

Das Paket **robotkarol**

Robot-Karol-Welten und Blockprogramme für Unterrichtsmaterial

Anselm Wagner*

24. September 2026 Version 3.10

Zusammenfassung

Das Paket **robotkarol** zeichnet Welten und Blockprogramme im Stil von *Robot Karol Online* (<https://karol.arrrg.de/>) für Arbeitsblätter, Präsentationen und Prüfungen. Welten können als 3D-Schrägbild oder als 2D-Draufsicht ausgegeben werden; Farben und Formen sind der Web-oberfläche nachempfunden. Einzelne Blöcke lassen sich als Inline-Blöcke in den Fließtext setzen, Aufgaben der Online-Umgebung mit `\karolonline` verlinken. Ein Druckmodus stellt alle Ansich-ten auf tonersparende Graustufen um. Das Paket benötigt Lua¹TeX und expl3. *English:* draws worlds and Blockly-style block programs of the German teaching environment Robot Karol for worksheets; 3D oblique or 2D top view; single blocks can be set inline in running text, aligned to the surrounding baseline; a toner-saving greyscale print mode; requires LuaLaTeX.

Inhaltsverzeichnis

1	Schnellstart	2
2	Die Umgebung <code>karolwelt</code>	2
2.1	Koordinatensystem	2
2.2	Schlüssel der Umgebung	3
2.3	Objekte	3
2.4	Beispiel in beiden Ansichten	3
2.5	Start- und Zielzustände	4
3	Einzeilige Welten: <code>\karolzeile</code>	5
3.1	Schlüssel	6
4	Blockprogramme	6
4.1	Bedingungen mit <code>Anzahl</code>	7
4.2	Himmelsrichtungen	7
5	Inline-Blöcke im Fließtext	8
6	Text-Programme und Struktogramme	9
7	Globale Einstellungen: <code>\karolsetup</code>	9
7.1	Schriften	9

*anselm.wagner@posteo.de

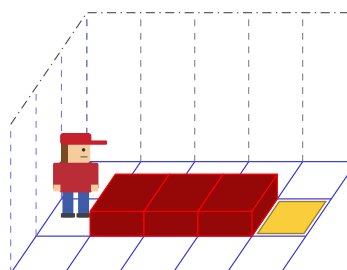
¹Genauer: eine Engine mit `\directlua`; getestet mit LuaLaTeX.

7.2	Weltfarben	10
7.3	Figur der Welten	10
7.4	Blockfarben	11
7.5	Druckmodus	11
8	Aufgaben verlinken: \karolonline	12
8.1	Mit oder ohne hyperref	12
9	Hinweise und Migration	12

1 Schnellstart

```
\usepackage{robotkarol} % Kompilierung mit LuaLaTeX
```

```
\begin{karolwelt}[breite=5, laenge=3]
  \karol[x=1, y=2, richtung=osten]
  \ziegel[x=2, y=2, bisx=4]
  \marke[x=5, y=2]
\end{karolwelt}
```



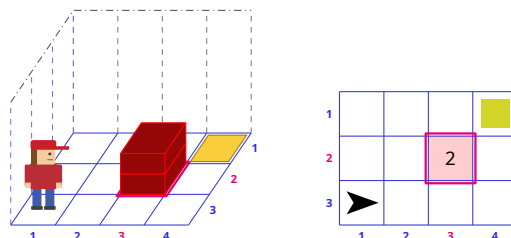
2 Die Umgebung karolwelt

```
\begin{karolwelt}[<Schlüssel>]
  <Objekte>
\end{karolwelt}
```

2.1 Koordinatensystem

Die Koordinaten folgen Robot Karol Online: x zählt die *Spalten* von links (1 bis **breite**), y die *Zeilen von hinten* (1 bis **laenge**). In der 2D-Draufsicht ist die Zeile $y = 1$ folglich die *oberste* Zeile; in der 3D-Ansicht liegt sie am weitesten hinten. **hoehe** bezeichnet wie in Karol Online die *Raumhöhe* (Anzahl Ziegellagen bis zum oberen Rahmen) und wirkt nur auf den 3D-Rahmen.

