Dokumentation von HTWself

Vorwort

Das Projekt ist etwas größer geworden als anfangs gedacht. Daher diese Dokumentation, um zum einen Ihnen eine bessere Übersicht zu geben, zum anderen jedoch auch, um diese Klasse für nachfolgende Dokumentationen zu nutzen.
Die Aufgabe, anlässlich der ich diese Klasse für den Kurs *LaTeX für Fortgeschrittene* erstellt habe, schrieb vor zwei Optionen, <exam> und <cv> zu erstellen. Mit <exam> sollte ein Schachbrett mit variierender Farbtiefe der schwarzen Felder, und mit <cv> ein Lebenslauf erstellt werden. Zusätzlich sollte mit jeder Option jeweils eine Minimalversion eines Dokumentencovers erstellt und ausgegeben werden.

Es ist dann doch noch etwas mehr geworden als nur von der Aufgabe vorgegeben. Es wurde in dem Kurs zB sehr stark pstricks thematisiert. Damit hatte man auf einmal ein großes Repetoir zur Verfügung, mit dem man das Dokument um einiges aufwerten kann. Daher habe ich ein bisschen mit Boxen und Überschriften experimentiert und versucht pstricks damit zu vereinigen. Auch multido gab mir interessante Möglichkeiten, verschiedene Aspekte der Klasse zu automatisieren.

Ich hoffe Sie haben mit der Klasse etwas Spaß und fühlen sich nicht komplett erschlagen. Ich habe nach Möglichkeit versucht alles so verständlich wie nur möglich zu schreiben, doch am Ende fehlte doch vllt etwas die Zeit den Code noch weiter aufzuräumen. Alle Operationen sollten zB auch ohne das xifthen- package laufen, welches nicht mit moderncv kompatibel ist. Insoweit könnte man dieses Paket aus der Klasse raus nehmen. Aber dies schränkt die Funktionaltät in keinster Weise ein, sodass alles funktionieren sollte.

Klassen-Optionen

Bei beiden Optionen gilt, dass mit xelatex kompiliert werden muss. Beide Optionen bedienen sich von den Optionen dessen zugrunde liegenden Klassen (siehe Beschreibung).

Die Option cv ist jetzt essentiell eine Bewerbungsmappe geworden. Dies war zwar in der Form von der Aufgabe nicht gefordert, jedoch wüsste ich nicht wo sonst man ein Deckblatt für einen Lebenslauf benötigt.

Die Option exam ist quasi die Option, welche ich wählen würde, um wissenschaftliche Texte zu erstellen.

cv

Diese Option erstellt eine Bewerbungsmappe. Mit dieser Option wird der Befehl \HTWCV erstellt, mit dem man einen Lebenslauf kompilieren lassen kann. Es werden alle Optionen und Eigenschaften aus der Klasse ”moderncv”übernommen. Es sei angemerkt, dass ein Photo nicht notwendig ist. Ich rate jedoch davon ab kein Photo in eine Bewerbungsmappe einzufügen, da dies meines Erachtens keinen guten Eindruck macht.

exam

Diese Option wurde mit dem Gedanken erstellt sie für Protokolle, Arbeiten und Publikationen zu nutzen. In ihr wird der Befehl \HTWChecker erstellt, mit dem man das Schachbrett erstellen kann. Es werden alle Optionen und Eigenschaften aus der Klasse ”labbook”übernommen.

Achtung: Diese Option benutzt das Paket ”currfile”, dass aus dem recorder-file (.fls) den Pfad der Tex-Datei liest. Dies wird dazu genutzt, um den Schriftsatz der HTW zu finden. Daher ist es erforderlich in den Optionen zum kompilieren für xelatex -recorder hinzuzufügen.

Commands

In nicht alphabetischer Reihenfolge. (sorry!)

\HTWChecker {<startcolor>} {<endcolor>}{<xsize>} {<ysize>}{<border>}

Erstellt ein Schachbrett mit (<xsize> x <ysize>) Feldern. Das linke untere Feld hat die Farbe <startcolor>, das rechte obere Feld <endcolor>, die Felder dazwischen werden linear interpoliert. Nur die schwarzen Felder werden gefärbt, die weißen Felder bleiben durchgängig weiß. Sobald <border> nicht leer ist wird ein Rand mit ausgegeben.

Das Schachbrett wird mit multido-Schleifen generiert und ist so ausgelegt, dass jede Feldgröße, auch ungerade Zahlen, eingestellt werden kann. Aufgrund der Beschriftung durch Buchstaben kann das Feld nur eine maximale Länge von 28 Feldern haben.

\HTWCV

Erstellt mit allen angegebenen Informationen ein Minimalbeispiel eines Lebenslaufs.

\HTWTitle {<Wert>}

Setzt den Titel des Dokumentes auf <Wert>.

\HTWSubtitle {<Wert>}

Setzt bei exam den Untertitel und bei cv den Text vor dem Beruf auf <Wert>.

\HTWCover

Erstellt passend zu \HTWLastpage das Deckblatt mit Hilfe der angegebenen Informationen.

\HTWLastpage

Erstellt passend zu \HTWCover die letzte Seite für das Dokument.

\HTWIssueDate {<Wert>}

Setzt das Datum des Dokumentes. Default ist das aktuelle Datum.

\HTWReference {<Wert>}

Nur bei exam. Gibt dem Dokument eine Referenz, welche im Deckblatt ausgegeben wird.

\HTWCourse {<Wert>}	Nur bei exam. Bestimmt im Dokument den Semesterkurs, in dem dieses Dokument erstellt wird.
\HTWJobname {<Wert>}	Nur bei cv. Gibt dem Dokument den Namen des Berufs oder der Stelle, bei der man sich bewirbt.
\HTWSignature {<Scale>}{<Pfad>}	Nur bei cv. Setzt die Größe \signatureScale der Unterschrift mit <Scale> und den Pfad \signature mit <Pfad>.
\HTWStudent {<firstname>} {<lastname>} {<matrikel>}{<mail>}	Nur bei exam. Fügt dem Dokument ein Autor hinzu. Es muss entweder <firstname> oder <lastname> angegeben werden. Alle anderen Eingabewerte können auch ausgelassen werden. Mit dem Wert <matrikel> wird die Matrikelnummer angegeben (bitte mit s0... angeben), mit <mail> kann für Kontaktdaten eine Mailadresse angegeben werden. Die Mail wird auf der letzten Seite, alle anderen Sachen auf dem Deckblatt angegeben. Der Befehl besitzt noch gesonderte Spezialfälle: Wird als Mailadresse ein einzelner Buchstabe angegeben, dann wird die Adresse ausgelassen. Wenn eine Matrikelnummer angegeben wird, aber keine Mailadresse, dann wird aus der Matrikelnummer automatisch eine Adresse generiert. Dies kann mit einem Buchstaben als <mail> unterdrückt werden.
\Catcolor	Die Farbe, die der Klassen-Option zugeordnet ist, ausgegeben in \color{<Farbe}. Für exam ist es das HTWGrün, während es für cv das HTWBlau ist.
\type	Die Farbe, die der Klassen-Option zugeordnet ist. Für exam ist es das HTWGrün, während es für cv das HTWBlau ist.
\HTWmakecvtitle	Nur bei cv. Verhält sich so wie \makecvtitle mit dem einzigen Unterschied, dass auch ein Titel ohne Photo erstellt werden kann.

