

Having Fun with L^AT_EX X

Adelheid Bonnetsmüller, bonnetsmueller@icloud.com

DANTE Frühjahrstagung 2024, Weimar

Dieses Mal mit

- einem kleinen Rückblick
- a bisserl Musi geht oiwei!
- Jetzt schlägts 12:00 – oder doch eher 12⁰⁰Uhr ?
- Wann sind wir denn endlich da?
- Rechnen leicht gemacht
- eine kleine (große) Hommage: Dinosaurier
- Vom Großen zum Kleinen: Das Bohr'sche Atommodell
- Was zu beweisen war...

... das gab's in 10 Ausgaben
"Having Fun With LaTeX"

- Having Fun With LaTeX I: Wien 2009 - vor 15 Jahren!
- Vorgestellt wurden unter anderem:
 - Kochbücher, (Kreuzwort-)rätsel, Notizen, Pseudocode
 - verschütteter Kaffee, Lizenzsymbole, Würfel, QR-Codes, EAN-/Barcodes
 - colorway, eemeir, kleine Pakete zur Graphikeinbindung
 - Übungsblätter, Korrekturlesen, Mathematik an Halloween, Akronyme
 - Boxen mit Logos, Zigarettenschachteln, altdeutsche Schriften
 - und viele, viele Bilder: Simpsons, Hüte, Smileys, ...

A bisserl Musi geht oiwei!

- Autor: Émile Daneault, 2011
- verwendet xcolor, ifthen und xargs
- zeichnet zwei Oktaven einer Klaviatur mit Punkten auf bestimmten Tasten (Noten)
- Standard-Highlight-Farbe: orange (kann in .sty-Datei geändert werden)
- keine Doku, Erläuterung in README- oder .sty-Datei
- Fehler bei den Argumenten werden nicht ausgegeben - es wird kein Punkt gesetzt!

Einbinden über `\usepackage{piano}` (keine Optionen)

Es wird ein einziger Befehl mit sieben Argumenten zur Verfügung gestellt:

```
\piano[1][2][3][4][5][6][7]
```

Mit den sieben Argumenten werden die Töne / Tasten angegeben.

Leicht in `.sty` erweiterbar

Folgende Töne sind definiert:

- 1. Oktave

Co, Cso, Do, Dso, Eo, Fo, Fso, Go, Gso, Ao, Aso,
Bo

- 2. Oktave

Ct, Cst, Dt, Dst, Et, Ft, Fst, Gt, Gst, At, Ast,
Bt

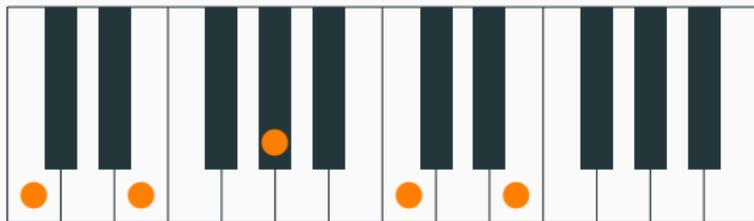


Abbildung 1: C+ Akkord

```
\keyboard[Co][Eo][Gso][Ct][Et]  
\caption{C+ Akkord}  
\label{Chord1}
```

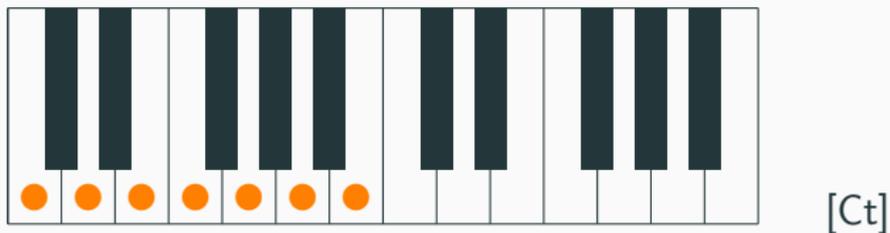


Abbildung 2: Achtes Argument verwendet

```
\keyboard[Co] [Do] [Eo] [Fo] [Go] [Ao] [Bo] [Ct]
```

Jetzt schlägt's 12:00 Uhr...

- Autor: Olaf Meltzer, 2016
- ermöglicht Darstellung von Uhrzeiten und Uhrzeitbereichen im ehemals in Deutschland gebräuchlichen Format
- ermöglicht Ausgabe der aktuellen Uhrzeit in verschiedenen Formaten
- aktuell bedeutet hier: zum Zeitpunkt des Kompilierens.

- Einbinden über `\usepackage{uhrzeit}` (keine Optionen)
- verwendet `soul`
- stellt folgende Befehle zur Verfügung:
 - `\uhr{Std.}{Min.}`
Angabe der Uhrzeit in altem Format
 - `\vonbis{Std1}{Min1}{Std2}{Min2}`
Angabe eines Uhrzeitenbereichs im alten Format
 - `\dtd \dte \uhri \uhrii \uhriii \uhriv`
Angabe der aktuellen Uhrzeit in verschiedenen Formaten

- `\uhr{8}{45}` 8⁴⁵ Uhr
- `\uhr{8}{05}` 8⁰⁷ Uhr
- `\vonbis{8}{45}{10}{30}` 8⁴⁵ – 10³⁰ Uhr

- geregelt für Deutschland in DIN 5008 (regelt Uhrzeit- und Datumsformat)
- gliedert die Teile von Uhrzeiten mit Doppelpunkten.
- es wird empfohlen, Std., Min. und Sekunden zweistellig anzugeben. Wenn nur die Stunde angegeben wird, dann ist auf die führende Null zu verzichten. Das Wort „Uhr“ ist durch ein Leerzeichen abzutrennen.

korrekt: 8 Uhr

falsch: 08 Uhr

korrekt: 08:45 Uhr

falsch: 8:45 Uhr

korrekt: 08:45:07 Uhr

falsch: 8:45:07 oder 8:45:7 Uhr

- in Österreich und der Schweiz gelten andere Regeln (im englischsprachigen Raum sowieso)
- ebenfalls anderes Format: taktische Uhrzeit oder technischer Gebrauch
- Duden inzwischen weniger restriktiv: 8.45, 8:45 oder 8⁴⁵ Uhr

Makro	Format	Gebrauch
<code>\dtd</code>	8.45	Eher ungebräuchlich
<code>\dtc</code>	08:45	International gebräuchlich
<code>\uhri</code>	8.45 Uhr	lt. Duden möglich, in der Schweiz üblich
<code>\uhrii</code>	08:45 Uhr	DIN 5008
<code>\uhriii</code>	8 ⁴⁵ Uhr	Ehemals in Deutschland üblich
<code>\uhriv</code>	08 ⁴⁵ Uhr	Auch früher eher selten gebraucht

Wann sind wir denn endlich da?

- Autor: Marcel Jira, 2013
- gibt farbige Fortschrittsbalken aus
- vielfältig veränderbar (Farben, Rahmen, Punkte, Größe, ...)
- benötigt `calc`, `kvsetkeys`, `kvoptions`, `tikz`

- Einbinden mittels
`\usepackage[.Optionen.]{progressbar}`
- alle Optionen können entweder global beim Einbinden oder lokal im Dokument geändert werden

Das Paket stellt zwei Befehle zur Verfügung:

- `\progressbar[.Optionen.]{.Nummer.}` – gibt den Fortschrittsbalken mit den (lokal gültigen) gewählten Parametern aus
- `\progressbarchange{.Definition.}` ändert ab dem Befehlsaufruf global die entsprechenden Variablen

Für beide Befehle stehen die gleichen Optionen zur Verfügung, die dem Paket auch beim Laden mitgegeben werden können.

- `\progressbar {0.666}`



Der Fortschrittswert sollte (logischerweise) im Bereich zwischen $[0;1]$ liegen.

Gibt man Werte > 1 oder < 0 an, wird der Balken mit dem Wert 1 bzw. 0 gezeichnet.

- `\progressbar {-0.5}`



- `\progressbar {15}`



Es sind folgende Optionen definiert:

- `width` – Breite des gesamten Fortschrittsbalkens, default:6em

`\progressbar {0.666}` 

`\progressbar[width=4cm] {0.666}` 

- `heightr` – **relative** Höhe (i. Vgl. zum Text), default:1
`\progressbar {0.666}` 
`\progressbar[heightr=0.5] {0.666}` 
- `heighta` – **absolute** Höhe des Fortschrittsbalkens, default:
(unused)
`\progressbar[heighta=10pt] {0.666}` 
`\progressbar[heighta=\heightof{Z}] {0.666}`
Z 
- `r` steht für **relative**, `a` steht für **absolute** - es empfiehlt sich die Optionenvariante mit `r` zu nutzen (Dokumenteneinstellungen wie Textgröße werden berücksichtigt).

- `roundnessr` – **relative** Rundung relativ zur Höhe, default:0.15

`\progressbar[roundnessr=0.4] {0.666}` 

- Werte über 0.5 ergeben wenig Sinn (mehr als die Hälfte der Höhe gekrümmt)

`\progressbar[roundnessr=0.7,heighta=1cm] {0.666}`



- `roundnessa` – **absoluter** Radius der Rundung relativ zur Höhe, default: (unused)

`\progressbar[roundnessa=3pt] {0.666}` 

`\progressbar[roundnessa=3pt,heighta=1cm] {0.666}`



- `borderwidth` – Rahmenbreite, default:0.8pt

`\progressbar[borderwidth=2pt] {0.666}`



`\progressbar[borderwidth=0.01em] {0.666}`



- `subdivisions` – Unterteilung des Balkens, default:10
`\progressbar[subdivisions=3] {0.666}` 
`\progressbar[subdivisions=15] {0.666}` 
- `tickwidth` – Breite der Unterteilungsstriche, default:0.4pt
`\progressbar[tickwidth=1mm] {0.666}` 
`\progressbar[tickwidth=0.1pt] {0.666}`

- `ticksheight` – Höhe der Unterteilungsstriche als Bruchteil der Gesamthöhe, default:0.33; Werte > 1 werden auf 1 gesetzt
`\progressbar[ticksheight=0.1] {0.666}` 
`\progressbar[ticksheight=1.5] {0.666}` 

- `linecolor` – Linienfarbe, default:black
`\progressbar[linecolor=red] {0.2}` 
- `tickscolor` – Linienfarbe, default:black
`\progressbar[tickscolor=red] {0.3}` 
- `emptycolor` – Linienfarbe, default:black!10
`\progressbar[emptycolor=red] {0.4}` 
- `filledcolor` – Linienfarbe, default:black!50
`\progressbar[filledcolor=red] {0.5}` 
- ... und alle vier Farboptionen zusammen:
`\progressbar[linecolor=red,tickscolor=blue,emptycolor=green!20,filledcolor=yellow!50] {0.8}`


Rechnen leicht gemacht ... mit
einem LaTeX-Taschenrechner

- Autor: Philippe de Sousa, 2015
- Darstellung eines wissenschaftlichen Taschenrechners
- Darstellung einzelner Taschenrechnertasten
- Darstellung von Rechenoperationen im Display
- Mini-Taschenrechner-Symbole stehen zur Verfügung
- Paketdokumentation nur auf französisch
- Definition der Farben, Tasten, Befehle usw. ebenfalls teilweise auf französisch

Um einen kompletten Taschenrechner darzustellen, gibt es den Befehl

- `TiCCalc[.Optionen.]` mit den Optionen:
 - `title` – Name des Taschenrechners, default: TI-Collège
 - `color calc` – Farbe des Rahmens, default: brown!20
- Farbe und Bezeichnung der Tasten sowie deren Anordnung sind so nicht änderbar!
- Es können aber einzelne Tasten „angesprochen“ werden (analog Koordinaten) - beispielsweise um einen Rahmen zu ziehen etc.



`\TiC`

Standardtaste



`\TiC[style=general]`

Tastenform für Funktionen (default)



`\TiC[style=number]`

Tastenform für Zahlen



`\TiC[style=arrows]`

Pfeiltaste(n)



sin

`\TiC[style=number, rounded=none]` keine Rundung



sin

`\TiC[style=number, rounded=left]` Rundung links



sin

`\TiC[style=number, rounded=right]` Rundung rechts

`rounded` kann nur in Kombination mit `style=number` verwendet werden, sonst hat die Option keinen Effekt.

Der Default-Wert ist `none`.

ticollege - einzelne Tasten, Tastenbeschriftung und Textposition

TEXT

`\TiC[principal=TEXT]` Beschriftung der Taste
default: `sin`

sin

`\TiC[position=0.5]` Position der Beschriftung

sin

`\TiC[position=1]` Position der Beschriftung

sin

`\TiC[position=1.5]` Position der Beschriftung

`position` ist bei der Pfeiltaste ohne Effekt. Default ist `0.9`



```
\TiC[principal={$\triangleright$ simp}]  
\TiC[principal={$\triangleright$ simp},  
fontsize=2pt]
```

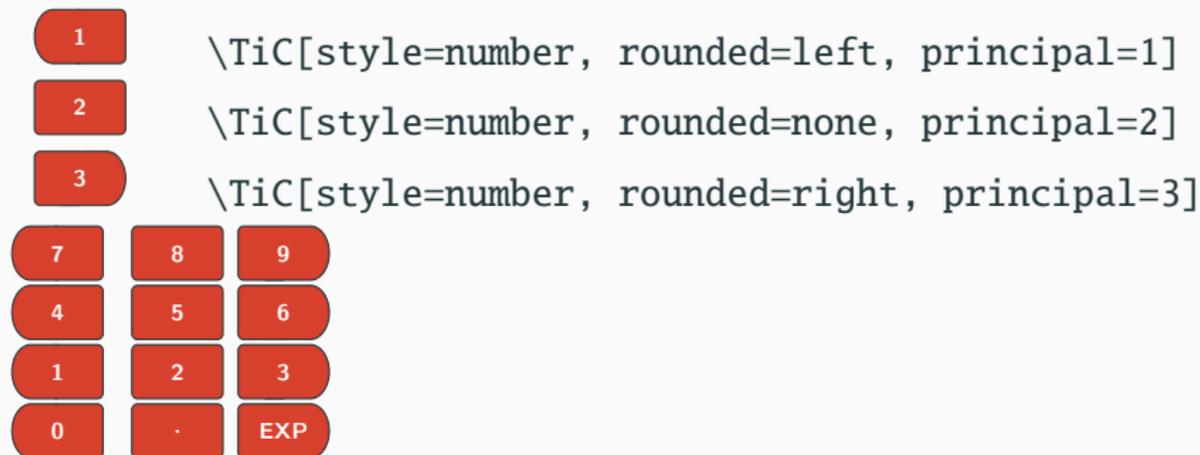
Schriftgröße innerhalb der Taste, default: 6pt

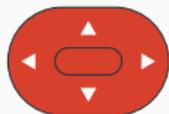
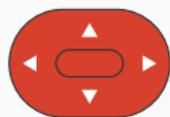


```
\TiC[style=number, principal=2]  
\TiC[style=number, principal=2,  
fontsize=10pt]
```

Schriftgröße innerhalb der Taste

fontsize ist bei der Pfeiltaste ohne Effekt.





default `\TiC[... ,raise=-0.25cm]` `\TiC[... ,raise=0.25cm]`
verschiebt die Tastenposition in Bezug zur Grundlinie um den Wert
 $raise = \pm x$

\sin^{-1}

sin

`\TiC[second=$\sin^{-1}]$`

e^x

ln

`\TiC[principal=ln, second=e^x]`

Für besondere Fälle wurden folgende Befehle definiert:

†

:

`\Div`

$\sqrt[n]{}$

x^n

`\TIRacine[n]`

$\sqrt{}$

x^2

`\TIRacine`

◀◀

—

`\ContrastDown`

▶▶

+

`\ContrastUp`

\sin^{-1}

sin

`\TiC[second=\sin^{-1}]` (default)

\sin^{-1}

sin

`\TiC[second=\sin^{-1}], colour text = black]`

\sin^{-1}

sin

`\TiC[second=\sin^{-1}], colour second = black]`

\sin^{-1}

sin

`\TiC[second=\sin^{-1}], colour key = blue!20]`

\sin^{-1}

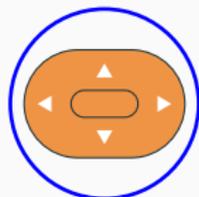
sin

`\TiC[second=\sin^{-1}], colour text = white!10
colour second = orange!40!yellow!100,
colour key = black!40]`

- Um die Funktionsweise einzelner Tasten zu erklären, können diese eingekringelt werden.
- bei den Kreisen können Farbe, Linienstärke und der Radius angepasst werden.



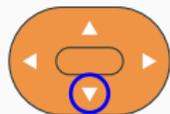
```
\TiC[principal=$x^2$, second={\TiRacine},  
circle=true, thickness=0.5pt]
```



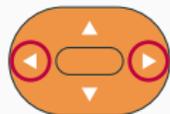
```
\TiC[style=arrows, colour key=TI0orange,  
circle=true, radius=25pt, colour circle=blue]
```



```
\TiC[style=arrows, colour key=TIOrange,  
circleup=true]
```



```
\TiC[style=arrows, colour key=TIOrange,  
circledown=true, colour circle=blue]
```



```
\TiC[style=arrows, colour key=TIOrange,  
circleleft=true, circleright=true,  
colour circle=purple]
```

- man kann den Tasten ein Label verpassen (`name=`), um sie später zu referenzieren

- mit dem Befehl `TiCMenu[.Optionen.]{.Argumente.}` kann man mit einer gesperrten Schrift das Menü angeben.
- Optionen sind `size`, `select`, `colour box`, `text`, das Argument ist der Name des Menüs.

RND

POL

```
\TiCMenu[select=true]{rnd} \TiCMenu[colour box=red]{pol}
```

POL

```
\TiCMenu[size=8pt]{pol}
```

Beispiel: komplette Anzeige des Menüs mit markierten Funktionen:

```

MATHS          NUM          RND          POL
1 : PGCD(
2 : PPCM(
3 : abs(
    
```

```

MATHS          NUM          RND          POL
1 : arrondi(
2 : reste(
3 : partEnt(
    
```

```

MATHS          NUM          RND          POL
1 : rand
2 : randn(
3 : P▶Rx
    
```

```

MATHS          NUM          RND          POL
1 : R▶Pr
2 : R▶P $\theta$ 
3 : P▶Rx
    
```

Um die Anzeige im Display darzustellen gibt es den Befehl:

```
\TiCScreen[.Optionen.]{.Argument.}
```



ein leeres Display (`\TiCScreen{}`)

Optionen:

`colour screen` = Farbe des Displays (default: ForestGreen!15)

`screenname` = Label für das Display zur späteren Referenzierung,

default: ecran

`width` = <Zahl> Breite, default: 4

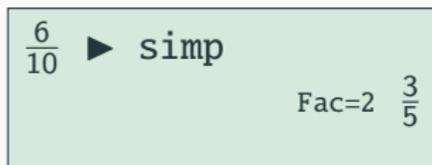
`height`= <Zahl> Höhe, default: 1

Im Argument des Befehls steht die Rechnung, die im Display angezeigt werden soll.

Die einzelnen Operationen werden mit Komma getrennt, ein / signalisiert das Ende der Rechnung und eine neue Zeile.

3+2	
	5
1+2+3+4+5+6+7+8+9+10+11+12+13	
	91

```
\TiCScreen[colour screen=blue!10, height=2, width=6]
{3+2
/5
/,
1+2+3+4+5+6+7+8+9+10+11+12+13+14+15+16
/91}
```


$$\frac{6}{10} \blacktriangleright \text{simp} \qquad \text{Fac}=2 \quad \frac{3}{5}$$

```
\TiCScreen[height=1.5]%  
{  
{\frac{6}{10}$ $\blacktriangleright$ simp}  
/  
\scriptsize Fac=2} $\frac{3}{5}$  
}
```

Zu guter Letzt bietet das Paket noch Taschenrechnersymbole an:
`\TiCCalc* [.Optionen.]`.

Optionen können sein

- `calcscale` = `<Zahl>` Größe des Symbols, default: 0.5
- `calcrotate` = `<Zahl>` Winkel, default: -30
- `calcraise` = `<dim>` Verschiebung des Symbols nach oben oder unten, default: -2ex



`\TiCCalc*`



`\TiCCalc*[calcscale=0.25]`



`\TiCCalc*[calcscale=1]`



`\TiCCalc*[calcrotate=0]`



`\TiCCalc*[calcrotate=90]`



`\TiCCalc*[calcrotate=-30]`



`\TiCCalc*[calcrotate=0]`



`\TiCCalc*[calcrraise=0ex]`



`\TiCCalc*[calcrraise=4ex]`

Eine kleine (oder große) Hommage: Dinosaurier und mehr

- Autoren: viele, gepflegt/hochgeladen von Fernando de Souza Bastos, 2022
- Einbinden über `\usepackage{figchild}`
- ermöglicht das Einbinden von Piktogrammen und kleinen Zeichnungen für Kinder
- es sind aktuell 454 Bildchen vorhanden, von denen hier nur eine kleine Auswahl gezeigt wird
- benötigt `tikz` und `xcolor`

Stellt folgenden Befehl zur Verfügung:

```
\imagedname{Picture size}{Picture color}{Line thickness}
```

mit folgenden Argumenten

- `Picture size` Größe des Bildes (Zahl)
- `Picture color` Farben der Linien
- `Line thickness` Linienstärke in pt (Zahl)



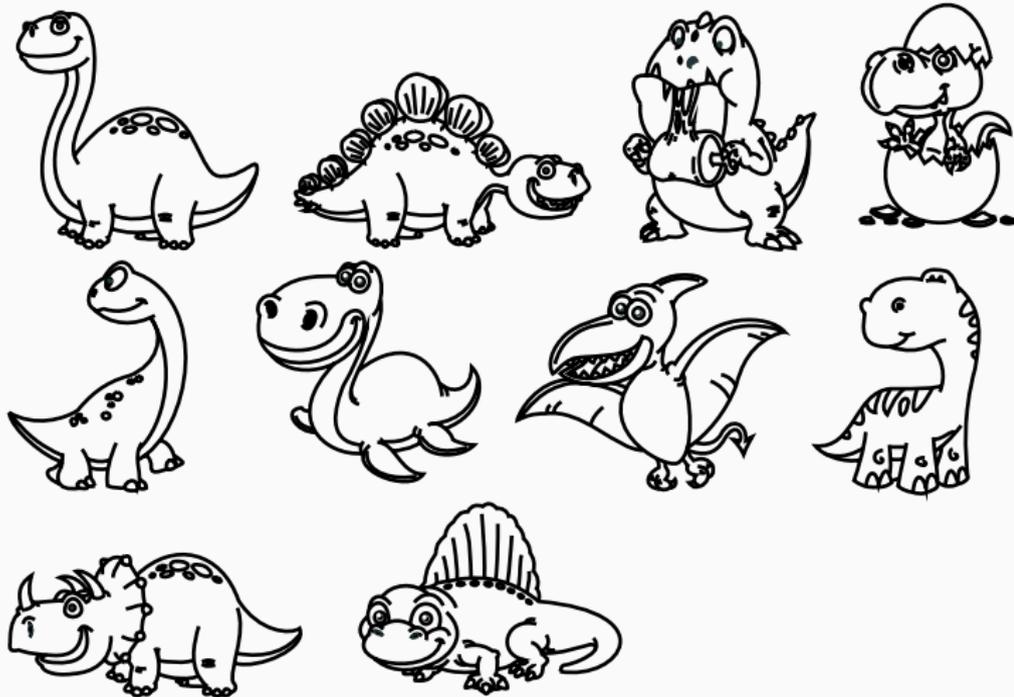
```
\fcDinosaurA{0.1}{black}{1}
```

```
\fcDinosaurA{0.1}{ForestGreen!20}{1}
```

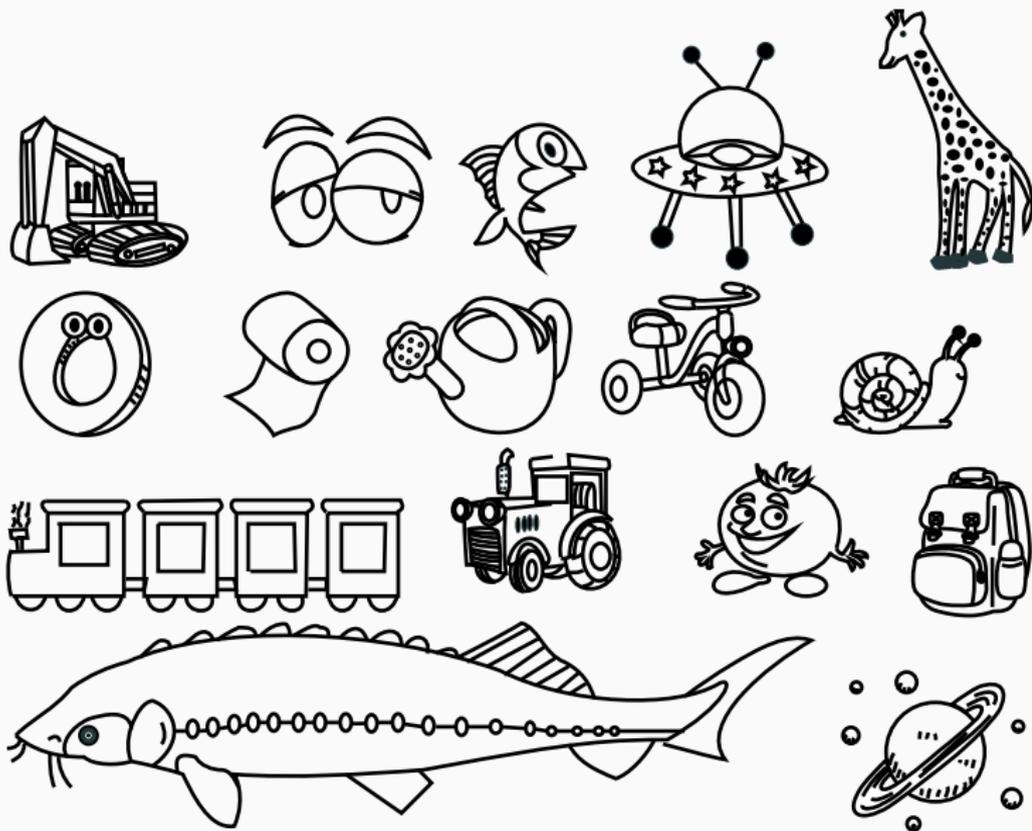
```
\fcDinosaurA{0.1}{black}{0.2}
```

```
\fcDinosaurA{0.05}{red!20}{0.8}
```

figchild - alle Saurier



figchild - Beispiele



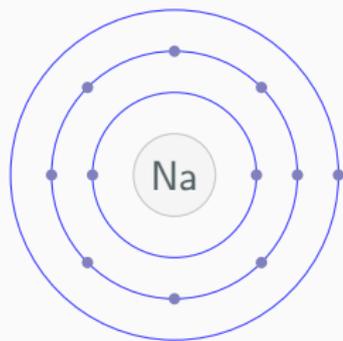
Vom Großen zum ganz Kleinen: das Bohr'sche Atommodell

- Autor: Clemens Niederberger, 2015
- Einbinden über `\usepackage[.Optionen.]{bohr}`
- verwendet `pgf`, `pgfopts`, `elements` und `cnltx-base`
- `elements` wird sicher auch mal in diesem Rahmen vorgestellt
- Optionen können global beim Einbinden mitgegeben werden oder lokal im Dokument

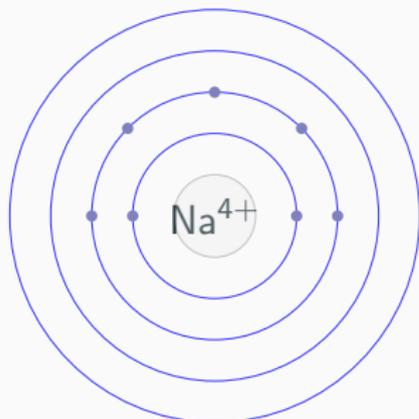
Das Paket stellt den Befehl `\bohr[.1.]{.1.}{.2.}` mit einer Option und zwei Argumenten zur Verfügung

- `[.1.]` Anzahl der Schalen
- `{.1.}` Anzahl der Elektronen
- `{.2.}` Atomname

bohr - Beispiel



`\bohr{11}{Na}`

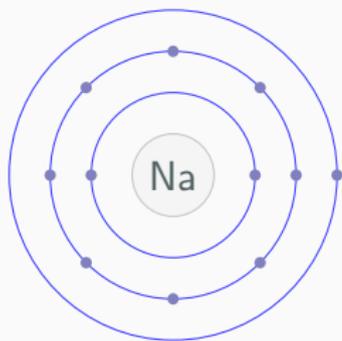


`\bohr[4]{7}{Na^{4+}}`

Mit `\setbohr{.Argumente.}` können verschiedene Anpassungen vorgenommen werden.

- `insert-symbol=true|false` setzt Atomname automatisch, falls `{.2.}` fehlt. Achtung! Funktioniert nur wenn alle Elektronen noch gebunden sind. (default: false)
- `insert-number=true|false` setzt Anzahl Elektronen automatisch, falls `{.1.}` fehlt, default: false
- `nucleus-radius=<dim>` Radius des Nukleus, default: 1em

bohr - Beispiel



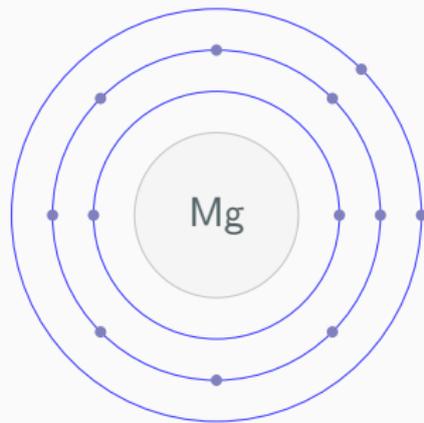
```
\setbohr{insert-symbol=true}
```

```
\bohr{11}{}
```



```
\setbohr{insert-number=true}
```

```
\bohr{}{Na}
```

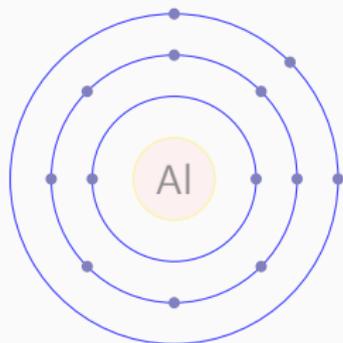


```
\setbohr{nucleus-radius=2em}
```

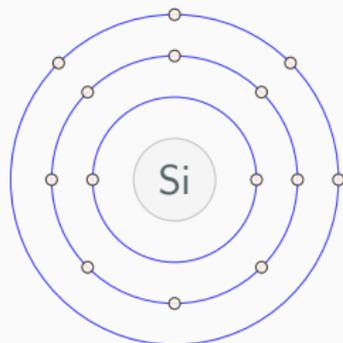
```
\bohr{12}{Mg}
```

- `nucleus-options-set=` Fülloptionen für den Kern (default: `draw=black!80,fill=black!10,opacity=.25`)
- `nucleus-options-add=`fügt angegebenen Wert den TikZ-Optionen hinzu
- `electron-options-set=` Fülloptionen für das Elektron (default: `blue!50!black!50`)
- `electron-options-add=`fügt angegebenen Wert den TikZ-Optionen hinzu
- `electron-radius=<dim>` Radius der Elektronen, default: 1.5pt

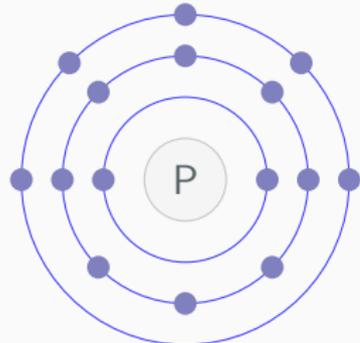
bohr - Beispiel



```
\setbohr{inucleus-options-set=  
{draw=yellow!80,fill=red!10,  
opacity=.50}}\bohr{14}{Si}
```



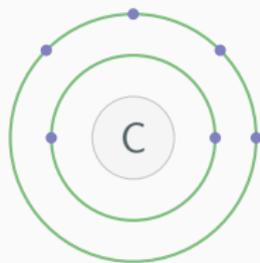
```
\setbohr{electron-options-set={draw=  
black!80,fill=orange!10,opacity=.90  
}\bohr{15}{P}
```



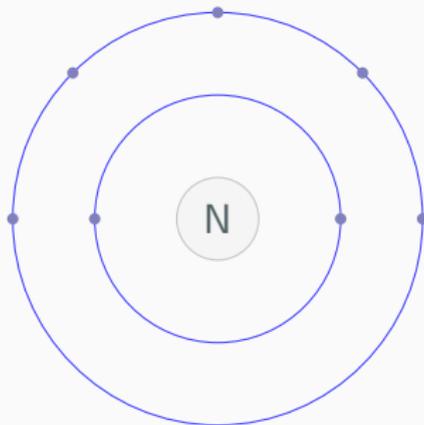
```
\setbohr{electron-radius=3pt}  
\bohr{14}{Al}
```

- `shell-options-set=` Optionen für die Schalenkreise (default: `draw=blue!75,thin`)
- `shell-options-add=`fügt angegebenen Wert den TikZ-Optionen hinzu
- `shell-dist=` Abstand der innersten Schale vom Kern und von den Schalen untereinander, default: `1em`

bohr - Beispiel



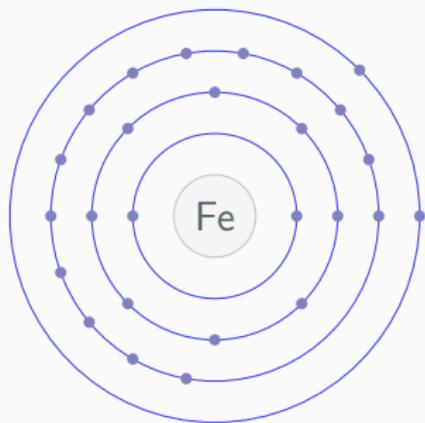
```
\setbohr{shell-options-set={draw=  
\setbohr{green!50!black!50,thick}}  
\bohr{7}{N}
```



```
\setbohr{shell-dist=1.5em}  
\bohr{6}{C}
```

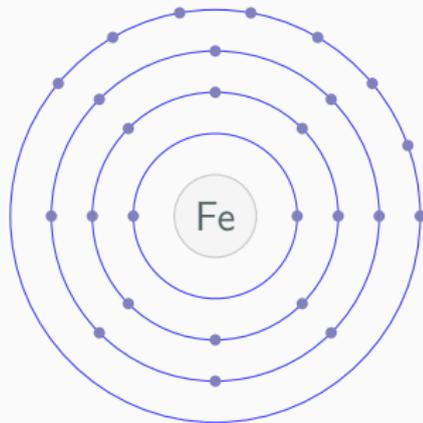
- `distribution-method=periodic|quantum` Angabe zum Verteilungsprinzip, default: quantum

bohr - Beispiel



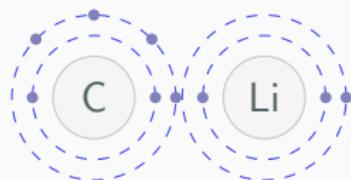
```
\setbohr{distribution-method=quantum}
```

```
\bohr{26}{Fe}
```



```
\setbohr{distribution-method=periodic}
```

```
\bohr{26}{Fe}
```



```
\setbohr{  
  shell-options-add = dashed,  
  shell-dist = .5em}  
\bohr{6}{C}\bohr{3}{Li}
```

Was zu beweisen war....

- Autor: S. Jensen, 2017
- Einbinden über `\usepackage{endofproofwd}`
- stellt genau einen Befehl zur Verfügung: `\wasserdicht` - ohne Optionen und Argumente, verwendet `graphicx` und `import`.



Endofproofwd - Beispiel

$$\ln \left\{ \lim_{n \rightarrow \infty} \left\{ \left[(X^T)^{-1} - (X^{-1})^T \right]! + \frac{1}{n} \right\}^n \right\} + (\sin^2 x + \cos^2 x) = \sum_{i=0}^{\infty} \frac{\cosh x \sqrt{1 - \tanh^2 x}}{2^i}$$

Unter Berücksichtigung folgender Zusammenhänge:

$$(X^T)^{-1} - (X^{-1})^T = 0$$

$$\left[(X^T)^{-1} - (X^{-1})^T \right]! = 1$$

folgt daraus:

$$\ln \left\{ \lim_{n \rightarrow \infty} \left\{ 1 + \frac{1}{n} \right\}^n \right\} + (\sin^2 x + \cos^2 x) = \sum_{i=0}^{\infty} \frac{\cosh z \sqrt{1 - \tanh^2 x}}{2^i}$$

In einem nächsten Schritt wird folgendes berücksichtigt:

$$\left\{ \lim_{n \rightarrow \infty} \left\{ 1 + \frac{1}{n} \right\}^n \right\} = e$$
$$\ln e = 1$$

Endofproofwd - Beispiel

sowie auf der rechten Seite:

$$1 = \cosh x \sqrt{1 - \tanh^2 x}$$

Damit folgt:

$$1 + (\sin^2 x + \cos^2 x) = \sum_{i=0}^{\infty} \frac{1}{2^i}$$

Nun ersetzt man noch folgendes:

$$\begin{aligned} \sin^2 x + \cos^2 x &= 1 \\ \sum_{i=0}^{\infty} \frac{1}{2^i} &= 2 \end{aligned}$$

Damit ergibt sich die allgemein bekannte Beziehung:

$$1 + 1 = 2$$



Und das war's wieder ... Danke
für's Zuhören!