Mit **nummern** schreibt die Umgebung die Koordinaten an den Rand: die Spaltennummern vor die vorderste Gitterkante, die Zeilennummern in der Draufsicht links daneben. Im Schrägbild stehen die Zeilennummern *rechts* – links lägen sie hinter den Stapeln der ersten Spalte, rechts verdeckt sie nichts. Gezählt wird ab 1, es stehen also genau die Zahlen am Rand, die man in x und y einträgt. Eine hervorgehobene Zelle färbt ihre beiden Nummern mit.



2.2 Schlüssel der Umgebung

breite	5	Anzahl Spalten
laenge	5	Anzahl Zeilen
hoehe	6	Raumhöhe des 3D-Rahmens in Ziegellagen
ansicht	3d	3d (Schrägbild) oder 2d (Draufsicht)
zustand	beides	start, ziel oder beides (Abschnitt 2.5)
auftrag	keiner	aufbauen oder aufraeumen: die Start- bzw. Zielwelt ist leer (Abschnitt 2.5)
nummern	false	Koordinaten an den Rand schreiben
rahmen	true	Raumrahmen zeichnen (nur 3D)
hervorheben		Liste hervorzuhebender Zellen, etwa {2/2, 4/1}
skalierung	1	Skalierungsfaktor des Bildes
titel		Beschriftung über dem Bild
hinweis		kursiver Hinweis unter dem Bild

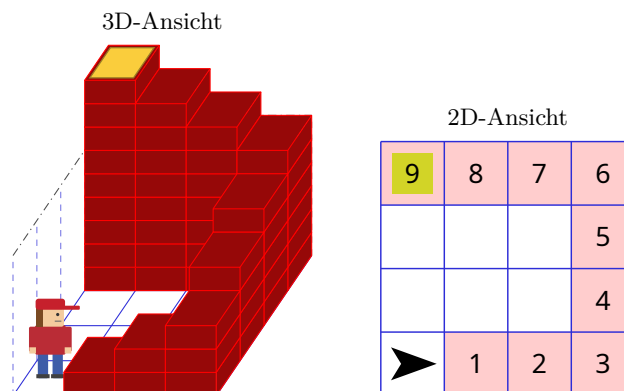
2.3 Objekte

Alle Objekte werden über Schlüssel platziert. `bisx/bisy` (auch `bisX/bisY`) füllen rechteckige Bereiche.

```
\karol[x=1, y=1, richtung=osten] % genau einmal je Welt
\ziegel[x=2, y=3, anzahl=4]      % Stapel aus 4 Ziegeln
\ziegel[x=1, y=5, bisx=5]        % Reihe einzelner Ziegel
\marke[x=3, y=3]                 % gelbe Marke
\marke[x=5, y=1, farbe=cyan!60]  % Marke in eigener Farbe
\quader[x=4, y=1, bisy=5]        % Quaderwand; \wand = Synonym
```

`richtung` akzeptiert `osten`, `westen`, `norden` und `sueden` sowie die Synonyme `rechts`, `links`, `hinten` bzw. `oben` und `vorn`, `vorne` bzw. `unten` (Vorgabe: `sueden`). Marken liegen stets *oben auf* dem Ziegelstapel ihrer Zelle; Karol steht auf Stapel und Marke.

2.4 Beispiel in beiden Ansichten



In der 2D-Ansicht zeigen Zahlen die Stapelhöhe; Ziegelzellen sind rosa, Marken eine einfarbige Fläche, die kleiner als das Feld ist — so bleibt der Ziegel ringsum sichtbar und „Marke auf Ziegel“ erkennbar. Quader sind grau, Karol ist der schwarze Richtungskeil.

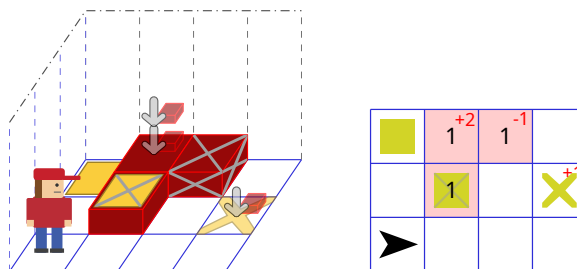
2.5 Start- und Zielzustände

Aufgaben in Robot Karol Online zeigen oft in *einem* Bild, was sich ändern soll: Ziegel und Marken, die bleiben, sehen aus wie immer; was verschwinden soll, ist durchgekreuzt; was entstehen soll, erscheint blass mit einem Pfeil bzw. als X-Fläche. Dasselbe leisten die Schlüssel **start** und **ziel** an den Objekten.

start	weg	nur im Startzustand: soll verschwinden
ziel	neu	nur im Zielzustand: soll entstehen
startanzahl		Ziegel im Startzustand
zielanzahl		Ziegel im Zielzustand

Wer beides oder keines von beiden nennt, bekommt ein Objekt, das in beiden Zuständen vorkommt — unverändert also. Statt **start** und **ziel** darf man die Befehlsnamen von Robot Karol schreiben: **hinlegen** und **aufheben** für Ziegel, **setzen** und **loeschen** für Marken. **startanzahl** und **zielanzahl** gibt es auch mit Bindestrich (**start-anzahl**). Die Reihenfolge der Schlüssel spielt keine Rolle: [**anzahl**=3, **ziel**] und [**ziel**, **anzahl**=3] bedeuten dasselbe.

```
\begin{karolwelt}[breite=4, laenge=3]
  \karol[x=1, y=3, richtung=osten]
  \ziegel[x=2, y=2] % bleibt liegen
  \ziegel[x=3, y=1, aufheben] % soll aufgehoben werden
  \ziegel[x=4, y=2, hinlegen] % soll hingelegt werden
  \ziegel[x=2, y=1, startanzahl=1, zielanzahl=3] % zwei kommen dazu
  \marke[x=1, y=1] % bleibt liegen
  \marke[x=2, y=2, loeschen] % soll geloescht werden
  \marke[x=4, y=2, setzen] % soll gesetzt werden
\end{karolwelt}
```



 soll hingelegt werden
  soll aufgehoben werden
  soll gesetzt werden
  soll gelöscht werden

In der 2D-Draufsicht gibt es keine Kreuze und keine Pfeile: dort steht die Stapelzahl des Startzustands mittig und die Veränderung als rote Zahl in der Ecke. Auch das entspricht der Vorlage.

Der Schlüssel **zustand** wählt, was gezeigt wird. **beides** ist die Vorgabe und liefert das Bild oben; **start** und **ziel** zeigen je *eine* Welt in der gewohnten Optik — eine Welt ohne **start**- und **ziel**-Angaben sieht in allen drei Fällen gleich aus.

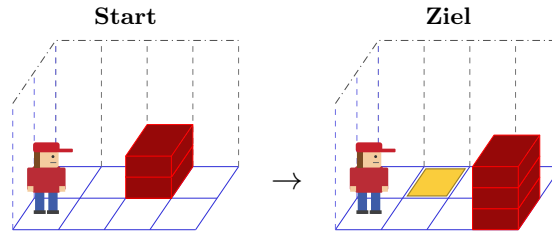
Für „vorher / nachher“ nebeneinander gibt es die Umgebung **karolweltpaar**. Ihr Rumpf wird zweimal gezeichnet, links im Zustand **links** (Vorgabe **start**), rechts im Zustand **rechts** (Vorgabe **ziel**); dazu **trenner**, **titellinks** und **titelrechts**. Alle übrigen Schlüssel reicht sie an beide Welten durch.

```
\begin{karolweltpaar}[breite=4, laenge=2]
  \karol[x=1, y=2, richtung=osten]
```

```

\ziegel[x=3, y=1, anzahl=2, aufheben]
\ziegel[x=4, y=2, anzahl=3, hinlegen]
\marke[x=2, y=1, setzen]
\end{karolweltpaar}

```



Soll am Ende *alles* weggeräumt sein, wäre an jedem Objekt ein **weg** nötig — und ein vergessenes fiele nicht auf. Umgekehrt bräuchte beim Bau einer Mauer jeder Ziegel ein **neu**. Dafür gibt es den Schlüssel **auftrag**. Er legt fest, wozu ein Objekt *ohne* eigene Angabe gehört:

keiner zu Start und Ziel (Vorgabe)
aufraeumen nur zum Start – die Zielwelt ist leer (auch **aufräumen**)
aufbauen nur zum Ziel – die Startwelt ist leer

Wer an einem Objekt **start**, **ziel** oder eine Anzahl angibt, behält seine Angabe. Quader und Karol betrifft **auftrag** nicht: Wände stehen in beiden Zuständen.

```

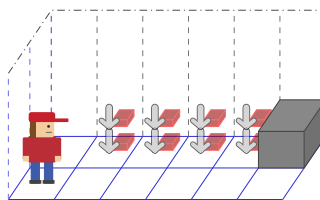
\begin{karolwelt}[breite=6, laenge=3, auftrag=aufraeumen]
  \ziegel[x=2, y=2, bisx=5, anzahl=2] % alles ohne Angabe = nur Start
  \marke[x=3, y=3, start, ziel] % Ausnahme: diese Marke bleibt
\end{karolwelt}

```

```

\begin{karolwelt}[breite=6, laenge=2, auftrag=aufbauen]
  \karol[x=1, y=2, richtung=osten]
  \ziegel[x=2, y=1, bisx=5, anzahl=2] % die Mauer entsteht erst
  \quader[x=6, y=1] % Wand: steht schon
\end{karolwelt}

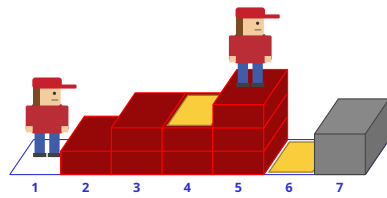
```



Nicht ausdrückbar sind eine Zielposition für Karol und ein Farbwechsel einer Marke: **farbe** gilt für beide Zustände, und Robot Karol Online vergleicht die Figur beim Prüfen ohnehin nicht. Quader kennen keinen Zielzustand — Karol kann sie weder setzen noch räumen.

3 Einzeilige Welten: `\karolzeile`

Für schnelle eindimensionale Welten gibt es die kompakte Zeichenketten-Notation (bis Version 2.x hieß das Kommando `\karolwelt`). Gezeichnet wird mit denselben Bausteinen wie in der Umgebung `karolwelt`: gleiche Projektion, gleiches Bodengitter, gleiche Ziegel, Marken, Quader und derselbe Avatar. Eine Zeilen-Welt ist damit optisch nichts anderes als eine `karolwelt` mit `laenge=1` – nur die Eingabe ist kürzer.



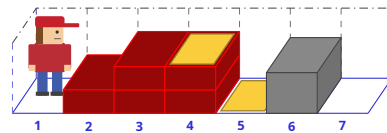
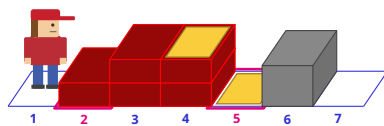
`\karolzeile{> Z 2 {2M} {3>} M Q}`

`> < ^ v` Karol mit Blickrichtung
`.` leeres Feld
`Z, 2...9` Ziegel bzw. Stapel
`M, Q` Marke, Quader
`{...}` mehrere Objekte auf einem Feld, von unten nach oben: `{M>}`, `{2M}`, `{3>}`

3.1 Schlüssel

skalierung 1 Skalierungsfaktor des Bildes
nummern true Feldnummern anzeigen (ab 1)
titel — Zeile über dem Bild
hinweis — kursive Zeile unter dem Bild
hervorheben — Komma-Liste von Feldnummern
rahmen false Raumrahmen wie in `karolwelt`
hoehe 3 Raumhöhe des Rahmens in Ziegellagen

Die **Feldnummern** stehen vor der vordersten Gitterkante und nicht auf dem Feld: im Schrägbild würde sie sonst jeder Ziegelstapel verdecken. Sie zählen wie **x** in `karolwelt` ab 1; **hervorheben** nimmt dieselben Nummern. **hervorheben** zeichnet einen magentafarbenen Rahmen knapp außerhalb des Feldes und färbt dessen Nummer mit – ein roter Rahmen wäre neben den reinroten Ziegelkanten nicht zu erkennen (Farbe: **hervorfarbe**). Den Raumrahmen zeichnet `\karolzeile` nur auf Wunsch; für eine einzelne Reihe ist er meist zu wuchtig.



Dazu gibt es

`\karolpaar[keys][keys2]{vorher}{nachher}` Vorher-nachher-Paar; **keys** gilt für beide Bilder, **keys2** nur für „nachher“
`\karolgitter[keys]{n}` leeres Gitter aus *n* Feldern
`\karollegende` Standardlegende der Weltsymbole
`\karollegendenitem{Welt}{Text}` einzelner Legendeneintrag

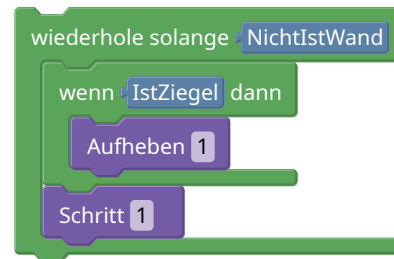
4 Blockprogramme

Die Umgebung `karolbloecke` stapelt Blockly-Blöcke; Farben und Formen entsprechen der Karol-Online-Oberfläche.

```

\begin{karolbloecke}
  \bwiederholesolange{NichtIstWand}{%
    \bwenndann{IstZiegel}{%
      \bbefehl[1]{Aufheben}}
    \bbefehl[1]{Schritt}}
\end{karolbloecke}

```



<code>\bbefehl[n]{Name}</code>	Anweisung (violett); optionale Zahl als Pille
<code>\bwiederholemal{n}{...}</code>	Zählschleife
<code>\bwiederholesolange{Bed}{...}</code>	kopfgesteuerte Schleife
<code>\bwiederholeimmer{...}</code>	Endlosschleife
<code>\bwenndann{Bed}{...}</code>	einseitige Verzweigung
<code>\bwenndannsonst{Bed}{...}{...}</code>	zweiseitige Verzweigung
<code>\bbedingung[n]{IstZiegel}</code>	Bedingung mit linkem Stecker; auch im Fließtext; optionale Anzahl als Pille; leeres Argument = weißer Leersockel
<code>\bist{Norden}</code>	Himmelsrichtung als Auswahlmenü
<code>\bnichtist{Norden}</code>	dasselbe verneint
<code>\bluecke[Breite]</code>	gestrichelter Geisterblock
<code>\bkommentar{Text}</code>	Kommentar (khaki, //)
<code>\baufruf[n]{Name}</code>	Aufruf einer eigenen Anweisung (rot)
<code>\banweisung{Name}{...}</code>	Definition einer eigenen Anweisung
<code>\bhauptprogramm</code>	Hauptprogramm-Hut

4.1 Bedingungen mit Anzahl

In Robot Karol Online nehmen `IstZiegel` und `NichtIstZiegel` eine Anzahl entgegen – im Blockeditor ein Zahlenfeld, im Programmtext `IstZiegel(2)`. Im Paket steht sie als optionales Argument vor der Bedingung; die Pille trägt kein Auswahldreieck, denn im Original ist es ein Zahlenfeld und kein Menü.

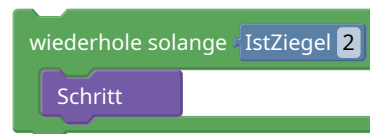


Die Zahl gehört der Bedingung, nicht dem Block. Die Kontrollblöcke reichen sie als erstes, optionales Argument an ihre Bedingung durch – die Pille bleibt darum auch im grünen `\bwenndann` stahlblau.

```

\bwiederholesolange[2]{IstZiegel}{%
  \bbefehl{Schritt}}

```



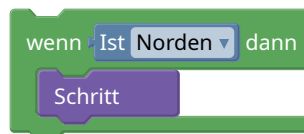
4.2 Himmelsrichtungen

Die acht Blöcke `Ist/NichtIst` zeigen die Richtung in Robot Karol Online nicht als Text, sondern als Auswahlmenü: eine helle Pille mit Auswahldreieck. `\bist{...}` und `\bnichtist{...}` bilden das nach und nehmen die Richtung als Argument.

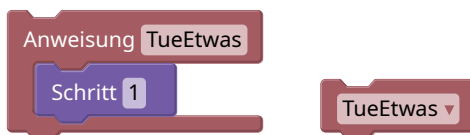


Für sich allein ergeben sie ein vollständiges Bedingungsplättchen. Als Argument eines Kontrollblocks steuern sie nur dessen Beschriftung bei – das Plättchen baut dort wie gewohnt `\bbedingung`:

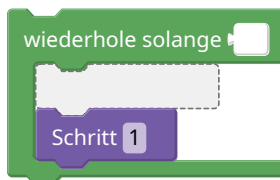
```
\bwenndann{\bist{Norden}}{%  
  \bbefehl{Schritt}}
```



Dasselbe Auswahldreieck trägt der Aufrufblock `\baufruf`: dort wählt man in der Oberfläche die aufzurufende Anweisung aus. Die Definition `\banweisung` zeigt den Namen dagegen in einer Pille ohne Dreieck.



Beispiel Lückenaufgabe mit Leersockel und Geisterblock:



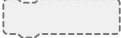
5 Inline-Blöcke im Fließtext

Die `\i...`-Kommandos setzen genau *einen* Block in die laufende Zeile. Die Textgrundlinie des Blocks liegt dabei exakt auf der Grundlinie des umgebenden Textes, und die Beschriftung übernimmt dessen Schriftgröße; Kerbe, Nase und Innenabstände sind flacher als im Stapel, damit die Blockhöhe der Zeilenhöhe nahekommt.

Mit `\iwenndann{IstZiegel}` prüft Karol, ob ein Ziegel vor ihm liegt;
`\iwiederholemal{5}` wiederholt fünfmal, `\ibefehl{Schritt}` geht weiter.

Mit `wenn IstZiegel dann` prüft Karol, ob ein Ziegel vor ihm liegt; `wiederhole 5 mal` wiederholt fünfmal, `Schritt` geht weiter.

Blöcke, die im Stapel mehrzeilig wären – Schleifen, Verzweigungen und eigene Anweisungen –, erscheinen inline **nur mit ihrem Kopf**. `\iwenndann{IstZiegel}` ergibt also den Block `wenn IstZiegel dann` ohne Rumpf. Für echte Rümpfe bleibt die Umgebung `karolbloecke` zuständig.

<code>\ibefehl[n]{Name}</code>	Anweisung Schritt 3
<code>\iaufruf[n]{Name}</code>	Aufruf einer eigenen Anweisung
<code>\ianweisung{Name}</code>	Kopf einer eigenen Anweisung
<code>\ikommentar{Text}</code>	Kommentar
<code>\ihauptprogramm</code>	Hauptprogramm-Hut
<code>\iwiederholemal{n}</code>	Kopf der Zählschleife wiederhole 5 mal
<code>\iwiederholesolange[n]{Bed}</code>	Kopf der kopfgesteuerten Schleife
<code>\iwiederholeimmer</code>	Kopf der Endlosschleife
<code>\iwenndann[n]{Bed}</code>	Kopf der Verzweigung
<code>\isonst</code>	sonst-Balken sonst
<code>\ibedingung[n]{Bed}</code>	Bedingung IstMarke , mit Anzahl IstZiegel 2
<code>\iist{Norden}</code>	Himmelsrichtung Ist Norden
<code>\inichtist{Norden}</code>	verneint NichtIst Osten
<code>\iluecke[Breite]</code>	Geisterblock 

Die `\b...`-Kommandos erkennen den Fließtext selbst und setzen dort ebenfalls inline; die `\i...`-Fassungen ersparen nur die leeren Rumpf-Argumente und erzwingen den Modus auch im Stapel.

Inline-Blöcke sind breite, unzerbrechliche Kästen. In schmalen Spalten kann T_EX sie nicht immer umbrechen – dann hilft `\sloppy` oder eine höhere `\tolerance` für den Absatz.

6 Text-Programme und Struktogramme

Die Umgebung `karolcode` setzt Karol-Quelltext mit Syntaxhervorhebung (`listings`); `\kb{...}` und `\kw{...}` formatieren Bezeichner und Schlüsselwörter im Fließtext. Für Struktogramme lädt das Paket `struktex` und ergänzt `\leeranweisung` sowie `\stklucke[Breite]`.

7 Globale Einstellungen: `\karolsetup`

```
\karolsetup{felddbreite=8.4mm, befehlfarbe={RGB}{116,91,165}, ...}
```

7.1 Schriften

Die Oberfläche von Robot Karol Online setzt ihre Beschriftungen in *Noto Sans* und den Programmtext des Editors in *Hack*. Das Paket tut das seit v3.6 ebenso: Beschriftungen der Blöcke, Feldnummern und Stapelzahlen erscheinen in Noto Sans, `karolcode` sowie `\kb` und `\kw` in Hack. Dafür lädt das Paket `fontspec`. Noto Sans liegt in T_EX Live; Hack nicht – ist es im System installiert, wird es benutzt, sonst DejaVu Sans Mono, aus dessen Vorläufer Hack hervorgegangen ist. Welche der beiden es geworden ist, steht in der `.log`-Datei.

Wer die Schriften des eigenen Dokuments übernehmen will – oder `fontspec` nicht geladen haben möchte –, lädt das Paket mit

```
\usepackage[schrift=dokument]{robotkarol}
```

Dann benutzt das Paket `\sffamily` und `\ttfamily` des Dokuments; die Ausgabe ist Zeichen für Zeichen dieselbe wie bis v3.5. Umschalten geht auch mitten im Dokument, in einer Gruppe nur dort:

```
schrift=karol      Noto Sans und Hack (Vorgabe)
schrift=dokument   \sffamily und \ttfamily des Dokuments
textschrift={...}  eigene Schrift für Beschriftungen
codeschrift={...}  eigene Schrift für Programmtext
```

Die beiden letzten Schlüssel nehmen den Umschaltbefehl selbst:

```
\karolsetup{textschrift={\fontspec{Fira Sans}}}
```

Ein `\karolsetup{schrift=karol}` in einem Dokument, das mit `schrift=dokument` geladen wurde, ist wirkungslos (mit Warnung) – die Schriften sind dann gar nicht geladen.

Titel und Hinweis einer Welt bleiben absichtlich in der Schrift des Dokuments: sie sind Text des Dokuments, keine Beschriftung der Zeichnung. Für eigenen Text in der Paketschrift gibt es die Umschalter `\karolschrift` und `\karolcodeschrift`.

7.2 Weltfarben

Sie wirken auf `\karolzeile` und die Umgebung `karolwelt`, die dieselben Bausteine zeichnen.

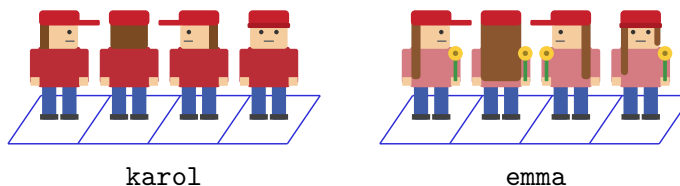
<code>feldbreite</code>	8.4 mm	Zell-/Feldbreite aller Welten
<code>ziegelfarbe</code>	reines Rot	Lagenkante; Körper daraus abgeleitet (!59!black)
<code>ziegelkoerperfarbe</code>	150,8,8	nur die Ziegelflächen
<code>ziegelkantenfarbe</code>	255,0,0	nur die Lagenkanten
<code>markenfarbe</code>	250,205,60	Markenplatte
<code>quaderfarbe</code>	128,128,128	Quader / Wand
<code>bodenfarbe, gitterfarbe</code>	45,45,215	Bodengitter
<code>hervorfarbe</code>	226,0,122	hervorheben

7.3 Figur der Welten

Das Paket bringt zwei Figuren mit: `karol` (Vorgabe) und `emma` mit langem Haar und einer Blume in der Hand. