links auf Deine Homepage. Zu diesem Zweck muß die Datei **postimage.ui** eingebunden werden.

```
/imageAboutUdo        % - imageAboutUdo -
{
  /Mitte rightmargin leftmargin sub 2 div leftmargin add def
  currentpoint
  /Y exch def
  /X exch def
  /NY1 Y 31 sub def
  /NX1 Mitte 44 sub def
  /EPSsave save def
  NX1 NY1 translate
  4 4 scale

  [{ImageAboutUdo} /SP pdfmark

  EPSsave restore
  /Image true def
  /NX2 NX1 88 add def
  /NY2 NY1 31 add def
  [NX1 NY1 NX2 NY2] (http://www.dirk-hagedorn.de) 0 0 255 WebLink
  /Image false def
  /acty NY1 6 sub def
  newline
} bind def
```

Beispiel:

imageAboutUdo

imagePre

Die Funktion imagePre wird vor jedem Bild ausgegeben. Der Funktion muß die Breite, Höhe, die Skalierung und die Ausrichtung übergeben werden. Die Funktion sorgt selbst dafür, daß das Bild nicht über das Seitenende lappt.
Den Parameter Sc muß der Benutzer setzen können, damit die Bilder skaliert werden können.

```
/imagePre       % Width Height Sc iAlign imagePre -
{
  /iAlign exch def
  /Sc exch def
  /Height exch 4 Sc div div def
  /Width exch 4 Sc div div def

  newline       % Damit die Bilder nicht am Seitenende abgeschnitten werden

  /Mitte rightmargin leftmargin sub 2 div leftmargin add def
  currentpoint
  /Y exch def
  /X exch def

  /Y Y Height sub def
  Y lowermargin lt
  {
    newpage
  } if
  /Y acty def

  /NY1 Y Height sub def

  iAlign 2 eq
  {
    /NX1 rightmargin Width sub def
  } if
  iAlign 1 eq
  {
    /NX1 Mitte Width 2 div sub def
  } if
  iAlign 0 eq
  {
    /NX1 X def
  } if
  /EPSsave save def
  NX1 NY1 translate
  Sc Sc scale
} bind def
```

Beispiel:

siehe imagePost

imagePost

Die Funktion imagePost wird nach einem Bild ausgegeben. Der Funktion muß die Breite und Höhe sowie die Adresse eine Verknüpfung (optional) übergeben werden. Wenn keine Verknüfung vorgesehen ist muß die Adresse als Leerstring übergeben werden.

```
/imagePost      % Width Height Adresse Anker Type imagePost -
{
  /Type exch def
  /Anker exch def
  /Adresse exch def
  /Height exch def
  /Width exch def

  Adresse length 0 ne Anker /Null ne or   % Steckt hinter d. Bild ein Link/Anker?
  {
    /Image true def
    /NX2 NX1 Width add def
    /NY2 NY1 Height add def
    Type 1 eq
    {
      [NX1 NY1 NX2 NY2] Adresse 0 0 255 WebLink
    } if
    Type 2 eq
    {
      [NX1 NY1 NX2 NY2] Adresse Anker 0 0 255 FileLink
    } if
    Type 3 eq
    {
      [NX1 NY1 NX2 NY2] Anker 0 0 255 Link
    } if
    /Image false def
  } if
  /acty NY1 def
  newline
} bind def
```

Beispiel:

```
176 146 4 Center imagePre
(!BILDPATH)/condeal.eps
EPSsave restore
176 146 (http://www.dirk-hagedorn.de) () Wimage imagePost
```

Fußnoten-Übergabe

Die folgenden Funktionen dienen zur Erzeugung von Fußnoten. Die Numerierung startet auf jeder Seite wieder mit 0, die Fußnoten werden unten auf der Seite ausgegeben. Es sind derzeit maximal 9 Fußnoten pro Seite möglich; das kann aber geändert werden, indem das Array `footnotetext` (siehe bei den Änderungen zu `newpage` und `newline`) entsprechend vergrößert wird. Sollte der Benutzer versuchen mehr Fußnoten zu generieren wird dies automatisch von dieser Funktion abgefangen; die Fußnote erscheint allerdings auch nicht im Text.