\HTWAttachment {<name>}{<pfad>} {<typ>}	Nur bei cv. Fügt dem Dokument einen Anhang hinzu, welcher mit \HTWIncludeAttachments bzw mit \HTWIncludeAttachment in das Dokument eingefügt werden kann. Es muss <name> angegeben werden. Alle anderen Eingabewerte können auch ausgelassen werden. Mit dem Wert <pfad> wird der Dateipfad von der tex-Datei aus angegeben, mit <typ> muss im Falle eines Pfades die Art des Anhanges angegeben werden. Dazu stehen zur Auswahl: pdf, tex, png, jpg. Wird kein Pfad oder kein Typ angegeben, dann wird der Anhang von \HTWIncludeAttachments und \HTWIncludeAttachment nicht ausgegeben.
\HTWIncludeAttachments	Nur bei cv. Fügt alle angegebenen Anhänge dem Dokument hinzu.
\HTWIncludeAttachmentByPath{<Pfad>} {<Typ>}	Nur bei cv. Fügt den Anhang mit dem Dateipfad <Pfad> und dem Typ <Typ> dem Dokument hinzu. Es gibt die Typen pdf, tex, png, jpg.
\HTWIncludeAttachmentByName{<name>}	Nur bei cv. Fügt den Anhang mit dem Namen <name> dem Dokument hinzu.
\HTWIncludeAttachment{<name>}	Nur bei cv. Siehe \HTWIncludeAttachmentByName
\HTWTexAttachmentTag{<tagname>}	Nur bei cv. Gibt dem Dokument den tagname an, der benutzt wird um den Teil von tex-Dateien als Anhang zu laden, der zwischen %<*tagname> und %</tagname> steht. Wird der Tag nicht gefunden, dann wird die komplette Datei geladen. Default ist undefiniert.
\HTWAttachmentHeight{<Höhe>}	Nur bei cv. Setzt die Höhe aller als Anhang eingefügter Bilder in Abhängigkeit zu \paperheight auf <Höhe>.
\HTWAttachmentAngle{<Winkel>}	Nur bei cv. Setzt die Rotation in Grad aller als Anhang eingefügter Bilder auf <Winkel>.

\HTWListsOfx	Nur bei exam. Fügt ein Inhaltsverzeichnis der Gliederungstiefe 1 hinzu.
\getLetter {<Wert>}	Nur bei exam. Gibt das ASCII-Zeichen von <Wert> aus. Dieser Befehl wird für den Rand des Schachfeldes benutzt.
\help {<Befehl>}	Lädt die Definition von <Befehl> aus dieser Dokumentation.
\h	Lädt dieses Dokument.
\createHTWbox {<Weite>}{<Inhalt>}	Erstellt eine unsichtbare Box der Textweite <Weite> mit dem Inhalt <Inhalt>. Dieser Befehl bildet die Grundlage aller Boxen.
\HTWInfobox {<Weite>}{<Inhalt>} {<Titel>}	Nur bei exam. Erstellt eine eingerahmte Box der Textweite <Weite> mit dem Inhalt <Inhalt>. Optional kann mit <Titel> ein Titel in der oberen rechten Ecke der Box dargestellt werden. Ein `i` in der oberen linken Ecke zeigt, dass es sich um eine Infobox handelt.
\HTWQuotebox {<Weite>}{<Inhalt>} {<Titel>}	Nur bei exam. Erstellt eine teilweise eingerahmte Box der Textweite <Weite> mit dem Inhalt <Inhalt>. Optional kann mit <Titel> ein Titel in der oberen rechten Ecke der Box dargestellt werden.
\HTWBracketbox {<Weite>}{<Inhalt>}	Nur bei exam. Erstellt eine durch Klammern eingerahmte Box der Textweite <Weite> mit dem Inhalt <Inhalt>.
\HTWHBracketbox {<Weite>}{<Inhalt>}	Siehe \HTWBracketbox.
\HTWFramebox {<Weite>}{<Inhalt>} {<Titel>}	Nur bei exam. Erstellt eine an den Ecken eingerahmte Box der Textweite <Weite> mit dem Inhalt <Inhalt>.

\HTWVBracketbox {<Weite>}{<Inhalt>} {<Titel>}	Nur bei exam. Erstellt eine durch horizontale Klammern eingerahmte Box der Textweite <Weite> mit dem Inhalt <Inhalt>. Optional kann mit <Titel> ein Titel im Zentrum der Klammern dargestellt werden.
\HTWSpeechbubble-box{<Weite>}{<Inhalt>} {<Titel>}{<Richtung>}	Nur bei exam. Erstellt eine Box in Form einer Sprechblase mit der Textweite <Weite> mit dem Inhalt <Inhalt>. Optional kann mit <Titel> ein Titel dargestellt werden. Wenn <Richtung> gleich l ist, dann ist die Sprechblase nach links, ansonsten nach rechts ausgerichtet.
\begin{HTWBulletList} ...\end{HTWBulletList}	Eine Listen-Umgebung, in der der Befehl \item dem Farbschema passende Quadrate sind.
\begin{HTWdefList} ...\end{HTWdefList}	Eine Listen-Umgebung, in der der Befehl \item<name> dem Farbschema passende kleine Teilüberschriften sind.

HTW-Packages

In der Klasse HTWself werden zwei kleine packages verwendet, die ich selber geschrieben habe: HTWMath und HTWColor.

HTWMath

HTWMath lädt das package \xifthen und \fp. Es werden folgende Komparatoren hinzugefügt:

\isBigger{<Wert1>}{<Wert2>}	Wert1 > Wert2
\isSmaller{<Wert1>}{<Wert2>}	Wer1 < Wert2
\isZero{<Wert1>}	Wert1 = 0
\isOne{<Wert1>}	Wert1 = 1
\isNegative{<Wert1>}	Wert1 < 0
\isInRange{<Wert>}{<Min>}{<Max>}	Wert1 > Min AND Wert1 < Max

Folgende Operatoren werden durch HTWMath hinzugefügt:

\FPmod{<Wert1>}{<Wert2>}	Wert1 modulo Wert2
\FPinc{<Wert1>}	Wert1 := Wert1 + 1
\FPdec{<Wert1>}	Wert1 := Wert1 - 1

HTWColor

HTWColor lädt das package \xcolor mit der Option table. Die zusätzliche Option ist für die Klasse moderncv notwendig. In HTWColor werden die Farben der HTW Berlin definiert: HTWGreen ist die Hauptfarbe, HTWBlue steht für Service-Einrichtungen, HTWOrange für Infrasruktur und HTWGray existiert, um die Farbpalette abzurunden.

Farben sind folgend benannt: **HTW<color><intensity>**
zB HTWGreen100 ist HTWGreen in voller Intensität. HTWGreen50 ist HTWGreen mit halber Intensität.Die Farben sind zudem in 5er-Schritten definiert. Unzulässig ist zB HTWGreen52.
Es gibt des weiteren noch eine dunkle Variante der Farbe mit 'Dark' statt der Intensität.zB HTWGreenDark ist dunkelgrün.

main.m

```
1  // SodokuSolver.cpp : Diese Datei enthält die Funktion "main". Hier
       beginnt und endet die Ausführung des Programms.
2  //
3
4  #include "pch.h"
5  #include <stdint.h>
6  #include <stdio.h>
7  #include <conio.h>
8  #include <windows.h>
9
10 #include "drawings.h"
11 #include "sodokuSolver.h"
12
13 void gotoxy(short x, short y) //cursor-positioner
14 {
15   COORD pos = { x,y };
16   SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), pos);
17 }
18
19 COORD getxy() {
20   CONSOLE_SCREEN_BUFFER_INFO screenBufferInfo;
21   HANDLE hStd = GetStdHandle(STD_OUTPUT_HANDLE);
22   if (!GetConsoleScreenBufferInfo(hStd, &screenBufferInfo))
23     printf("GetConsoleScreenBufferInfo (%d)\n", GetLastError());
24   return screenBufferInfo.dwCursorPosition;
25 }
26
27 COORD add(COORD x, COORD y) { return {x.X + y.X, x.Y + y.Y}; }
28 COORD move(COORD pos, short steps) { return { (pos.X + steps) % 9, pos.
       Y + (pos.X + steps) / 9 }; }
29
30 COORD getInput(COORD pos, char *input)
31 {
32   COORD dir = {0,0};
33   uint8_t in;
34
35   //positioniere cursor:
36   pos.X = (pos.X+1) * 2 + 2 * (pos.X/3);
37   pos.Y = 4 + pos.Y + (pos.Y / 3);
38   gotoxy(pos.X, pos.Y);
39
40   in = _getch();
41
42   if (in == 224) {
43
```

main.m

main.m

```
44      in = _getch();
45      switch (in)
46      {
47      case 75: dir = { -1,0 }; break;//left
48      case 72: dir = { 0,-1 }; break;//up
49      case 77: dir = { 1,0 }; break;//right
50      case 80: dir = { 0,1 }; break;//down
51      }
52      *input = '\x00';
53    }
54    else { *input = in; }
55    return dir;
56  }
57
58  int main()
59  {
60    //Erstelle Eingabefeld:
61    system("CLS");//clear screen
62    printf("Sodoku- Solver v0.1\n\n");
63
64    printf("Bitte gebe die Werte des Sodokus ein:\n");
65    drawFrame();
66
67    //*
68    //Eingabe der vorgegebenen Werte:
69    char in;
70    COORD pos;
71    COORD pos_old;
72    initSodoku();
73
74    while (true)
75    {
76      pos = { 0,0 };
77      //Eintragezyklus:
78      do
79      {
80        pos = add(getInput(pos, &in), pos);
81        if (in > 48 && in <= 57) { //Zahl eingegeben
82          setSodoku(pos, emptyField | (in - 48));
83          printf("%hhu", in-48);
84          if (pos.X != 8 || pos.Y != 8) pos = move(pos, 1);
85        }
86        else if (in == 32 || in == 48) { _putch(' '); setSodoku(pos,
      emptyField); pos = move(pos, 1); }//Space oder 0
87        else if (in == 8) { _putch(' '); setSodoku(pos, emptyField);
      continue; }//Backspace
```

main.m

```
88        else if (in == 27 || in == 13) break;//Escape oder Enter ->
      Abbruch
89
90        if (pos.X > 8) pos.X = 0;
91        if (pos.X < 0) pos.X = 8;
92        if (pos.Y > 8) pos.Y = 0;
93        if (pos.Y < 0) pos.Y = 8;
94      }while (true);
95
96      //Abfrage, ob so korrekt:
97      gotoxy(0, 16);
98      if (in == 27) {
99        printf("Programm abbrechen? (y/n)");
100       in = _getch(); _putch(in);
101       if (in == 'y' || in == 'Y') return 0;
102     }
103     else {
104       printf("Eingabe korrekt? (y/n)");
105       in = _getch();
106       _putch(in);
107       if (in == 'y' || in == 'Y') break;
108     }
109
110     gotoxy(0, 16);
111     printf("                          ");
112   }
113   printf("\n\nStarte Auswertung...\n");
114
115   int res = solveSodoku();
116
117   //Lösung des Problems:
118   switch (res)
119   {
120   case 1: printf("Vorgang wurde abgebrochen!"); break;
121   case 2: printf("Es konnte keine eindeutige L\x94sung gefunden werden
        .\nWurde vielleicht eine Zahl vergessen?"); break;
122   case 3: printf("Widerspruch gefunden: Das Sodoku ist nicht l\x94sbar
        !\nWurde vielleicht eine Zahl falsch eingegeben?"); break;
123   default:
124     printf("L\x94sung ermittelt:\n");
125     pos_old = getxy();
126     drawFrame();
127     //Fülle das Sodoku aus (Zeile 21):
128
129     for (int x = 0; x < 9; x++) {
130       for (int y = 0; y < 9; y++) {
131         //positioniere cursor:
```