

DAS TUCSEATING-PAKET

VO.4.0 2025-07-23

Erstellung von Sitzplänen

Matthias WERNER¹

<https://github.com/tuc-osg/tucseating>

Das Paket ermöglicht es, Sitzpläne, wie sie z. B. für Prüfungen benötigt werden, einfach zu erstellen. Eine Reihe verschiedener automatischer Platzierungsschemata sind vordefiniert, aber man kann auch feingranular eigene Platzierungen vornehmen. Während das Paket zunächst für den internen Gebrauch an der TU Chemnitz gedacht war und für (einige) der Räume der TU Chemnitz vordefinierte Sitzlayouts enthält, sind jedoch einerseits sowohl die Raumdaten leicht erweiter- oder ersetzbar, andererseits können Räume auch ad hoc erstellt werden.

Inhaltsverzeichnis

1 Einführung	2	4.4 Sitzplatzliste	9
2 Abhängigkeiten	2	5 Beispiele	9
3 Sitzlayout	2	5.1 Vordefinierter Raum, rechteckig	9
3.1 Klassenoptionen	2	5.2 Vordefinierter Raum, bo- genförmig	10
3.2 Späte Optionswahl	4	5.3 Eigenes Platzlayout und Sitz- schema	11
3.3 Modifikation des Sitzlayouts .	4	5.4 Dame	12
3.4 Ausgabe	5		
4 Sitzplatzbelegung	6	6 Deklaration neuer Raumlayouts	13
4.1 Parameter zur Konstruktion von Belegungsschemata	6	7 Beschränkungen und Bugs	14
4.2 Vordefinierte Sitzschemata . .	7	8 Lizenz	15
4.3 Sitzbeschriftung	8		

¹ matthias.werner@informatik.tu-chemnitz.de

1 Einführung

Für die Durchführung von Prüfungen benötigen wir mitunter Sitzpläne. So haben sich über die einige mit Pläne in Form von TikZ-unterstützten \LaTeX -Dateien angesammelt. Je nach Anzahl der Studierenden in einer Prüfung (und für wie groß wir die Gefahr eines Betrugsversuches bewerten) nutzen wir unterschiedliche Platzierungsschemata, so dass wir die Dateien in anpassen müssen. Außerdem wird uns von Zeit zu Zeit ein neuer Raum zugewiesen, für den wir noch keine Pläne haben.

Dies war die Motivation zur Erschaffung des tucseating-Pakets. Es ...

- ermöglicht eine schnelle und einfache Erstellung von Sitzplänen;
- trennt das Sitzlayout und das Platzierungsschema voneinander;
- bietet eine Reihe von Standardschemata für die Platzierung an;
- enthält bereits eine Anzahl vordefinierter Räume mit Layouts der Sitze;
- erlaubt eine *Ad-hoc*-Erstellung neuer Räume und Platzierungsschemata.

2 Abhängigkeiten

Das tucseating-Paket arbeitet nur mit Lua \LaTeX und benötigt eine hinreichend moderne \LaTeX -Version, mindestens vom Juli 2022. Es lädt folgende Pakete:

- etoolbox
- luacode
- tikz

Diese Pakete sind in allen gängigen \TeX -Distributionen vorhanden und haben wiederum andere Pakete als Abhängigkeit. Insbesondere wird durch tikz das Paket xcolor geladen, dessen Farbdefinition auch in tucseating verwendet werden.

3 Sitzlayout

Das Paket wird wie üblich mit `\usepackage[optionen]{tucseating}` geladen. Dabei kann bereits weitgehend das Layout der Sitze festgelegt werden, also die Darstellung der Sitze im Raum. Die Zuweisung der Sitzbelegung erfolgt später, siehe Abschnitt 4.

3.1 Klassenoptionen

Durch die Nutzung bzw. Nichtnutzung der Option `room` werden zwei grundsätzliche Anwendungsfälle unterschieden:

`room = Raum`

Voreinstellung:

Diesen Optionsschlüssel sollte man nutzen, wenn der gewünschte Raum ist bereits in der Datenbank von tucseating vorhanden ist.

`room file = {Dateiname}`

Voreinstellung: `rooms1`

Die Daten für die vordefinierten Räume werden aus `Dateiname.tsr` gelesen. Dieser Schlüssel kann genutzt werden, wenn man eigene Räume anlegt, siehe Abschnitt 6.

3 Sitzlayout

Falls der Raum noch nicht bekannt ist, werden einige Schlüssel-Wert-Paar zur Beschreibung des Raumlaysout gebraucht.

`shape = rectangle|arc` Voreinstellung: rectangle
Hier wird beschrieben, ob das Layout der Sitze rechteckig (rectangle) oder bogenförmig (arc)² ist.

`rows = \langle Anzahl der Sitzreihen \rangle` (erforderlich)

`seats per row = \langle Sitze pro Sitzreihe \rangle` (erforderlich)

Besimmt Anzahl der Sitzreihen und Sitze pro Reihe. Dabei muss immer von der jeweils maximalen Anzahl ausgegangen werden. Für unvollständige Reihen werden später Sitze entfernt, aber es können keine Sitze mehr zugefügt werden, die außerhalb des einmal festgelegten Layoutrahmens liegen. **Achtung:** Auch Gänge zwischen Blöcken von Sitzen müssen hier als Sitze mitgezählt werden.

Wenn Sie bei den Klassenoptionen **sowohl** den Schlüssel `room`, **als auch** eine der Schlüssel `rows` oder `seats per row` angeben, ist das Ergebnis unbestimmt. Entsprechend wird eine Warnung ausgegeben.

Alle weiteren Klassenoptionen legen die Darstellung des Raumlaysouts fest. Sie können auch später mit `\tucsConfig` gesetzt werden, vgl. Abschnitt 3.2.

`blackboard = true|false` Voreinstellung: false
Zeichnet eine Tafel ein.

`seat distance = \langle Abstand \rangle` Voreinstellung: 2pt
Abstand der Sitze zueinander. Damit wird auch der Reihenabstand bestimmt. Möchte man beides unterschiedlich haben, so muss man die beiden folgenden Schlüssel nutzen:

`seat neighbor distance = \langle Abstand \rangle` Voreinstellung: 2pt

`row distance = \langle Abstand \rangle` Voreinstellung: 2pt

`rownumbers = none|left|right|both` Voreinstellung: none
Legt beim Rechteck-Layout fest, ob links und oder rechts der Reihen die Nummer der Sitzreihe angegeben wird. Dabei wird von vorn (Tafel) gezählt.

Anmerkung: Für das bogenförmige Layout ist diese Funktion derzeit nicht implementiert.

`rownumber distance = \langle Abstand \rangle` Voreinstellung: 2pt
Abstand der Reihennummern zu den äußeren Sitzen, wenn `rownumbers` nicht none ist.

`empty seat background color = \langle Farbe \rangle` Voreinstellung: lightgray!20

`empty seat border color = \langle Farbe \rangle` Voreinstellung: lightgray

`assigned seat background color = \langle Farbe \rangle` Voreinstellung: lightgray!30

2. Z. B. an der TU Chemnitz der Raum A10.316.

`assigned seat border color = <Farbe>` Voreinstellung: `black`
 Legt die Farben für den Hintergrund und die Umrandung von leeren bzw. durch ein Sitzschema belegten Sitzen fest. Falls die Sitze nicht farblich unterschieden werden sollen, können auch die Schlüssel

`seat background color = <Farbe>` (zunächst leer)

`seat border color = <Farbe>` (zunächst leer)
 verwendet werden.

`assigned seat label font = {<Fontbefehl>}` Voreinstellung: `\small`

`assigned seat label color = <Farbe>` Voreinstellung: `black`
 Setzt die Größe und Farbe der Sitzbeschriftung.

3.2 Späte Optionswahl

`\tucsConfig`

Mit diesem Kommando können außer `room`, `shape`, `rows` und `seats per row` alle im Abschnitt 3.1 genannten Schlüssel auch außerhalb von `\documentclass`-Optionen gesetzt werden.

3.3 Modifikation des Sitzlayouts

Häufig ist nicht in jeder Reihe alle Sitze vorhanden. Bei Angabe eines Raums ist dies bereits berücksichtigt, aber auch hier kann es vorkommen, dass beispielsweise ein Klappsitz defekt ist und aus dem Layout genommen werden muss. Beim Erstellen eines eigenen Layouts ist die Modifikation die Regel. Dafür stellt folgende Kommandos zur Verfügung

`\tucsRemoveSeatAt{<Reihe>}{<Sitznummer>}`

Entfernt den Sitz `<Sitznummer>` in der Reihe `<Reihe>` aus dem Layout. Dabei beziehen sich `<Reihe>` und `<Sitznummer>` auf das vollständige Layout, ändern sich also nicht durch bereits entfernte Sitze. Wenn die `<Sitznummer>` negativ ist, wird von rechts aus gezählt

`\tucsRemoveSeats{<Liste>}`

Entfernt alle in der Liste vorkommenden Sitze aus dem Layout. Die Liste enthält dabei komma-separierte Einträge der Form `{<Reihe>,<Sitznummer>}`. Für `<Reihe>` und `<Sitznummer>` gilt wie bei `\tucsRemoveSeatAt` das ursprüngliche Layout. Im folgenden Beispiel werden die ersten drei Sitze links und der erste Sitz rechts in der ersten Reihe entfernt:

```
1 \tucsRemoveSeats{{1,1},{1,2},{1,3},{1,-1}}
```

`\tucsSetAisle[<Startreihe>-<Endreihe>]{<Sitznummer>}`

An der Stelle von `<Sitznummer>` wird durch alle Reihen ein Gang eingefügt. Nutzen Sie das optionale Argument, wenn der Gang nicht alle Reihen erfassen soll (Stichgang). Für horizontale Gänge (die also einen Sitzblock in einen vorderen und hinteren Teil zerlegen, anstatt

einen rechten und einen linken) nutzen Sie bitte `\tucsRemoveSeats`. Das ist im Prinzip auch bei vertikalen Gängen möglich, jedoch können bei der automatischen Sitzschemazuweisung mit `\tucsSetAisle` erzeugte Gänge anders als die mit `\tucsRemoveSeats` erzeugten Gänge behandelt werden.

3.4 Ausgabe

`\tucsDrawSeating[seat width=<Breite>, seat height=<Höhe>]`

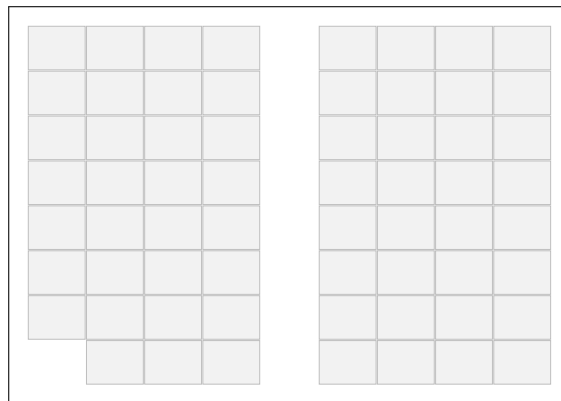
Gibt den Sitzplan aus. Dabei versucht tucseating, die Ausmaße der Sitz so zu berechnen, dass der zur Verfügung stehenden Platz des Gesamtplans vollständig ausgenutzt wird. Da damit nicht immer alle Bedürfnisse getroffen werden, können in der über das optionale Argument die Breite und Höhe der Sitze auf dem Sitzplan einzeln eingestellt werden.

Das folgende Beispiel zeigt den Code und das Ergebnis für ein einfaches Sitzplanlayout mit einem Mittelgang und einem fehlenden Sitz in der ersten Reihe.

```

1 \documentclass{scrartcl}
2 % Um das Papier gut auszunutzen, nehmen wir es Landscape-Format
3 % und reduzieren die Ränder
4 \usepackage[a4paper,landscape,inner=1cm,outer=1cm,top=1cm,bottom=1cm]{
  geometry}
5 \usepackage[shape=rectangle, rows = 8, seats per row=9]{tucseating}
6 \begin{document}
7   \tucsSetAisle{5}
8   \tucsRemoveSeatAt{1}{1}
9   \tucsDrawSeating
10  \end{document}

```



4 Sitzplatzbelegung

4.1 Parameter zur Konstruktion von Belegungsschemata

Natürlich ist ein Sitzplan ohne die Markierung von belegten Plätzen nur bedingt nützlich. Die Belegung wird in tucseating mit Sitzplatzschemata realisiert. Zur Erzeugung eines Sitzschemas gibt es den Befehl

```
\tucsSeatingScheme[⟨Schlüsselliste⟩]{⟨Name⟩}
```

```
\tucsSeatingScheme*[⟨Schlüsselliste⟩]{⟨Schema⟩}
```

Das Pflichtargument enthält in der Standardvariante des Befehls eine Bezeichnung für ein vorgegebenes Sitzschema, siehe Abschnitt 4.2. In der Sternvariante wird hier eine Zeichenkette von „X“ und „-“ übergeben, die (den Anfang) eines Sitzschemas für eine einzelne Reihe beschreibt, wobei „X“ für einen belegten und „-“ für einen leeren Platz steht. Ist die Zeichenkette kürzer als die Anzahl der Sitze in der Reihe, wird sie wiederholt angewendet. Reihe mehr Plätze als Beispielsweise wird mit

```
1 \tucsSeatingScheme*{X--}
```

jeder dritte Sitz belegt.

Im optionalen Argument kann wieder eine Liste von Schlüssel-Wert-Paaren übergeben werden, die die restlichen Einstellung für das Sitzschema steuern. Es ist insbesondere in der Sternvariante wichtig, kann aber auch in der Standardvariante genutzt werden. werden:

row sep = ⟨Anzahl⟩ Voreinstellung: 2

Legt fest, in jeder wievielten Reihe Sitze belegt werden.

start row = ⟨Nummer⟩ Voreinstellung: 1

end row = ⟨Nummer⟩ Voreinstellung: ⟨Anzahl Reihen⟩

Durch diese beiden Schlüssel wird gesteuert, in welchen Reihen ein Schema angewendet wird. Dadurch können unterschiedliche Schemata für verschiedene Reihen genutzt werden.

row restart after = ⟨Anzahl⟩ (zunächst leer)

Das Sitzschema wird nach ⟨Anzahl⟩ Reihen zurückgesetzt. Damit wird es möglich, alternierende Reihenabstände zu erzielen, vergleiche Beispiel in Abschnitt 5.3.

aisle counts = ⟨Anzahl⟩ Voreinstellung: 1

Legt fest, wie viele Sitze ein Gang zählt. Dadurch kann berücksichtigt werden, dass ein Gang häufig breiter als ein Sitzplatz ist.

aisle restarts scheme = true|false Voreinstellung: false

Startet das Schema nach einem Gang erneut. Der Schlüssel **aisle counts** wird damit wirkungslos.

ignore aisle = true|false Voreinstellung: false

`ignore removed seats = true|false`

Voreinstellung: `false`

Bei der Zählung von Sitzen, die im Platzlabel genutzt werden kann (vgl. Abschnitt 4.3) werden auch entfernte Sitze und Gänge mitgezählt. Dies ist sinnvoll, damit in Reihen (im Rechteck-Grundlayout) sich gleiche Sitznummern hintereinander befinden. Wird einer der Schlüssel gesetzt, werden entfernte Sitze bzw. Gänge nicht mitgezählt. Dies bietet sich insbesondere im Bogen-Grundlayout an.

`assigned seat label = {<Formatzeichenkette>}`

Voreinstellung: `m{\,}D`

Über diesen Schlüssel kann festgelegt werden, wie die belegten Sitze beschriftet werden. Voreingestellt ist die Nummer der Reihe, gefolgt von einem schmalen Leerzeichen und einem Buchstaben für den laufenden belegten Sitz (z. B. „3 c“). Es sind hier eine Reihe anderer Möglichkeiten einstellbar. Details dazu sind im Abschnitt 4.3 beschrieben.

`\tucsConfigScheme{<Schlüsselliste>}`

Setzt die Schlüssel wie beim optionalen Argument von `\tucsSeatingScheme`, führt aber keine Zuweisung von Sitzen aus.

Schlüsselwerte, die durch `\tucsSeatingScheme` oder `\tucsConfigScheme` gesetzt werden, bleiben erhalten, bis sie explizit neu gesetzt werden.

4.2 Vordefinierte Sitzschemata

Das `tucseating`-Paket definiert eine Reihe vordefinierter Platzierungsschemata. Manche haben auch einen Alternativnamen.

1x1

Jeder Platz ist markiert.

Alternativname: `all`

2x2

Zwischen zwei belegten Plätzen ist jeweils ein Platz bzw. eine Reihe Abstand.

Alternativname: `simple`

2x2-

Zwischen zwei belegten Plätzen ist jeweils ein freier Platz. Nach einer freien Reihe kommen zwei Reihen mit belegten Plätzen. Dies ist das Schema, mit dem in einer Prüfung die größtmögliche Zahl an Studierenden in einem Raum untergebracht werden können, aber trotzdem ein seitlicher Minimalabstand gegeben ist und sich die Aufsicht zu jedem Studierenden kommen kann (entweder von vorn, oder von hinten).

Alternativname: `dense`

2x3

Belegte Sitze haben einen seitlichen Abstand von zwei Sitzen und belegte Reihen einen Abstand von einer Leerreihe. Dies wird in unserer Gruppe als das anzustrebende Standardschema für Prüfungen betrachtet.

Alternativname: `sixpack`

2x3-

Die Reihen werden wie bei 2x2- belegt, der seitliche Abstand innerhalb einer belegten Reihe beträgt jedoch zwei Sitze.

2x4

Belegte Sitze haben einen seitlichen Abstand von drei Sitzen und belegte Reihen einen Abstand von einer Leerreihe.

3x4

Belegte Sitze haben einen seitlichen Abstand von drei Sitzen und belegte Reihen einen Abstand von zwei Leerreihen.

4.3 Sitzbeschriftung

Die Beschriftung der zugewiesenen Sitzplätze kann auf unterschiedliche Weise erfolgen, die durch Zuweisung einer Formatzeichenkette an den Schlüssel `assigned seat label` gesteuert wird. Es stehen dafür vier Zähler zur Verfügung:

- absolute Reihe: die Nummer der Reihe im Sitzlayout
- laufende Reihe: die Nummer der *belegten* Reihe. Reihen, in denen keine Sitz zugewiesen werden, werden hier übersprungen.
- absolute Sitznummer: die Nummer des Sitzes im Sitzlayout. Ob hier auch entfernte Sitze berücksichtigt werden, hängt vom Wert des Schlüssels `ignore removed seats` ab.
- laufende Sitznummer: die Nummer der *belegten* Sitze. Unbelegte Sitze werden bei der Zählung übersprungen.

Jeder dieser Zähler kann unterschiedlich formatiert werden. Dafür werden in der Formatzeichenkette verschiedene Formatierungszeichen benutzt, die in der folgenden Tabelle gelistet sind:

Zähler	Anmerkung	Darstellung als...				
		(arabische) Zahl	Kleinbuchstabe	Großbuchstabe	kleine römische Zahl	große römische Zahl
absolute Reihe	<i>bezieht sich auf das Sitzlayout</i>	m	a	A	y	Y
laufende Reihe	<i>bezieht sich auf das Belegungsschema</i>	r	b	B	i	I
absolute Sitznummer	<i>bezieht sich auf das Sitzlayout</i>	n	c	C	x	X
laufende Sitznummer	<i>bezieht sich auf das Belegungsschema</i>	s	d	D	j	J

Teile der Beschriftung, die nicht als Formatierungszeichen interpretiert werden sollen, werden in doppelte geschweifte Klammern gesetzt (`{{geschnittene Zeichenkette}}`). Beispielsweise erzeugt


```
1 \tucsSeatingScheme[assigned seat label=Y{{-}}D]{2x3}
```

beim vierten Sitz (von links) in der dritten Reihe die Beschriftung „III-B“.

4.4 Sitzplatzliste

Insbesondere wenn es um den Anwendungsfall einer Prüfung geht, ist es nützlich, eine Sitzplatzliste zu erhalten, also eine tabellarische Zuordnung zwischen Prüfling und Sitzplatz. In der aktuellen Version erstellt tucseating keine (L^AT_EX)-Tabelle, kann aber die Tabellenerstellung (mit L^AT_EX oder einer Tabellenkalkulation) unterstützen.

```
\tucsSeatingList[⟨Eingabedatei⟩]{⟨Ausgabedatei⟩}
```

```
\tucsSeatingList*[⟨Eingabedatei⟩]{⟨Ausgabedatei⟩}
```

Erstellt eine CVS-Datei *⟨Ausgabedatei⟩* mit den Labels der belegten Plätze, jeweils einen pro Zeile.

Achtung! Die in die *⟨Ausgabedatei⟩* geschriebenen Labeltexte werden aus den entsprechenden L^AT_EX-Boxen extrahiert und nur die druckbaren Zeichen ausgegeben. Es empfiehlt sich bei Nutzung dieser Funktion daher, in der Formatzeichenkette von `assigned seat label` auf zuviel T_EX-Magie zu verzichten.

In der Sternvariante werden in jeder Zeile die absoluten Koordinaten (Reihe, Platznummer, bezogen auf das Grundlayout) kommasepariert dem Labeln vorangestellt.

Wenn eine Eingabedatei angegeben wurde, wird ihr Inhalt zeilenweise den erzeugten Zeilen vorangestellt. Diese Eingabedatei könnte beispielsweise die Namen oder/und die Immatrikulationsnummern von Studierenden enthalten, die dann in der Ausgabedatei den markierten Plätzen zugeordnet werden. Sind weniger Einträge in *⟨Eingabedatei⟩* als Sitze belegt sind, werden die Felder in der Ausgabe leer gelassen. Wenn dagegen die Anzahl der Sitze nicht ausreicht, gibt tucseating eine Information aus, wer nicht platziert werden konnte.

5 Beispiele

Zur Demonstration des Verhaltens von tucseating sind hier einige Beispiele des Einsatzes dokumentiert.

5.1 Vordefinierter Raum, rechteckig

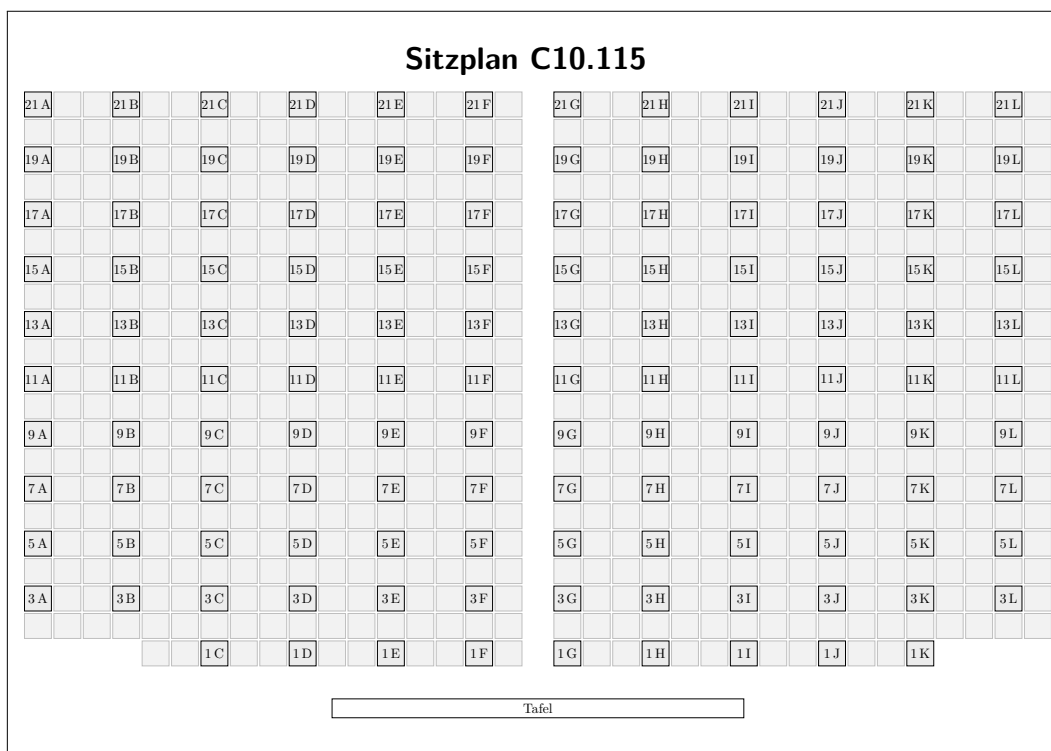
```
1 \documentclass{scrartcl}
2 % Um das Papier gut auszunutzen, nehmen wir es Landscape-Format
3 % und reduzieren die Ränder
4 \usepackage[a4paper,landscape,inner=10pt,outer=10pt,top=1cm,bottom=1cm]{
  geometry}
```

5 Beispiele

```

5 \usepackage[
6     room=C10.115,
7     blackboard
8 ]{tucseating}
9
10 \begin{document}
11 % Standardtitel nimmt viel Platz weg.
12 \centering\textbf{\Huge\sffamily Sitzplan C10.115}\bigskip
13 \tucsSeatingScheme{sixpack}
14 \tucsDrawSeating
15 \end{document}

```



5.2 Vordefinierter Raum, bogenförmig

```

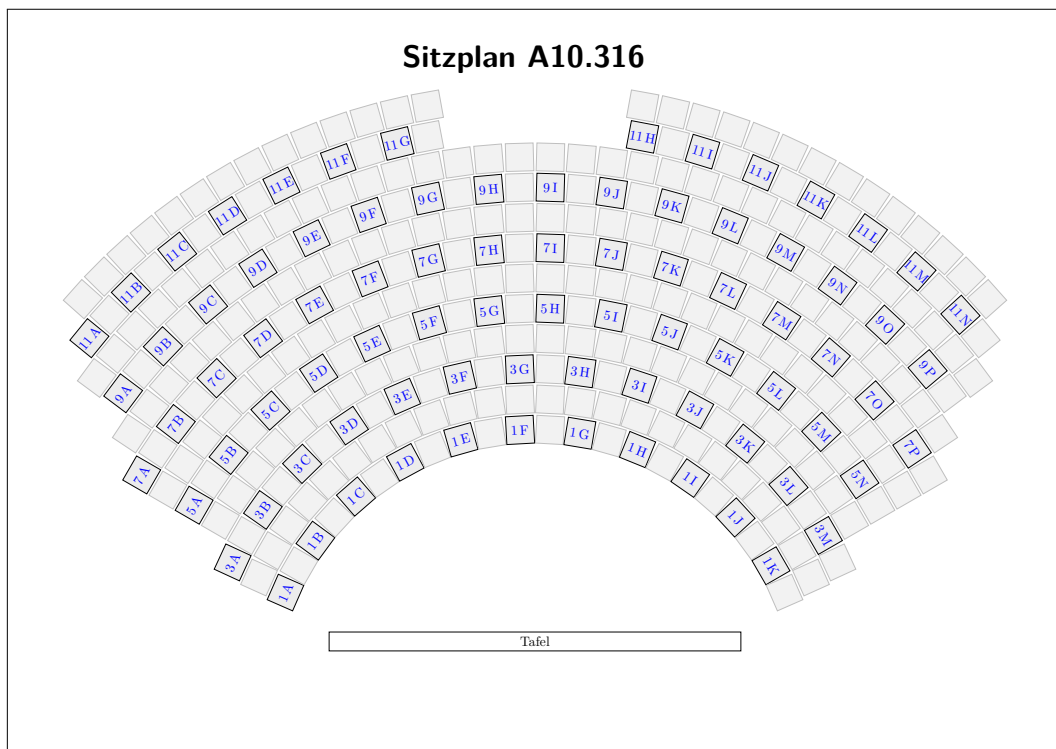
1 \documentclass{scrartcl}
2 % Um das Papier gut auszunutzen, nehmen wir es Landscape-Format
3 % und reduzieren die Ränder
4 \usepackage[a4paper,landscape,inner=10pt,outer=10pt,top=1cm,bottom=1cm]{
  geometry}
5 \usepackage[

```

```

6   room=A10.316,
7   blackboard,
8   assigned seat label color=blue
9   ]{tucseating}
10
11 \begin{document}
12   % Standardtitelei nimmt viel Platz weg.
13   \centering\textbf{\Huge\sffamily Sitzplan A10.316}\bigskip
14   \tucsSeatingScheme[ignore removed seats]{2x2}
15   \tucsDrawSeating
16 \end{document}

```



5.3 Eigenes Platzlayout und Sitzschema

```

1 \documentclass{scrartcl}
2   % Um das Papier gut auszunutzen, nehmen wir es Landscape-Format
3   % und reduzieren die Ränder
4   \usepackage[a4paper,landscape,inner=10pt,outer=10pt,top=1cm,bottom=1cm]{
5     geometry}
6   \usepackage[

```

5 Beispiele

```

6   shape=rectangle, rows=9, seats per row=9,
7   assigned seat background color=yellow
8   ]{tucseating}
9
10  \begin{document}
11    \tucsRemoveSeats{{1,1},{1,-1},{2,1},{2,-1},{3,1},{3,-1}}
12
13    \tucsSeatingScheme[ignore removed seats,end row=3]{2x3}
14    \tucsSeatingScheme[start row=5, end row=10]{2x4}
15    \tucsDrawSeating
16  \end{document}

```

9 A				9 B				9 C
7 A				7 B				7 C
5 A				5 B				5 C
	3 A			3 B				3 C
	1 A			1 B				1 C

5.4 Dame

```

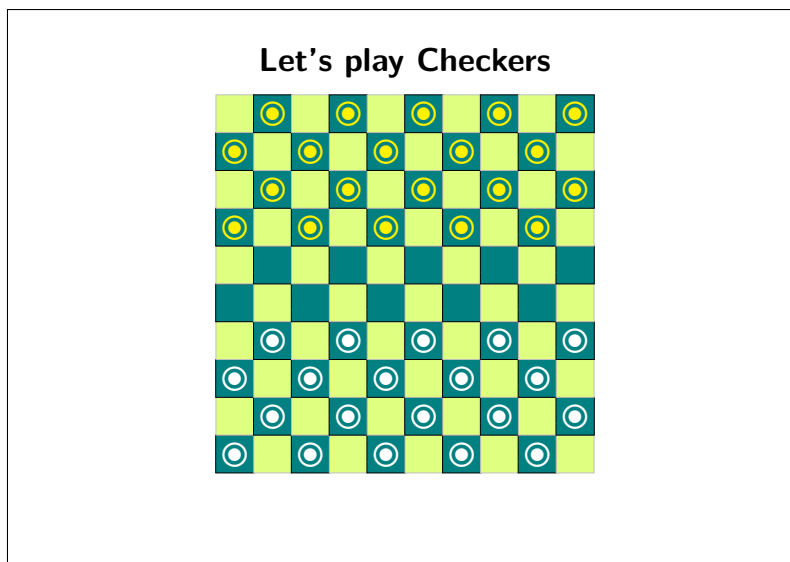
1  \documentclass{scrartcl}
2  \usepackage[a5paper,landscape,top=1cm,bottom=1cm]{geometry}
3  \usepackage[shape=rectangle, rows=10,seats per row=10,
4    assigned seat label color=blue, assigned seat background color=teal,
5    empty seat background color=lime!50,
6    seat distance=0pt]{tucseating}

```

```

7 \usepackage{stix}
8 \begin{document}
9 \centering\textbf{\Huge\sffamily Let's play Checkers}\bigskip
10
11 \tucsConfigScheme{assigned seat label={}}
12 \tucsSeatingScheme*[start row=5, end row=5]{X-}
13 \tucsSeatingScheme*[start row=6, end row=6]{-X}
14 \tucsConfigScheme{assigned seat label font=\Huge, assigned seat label={{\textcolor{white}{\circledbullet}}}}
15 \tucsSeatingScheme*[start row=1, end row=3, row sep=2]{X-}
16 \tucsSeatingScheme*[start row=2, end row=4]{-X}
17 \tucsConfigScheme{assigned seat label={{\textcolor{yellow}{\circledbullet}}}}
18 \tucsSeatingScheme*[start row=7, end row=9]{X-}
19 \tucsSeatingScheme*[start row=8, end row=10]{-X}
20
21 \tucsDrawSeating[seat width=1cm, seat height=1cm]
22 \end{document}

```



6 Deklaration neuer Raumlayouts

In der augenblicklichen Version hat das tucseating-Paket erst eine gewisse Anzahl von Räumen mit ihrem Sitzlayout erfasst. Nutzer werden daher häufig darauf angewiesen sein, eigene Layouts anzulegen. Neben der Möglichkeit zur Ad-hoc-Kreation von Layouts, wie sie im Abschnitt 3 beschrieben sind, besteht auch die Möglichkeit das Paket selbst zu erweitern und damit Layouts für neue Räume anzulegen, die dann einfach über den `room`-Schlüssel angesprochen werden können. Neue Raumlyouts können über zwei Wege integriert werden:

- Die zu dem tucseating-Paket gehörende Datei `room1.tsr` wird erweitert.
- Es wird eine eigene `.tsr`-Datei angelegt und über die Option `layout file` eingebunden.

Egal welcher Weg gewählt wird, in `.tsr`-Dateien können zwei Befehle genutzt werden:

`\tucsDeclareRoom{⟨Raumbezeichnung⟩}{⟨Schlüsselliste⟩}`

Ein neues Rauml原因 wird für den Raum `⟨Raumbezeichnung⟩` angelegt. Die Schlüsselliste *muss* die drei Schlüssel

`shape = rectangle|arc` (erforderlich)

`rows = ⟨Anzahl der Sitzreihen⟩` (erforderlich)

`seats per row = ⟨Sitze pro Sitzreihe⟩` (erforderlich)

enthalten, die die analoge Bedeutung zu den gleichnamigen Schlüsseln aus Abschnitt 3 haben. Diese Angaben zum Basislayout *müssen* von dem Schlüssel

`init` (erforderlich)

gefolgt werden. Anschließend kann mit

`aisle = ⟨Sitznummer⟩`

`remove = {⟨Liste⟩}`

das Layout angepasst werden. Die beiden Schlüssel funktionieren analog zu `\tucsSetAisle` und `\tucsRemoveSeats`, vgl. Abschnitt 3.3.

```

1 \tucsDeclareRoom{C10.115}{
2   shape=rectangle,
3   rows=21,
4   seats per row=35,
5   init,
6   aisle=18,
7   remove={{1,1},{1,2},{1,3},{1,4},{1,-1},{1,-2},{1,-3},{1,-4}}
8 }
```

`\tucsAliasRoom{⟨Aliasname⟩}{⟨Originalname⟩}`

Setzt `⟨Aliasname⟩` als einen Ersatznamen für `⟨Originalname⟩`.

7 Beschränkungen und Bugs

Auch wenn das tucseating-Paket zumindest in unserer Gruppe im praktischen Einsatz ist, ist es als experimentell zu betrachten. Die 0 in der Hauptversionsnummer deutet diesen Umstand an. Er bedeutet insbesondere, dass sich die API noch ändern kann.

Das Paket unterliegt einigen Einschränkungen. Ob diese in kommenden Versionen behoben werden, steht noch nicht fest. Das Paket ist auf die Erstellung separater Sitzpläne ausgelegt und kann daher nicht dafür genutzt werden, mehrere Sitzpläne in einem Dokument darstellen. Ebenfalls können keine Sitzlayouts mit versetzten Sitzreihen dargestellt werden.

Die Anzahl der vordefinierten Räume ist derzeit noch eher gering. Es besteht die Absicht, in künftigen Patches weitere Räume der TU Chemnitz aufzunehmen. Die Mitarbeit durch Zusenden von Informationen über Sitzlayouts oder entsprechende Patches von `rooms1.tsr` ist ausdrücklich willkommen.

Im Moment sind keine Bugs bekannt, aber es werden sicherlich welche existieren. Bitte nutzen Sie für Bugs-Reports oder Pull-Requests GitHub: <https://github.com/tuc-osg/tucseating>.

8 Lizenz

Es ist erlaubt, diese Software unter den Bedingungen der \LaTeX Project Public License (LPPL), Version 1.3c oder später, zu kopieren und zu verteilen (<http://www.latex-project.org/lppl.txt>).