Der Funktion `footnote` wird der Fußnotentext übergeben. Sie erzeugt automatisch die hochstehende Ziffer im Text.

```
/footnote          % Text footnote -
{
  /localstring ( ) def

  footnumbers 1 add footnotetext length lt
  {
    /footlines 1 def
    /Text exch def
    Text udotest
    /lowermargin lowermargin fontsize footlines 1 add mul add def
    /footnumbers footnumbers 1 add def

    fontsize
    9 changeFontSize
    currentpoint
    /fy exch 5 add def
    fy moveto
    footnumbers localstring cvs udoshow
    currentpoint
    /fy exch 5 sub def
    fy moveto
    changeFontSize
    footnotetext footnumbers Text put
  } if
} bind def
```

Beispiel:

```
... HTML-Seiten) udoshow
(Dies ist eine Fussnote über HTML-Seiten, die eigentlich nichts aussagt
und nur möglichst lang ist, damit der Text umbrechen kann) footnote ( ...
```

Fußnoten-Ausgabe

Die Funktion `footout` gibt die Ziffer und den den Fußnotentext unten auf der Seite aus.

Für den Text wird die Größe des aktuellen Textes benutzt. Sollte am Seitenende oder am Seitenanfang der nächsten Seite gerade eine Überschrift stehen, wird eine zu große Zeichensatzgröße ermittelt. Vielleicht sollte eine feste Größe für Fußnoten benutzt werden?! Dann würde es allerdings nicht mehr zu verschiedenen Zeichensatzgrößen passen (mal in die Zukunft gedacht).

```
/footout           % - footout -
{
  footnumbers 0 gt
  {
    /foot true def
    /counter 0 def
    /localstring (   ) def

    ccleftmargin lowermargin 10 sub moveto
    ccleftmargin 80 add lowermargin 10 sub lineto
    stroke
    ccleftmargin lowermargin 20 sub moveto

    footnumbers
    {
      /counter counter 1 add def
      currentpoint
      /acty exch fontsize 1.5 mul sub def
      pop
      ccleftmargin acty moveto
      fontsize dup
      exch 2 sub changeFontSize
      currentpoint
      /acty exch 4 add def
      acty moveto
      counter localstring cvs udoshow
      currentpoint
      /acty exch 4 sub def
      pop
      /actx ccleftmargin 10 add def
      actx acty moveto
      changeFontSize
      footnotetext counter get udoshow
      /strlen 0 def
    } repeat
  } if
  /foot false def
} bind def
```

Beispiel:

siehe Änderungen zu `newpage`

Fußnoten-Hilfsfunktionen

Die folgenden Funktionen dienen zur Berechnung der Ausgabe des Fußnotentextes ohne ihn wirklich auszugeben.

```
/calcwordlentest        %% PRIVATE!
{
  /wordlentest
  exch
  stringwidth pop def
}
bind def


/wordtest            %% PRIVATE!
{
  exch
  dup
  () ne    %%-- Leere Strings ignorieren...
  {
    /strlentest strlentest wordlentest add def
    strlentest linelen gt
    {
      /footlines footlines 1 add def
      /strlentest wordlentest def
    } if
%    showout     %% -- Wort ausgeben
     pop
  }
  {
    pop  %%-- Leerstring entfernen
  } ifelse

  {             %% Spaceflag auswerten...
%    showout   %% -- Space ausgeben
     pop
    /strlentest strlentest spacewidth add def
  } if
}
bind def

%%--------------------

/udotest            %% PRIVATE!
{
  /linelen rightmargin 90 sub def
  /spacewidth ( ) stringwidth pop def

  {
    ( ) search      %%--- Spaces suchen
    {               %%--- gefunden
      dup           %%--- Wort duplizieren
      calcwordlentest
      true          %%--- 2 x show (mit Space)
      wordtest
    }
    {               %%--- kein Space gefunden
      dup
      calcwordlentest
      false         %%--- 1 x show
      wordtest
      exit
    }
    ifelse
  }
  loop
}
bind def
```

titlepage

Die Funktion titlepage gibt die Titelseite inklusive evtl. benutzter Bilder aus. Der Funktion müssen alle inhaltlichen Angaben übergeben werden; außerdem die Breite, Höhe und Skalierung des Programm-Images und die Breite, Höhe und Skalierung des Autor-Images.
Die Parameter scp und sca muß der Benutzer setzen können, damit die Bilder skaliert werden können und damit die Titelseite nicht auseinandergerissen wird. Das läßt sich leider nur durch Augenschein feststellen.
Der Programmteil wird von oben gesetzt, der Autorenteil von unten. Bei entsprechend großen Bildern können beide Teile aufeinander zu wachsen.

```
/titlepage      % Title Programm Version Date Author Adress1 Adress2 Adress3 _
                                              Adress4 wp hp scp wa ha sca titlepage -
{
  /sca exch def
  /ha exch 4 sca div div def
  /wa exch 4 sca div div def
  /scp exch def
  /hp exch 4 scp div div def
  /wp exch 4 scp div div def
  /Adress4 exch def
  /Adress3 exch def
  /Adress2 exch def
  /Adress1 exch def
  /Author exch def
  /Date exch def
  /Version exch def
  /Programm exch def
  /Title exch def

  /acty acty 50 sub def
  changeBaseFont
  18 changeFontSize
  newline
  Title Center setAlign
  11 changeFontSize

  programmimage
  {
    /Mitte rightmargin leftmargin sub 2 div leftmargin add def
    currentpoint
    /Y exch def
    /X exch def
    /NY Y hp sub def
    /NX Mitte wp 2 div sub def
    /EPSsave save def
    NX NY translate
    scp scp scale

    [{ImageTitleProgramm} /SP pdfmark

    EPSsave restore
    /acty acty hp sub def newline
  }
  {
    30 changeFontSize
    newline
    Bon
    Programm Center setAlign
    Boff
  } ifelse
  newline
  14 changeFontSize
  Version Center setAlign
  newline
```

```
  Date Center setAlign
  hp 0 gt
  {
    /acty lowermargin 150 ha add add def
  }
  {
    /acty acty 200 sub def
  } ifelse
  newline
  11 changeFontSize
  (von) Center setAlign newline

  authorimage
  {
    /Mitte rightmargin leftmargin sub 2 div leftmargin add def
    currentpoint
    /Y exch def
    /X exch def
    /NY Y ha sub def
    /NX Mitte wa 2 div sub def
    /EPSsave save def
    NX NY translate
    sca sca scale

    [{ImageTitleAuthor} /SP pdfmark

    EPSsave restore
    /acty acty ha sub def newline
  }
  {
    Author Center setAlign
  } ifelse
  newline
  Adress1 Center setAlign newline
  Adress2 Center setAlign newline
  Adress3 Center setAlign newline
  Adress4 Center setAlign
  changeBaseFont
} bind def
```

Beispiel:

```
[/BBox [0 0 w h] /_objdef {ImageTitleProgramm} /BP pdfmark
!rinclude program.eps
[/EP pdfmark
[/BBox [0 0 w h] /_objdef {ImageTitleAuthor} /BP pdfmark
!rinclude author.eps
[/EP pdfmark

(CONDEA Chemie GmbH (Werk Meerbeck)) (Gesetzeskataster) (Stand vom)
(26. Januar 1999) (CONDEA Chemie GmbH) (ein Unternehmen der RWE-DEA
Gruppe) (Römerstr. 733) (D-47443 Moers) (Deutschland) 176 146 4 176
146 4 titlepage
```

Wenn Du lieber den Code der Titelseite selbst schreiben willst, mußt Du nur an der Stelle `[{ImageTitleProgramm} /SP pdfmark` die entsprechende EPS-Datei für das Programmbild einbinden und an der Stelle `[{ImageTitleAuthor} /SP pdfmark` das Autorbild.

AboutUdo

So kann die Seite **aboutudo** aussehen, inklusive Bild und Link auf Deine Homepage. Wie das ganze wirklich aussieht, kannst Du auf der nächsten Seite sehen. Das Bild findest Du in der Datei `postimage.ui`; es muß entsprechend im Header eingebunden werden.

```
newpage
18 changeFontSize
newline
Bon
(UDO6) udoshow
Boff
11 changeFontSize
changeBaseFont
newline
(Dieser Text wurde erzeugt mit) Center setAlign
newline
Bon (UDO) Center setAlign Boff newline
(Release 6 Patchlevel 12) Center setAlign newline
(Win32) Center setAlign
newline
(Copyright \251 1995-1999 by) Center setAlign newline
(Dirk Hagedorn) Center setAlign newline
(Postfach 8105) Center setAlign newline
(D-59840 Sundern) Center setAlign newline
(E-Mail: info@dirk-hagedorn.de) Center setAlign
newline
imageAboutUdo
(UDO ist ein Programm, welches Textdateien, die im Universal Document Format) _
                                                    Center setAlign newline (
erstellt wurden, in das ASCII-, ST-Guide-, LaTeX-, Rich Text-, Postscript-,) _
                                                    Center setAlign newline (
Pure-C-Help-, Manualpage-, HTML-, WinHelp-, Texinfo-, Linuxdoc-SGML-,) _
                                                    Center setAlign newline (
LyX-, Apple-QuickView-, PDF-LaTeX, HTML-Help und Turbo-Vision-Help-Format) _
                                                    Center setAlign newline (
umwandeln kann.) Center setAlign
newline
(Weitere Informationen sowie die aktuellen Versionen findet man im World Wide Web _
                                                  unter) Center setAlign newline
(http://www.dirk-hagedorn.de) breite
(http://www.dirk-hagedorn.de) (http://www.dirk-hagedorn.de) 0 0 255 WebLink
newline
```

Einen Knackpunkt bei zentrierten Texten mit Verknüpfungen siehst Du bei der Ausgabe der Adresse deiner Webseite. Zusätzlich zur Ausgabe des WebLinks muß vorher noch die Breite bestimmt werden. Ich habe es aber (noch) nicht geschafft, das mit Hilfe von Postscript auseinanderzunehmen.

Sonstiges

Kleinere "Fehler", die mir aufgefallen sind:

- Bei der itemize-Umgebung wird u.a. ein `/dagger` ausgegeben. Das ist nicht im ISOLatin-Zeichensatz vorhanden sondern nur im PDFEncoding. Besser ist es ein `/minus` zu benutzen. Abhilfe könnte das Umschalten des Encodings schaffen. Das konnte ich allerdings nicht testen, da UDO bei der Benutzung von Makros und Defines Fehler bei der Ausgabe macht (siehe auch unten).

- Die Kommandos !smallskip, !medskip und !bigskip müssen in entsprechend viele `newline` übersetzt werden, die Ausgabe von einfachen Leerzeilen reicht, wie auch bei UDO, nicht.

- Die Angabe für den Seitenanfang ist m.M.n. etwas zu hoch gewählt. `/topmargin 720` bis `/topmargin 750` sähe besser aus.

- Die Ausgabe der Tabellen erzeugt Fehlermeldungen. Die Tabellenzeilen müssen als Strings ausgegeben werden mit entsprechender Ausrichtung und `newline`, wie es im Beispiel (mit rechtsbündiger Ausgabe) zu sehen ist.

```
Von
 (----------+----------+----------) Right setAlign newline
 ( Zelle 11 | Zelle 12 | Zelle 13 ) Right setAlign newline
 (----------+----------+----------) Right setAlign newline
 ( Zelle 21 | Zelle 22 | Zelle 23 ) Right setAlign newline
 (----------+----------+----------) Right setAlign newline
 ( Zelle 31 | Zelle 32 | Zelle 33 ) Right setAlign newline
 (----------+----------+----------) Right setAlign newline
(Tabelle 1: Versuchstabelle) Right setAlign
Voff
```

  Eine bessere Ausgabe mit richtigen Linien ist natürlich noch denkbar.

- Die Benutzung von `changeBaseFont` wird nicht richtig durchgeführt. Mal sind die Überschriften in Helvetica und der Text in Times , mal umgekehrt. Statt den Basis-Zeichensatz immer zu wechseln, würde ich die Fonts besser explizit neu setzen, wenn überhaupt notwendig.

- UDO unterstützt keine List-Umgebungen, obwohl schon alles in Criskers Postscript-Code vorbereitet ist. Der folgende Code-Schnipsel zeigt ein blist-Umgebung.

```
Bon
/offList (Begriff A00) addStrSpaceLeft
Boff
Bon (Begriff A) offList writeBeforeLeft Boff
(Dies ist ein langer, langer, langer, langer, langer, langer, sehr
sehr langer, ein ganz besonders langer Text.) udoshow
newline
Bon (Begriff B) offList writeBeforeLeft Boff
(Dies ist ein kurzer Text.) udoshow
offList subOffFromLeft
```

  Das `newline` macht den Unterschied zwischen !short und nicht !short.

- UDO unterstützt keine Description-Umgebung. Der folgende Code-Schnipsel zeigt zwei verschachtelte Description-Umgebungen.

```
Bon /offList (00000) addStrSpaceLeft Boff
description
Bon (Begriff alle meine ) offList
writeBeforeLeft Boff
(Dies ist ein langer, langer, langer, langer, langer, langer, sehr
sehr langer, ein ganz besonders langer Text.) udoshow
Bon /offList (00000) addStrSpaceLeft Boff
Bon (Begriff alle keine ) offList
writeBeforeLeft
Boff
(Dies ist kein langer, langer, langer, langer, langer, langer,
sehr sehr langer, ein ganz besonders langer Text.) udoshow
offList subOffFromLeft
Bon (Begriff B ) offList
writeBeforeLeft Boff
(Dies ist ein kurzer Text.) udoshow
description
offList subOffFromLeft
```

  Voraussetzung dafür ist eine Änderung in der Funktion `writeBeforeLeft` wie folgt (neuer Code ist grün eingefärbt) ...

```
/writeBeforeLeft    %% In: (String) var
{
  newline
  leftmargin exch sub acty moveto
  dup show

  descript
  {
    calcwordlen
    /strlen wordlen def
  }
  {
    leftmargin acty moveto
  } ifelse
}
bind def
```

  ...und eine Funktion zum Ein- und Ausschalten der Description-Umgebung.

```
/description
{
  descript
  {
    /descript false def
  }
  {
    /descript true def
  } ifelse
} bind def
```

  Es ist bei den Description-Umgebungen zu beachten, daß `description` nur einmal am Anfang und am Ende der äußersten Umgebung stehen dürfen.

- Bei der Enumerate-Umgebung ist ein Blank vor der Ausgabe des Textes zuviel, dadurch ist die erste Zeile nicht bündig mit den folgenden Zeilen (Code der zuviel ist, ist rot eingefärbt).

```
/off2 (000) addStrSpaceLeft
()
(1.) off2 writeBeforeLeft
(_Zust\344ndigkeiten) udoshow (
) udoshow newline
(Wer ist f\374r die Einhaltung dieser) udoshow (
Nebenbestimmung zust\344ndig?) udoshow
```

- Die Funktion `newpage` müßte geändert und erweitert werden, da sonst Teile des Textes innerhalb eines Absatzes zwischen den Seiten verloren gehen und damit eine andere Textfarbe als schwarz auf jeder Seite wieder gesetzt wird. Offensichtlich wird durch `showpage` die gesetzte Farbe wieder gelöscht (Neuer Code ist grün eingefärbt).

```
/newpage           %% In: void
{
  /strlen 0 def
  /acty topmargin fontsize linespacing mul sub def
  /actx leftmargin def
  actx acty moveto
  showpage
  0 255 255 setBaseColor
}
bind def
```

- Bei der Ausgabe von Makros und Defines gibt UDO u.U. zuviele Klammern aus, d.h. UDO kommt, je nachdem was im Makro steht, komplett durcheinander. Bei solchen kleinen Fehlern bricht der Distiller sofort die Übersetzung ab. Grundsätzlich werden auch zuviele `() udoshow` ausgegeben. Folgende Defines habe ich u.a. benutzt:

```
!define yellow ) udoshow 1 1 0 setrgbcolor (
!define navy ) udoshow 0.7 0.7 1 setrgbcolor (

!define coloroff ) udoshow bcr bcg bcb setrgbcolor (
```

  Der String wird abgeschlossen, die Textfarbe geändert und ein neuer Stringanfang eingefügt. Das ganze ist m.M.n. transparent eingefügt. UDO erzeugt folgendes daraus (Code, der zuviel ist, wurde rot eingefäbt):

```
newline
()
/bullet off1 writeBulletLeft
( Text1 ) udoshow 1 1 0 setrgbcolor () udoshow (
yellow ) udoshow bcr bcg bcb setrgbcolor ( Text2) udoshow
newline
()
/bullet off1 writeBulletLeft
( Text1 ) udoshow 0.7 0.7 1 setrgbcolor) udoshow ((
navy ) udoshow bcr bcg bcb setrgbcolor ( Text2) udoshow
newline
```

  Der erste Eintrag ist eigentlich nur ein Kosmetikproblem, beim zweiten Eintrag liegt ein Fehler vor. Dieser Fehler mit den Klammern tritt nur auf, wenn für die RGB-Werte mehrere Fließkommazahlen (0.7) benutzt werden.

- Es werden zu viele `udoshow` ausgegeben. Normalerweise reicht ein `udoshow` pro Absatz, wenn nicht Links auftauchen. Durch das häufige Setzen von `udoshow` kommt es nämlich beim Einsatz von Klammern durch den Benutzer zu Fehlern, weil Strings nicht abgeschlossen werden. Das führt im besten Fall dazu, daß Postscript-Quellcode im Text steht, schlimmstenfalls zum Abbruch der Übersetzung.

- Die Funktion `newpage` könnte auch für Kopf- und Fußzeilen sowie Fußnoten erweitert werden.

```
/pagenumber 0 def
/Seite (Seite    ) def
/footnumbers 0 def
/footlines 1 def
/strlentest 0 def
/footnotetext 10 array def
/foot false def
```

  Die oben stehenden Definitionen können irgendwo am Anfang stehen. Das Wort **Seite** kann durch das in der jeweiligen Zielsprache übliche Wort ersetzt werden. Der Platz im String reicht für 999 Seiten; vielleicht sind vier Stellen doch besser. Die Koordinaten müssen noch genauer bestimmt werden, entsprechend muß auch `/topmargin` und `/lowermargin` angepasst werden (Neuer Code ist grün eingefärbt).

```
/newpage           %% In: void
{
  footout
  /footnumbers 0 def
  /lowermargin cclowermargin def

  ccleftmargin cclowermargin 30 sub moveto
  (Text für eine Fußzeile) udoshow

  /strlen 0 def
  /acty topmargin fontsize linespacing mul sub def
  /actx leftmargin def
  actx acty moveto

  ccleftmargin cctopmargin moveto
  /pagenumber pagenumber 1 add def
  /Seite (Seite    ) def
  Seite 6 pagenumber (xx) cvs putinterval
  (Titeltext Programmtext) udoshow Seite Right setAlign

  showpage
}
bind def
```

In diesem Zusammenhang müßte auch die Funktion `newline` erweitert werden.

```
/newline            %% In: void
{
  /strlen 0 def                %% akt. Ausgabelaenge = 0
  foot
  {
    /actx ccleftmargin 10 add def
    /acty acty fontsize sub def
  }
  {
    /actx leftmargin def
    /acty acty fontsize linespacing mul sub def
    acty lowermargin lt
    {
      newpage
    }
    if
  } ifelse
  actx acty moveto
}
bind def
```

- So könnte der Anfang eine Textdatei aussehen:

```
[ /Title (Projekt Seitz)
    /Author (Norbert Hanz)
    /Subject (Genehmigungsdatenbank)
    /Keywords (Seitz Hanz)
    /Creator (UDO 6 Patchlevel 12 for Win32)
    /ModDate (D:19990406130500)
  /DOCINFO pdfmark

/UseOutlines 4 true false /SinglePage HowToOpen

(Aufgaben) /Aufgaben -2 Bookmarks
(Allgemeines) /Allgemeines 0 Bookmarks
(Emissionen) /Emissionen 0 Bookmarks
(Status) /Status -0 Bookmarks

/showpage { } def
0 255 255 setBaseColor

/programmimage true def
/authorimage true def

/cctopmargin    780 def
/cclowermargin   72 def
/ccrightmargin  540 def
/ccleftmargin    90 def
```

Die unveränderlich Seitenränder (`cc*`) müssen festgelegt werden. Der exakte Wert für den oberen Seitenrand (`cctopmargin`) müßte in Zusammenhang mit `topmargin` noch festgelegt werden.

Der Eintrag `/showpage { } def` muß vorhanden sein.

History

27/04/1999

- Fehler bei den drei Verweisfunktionen beseitigt; der Link wurde nicht mit umgebrochen, wenn der auszugebende Text nicht mehr in die Zeile passte.

28/04/1999

- Fehler bei der Enumerate-Umgebung
- Hinweis für NameDest

14/05/1999

- Ausgabe von Fußnoten
- Ausgabe von Kopf- und Fußzeilen
- Hinweis für die Description-Umgebung

17/05/1999

- Links zu anderen PDF-Dateien und Links innerhalb von PDF-Dateien können jetzt auch als Bild dargestellt werden
- Der EPS-Code des AboutUDO-Images ist jetzt Teil der Datei **`postscript.ui`**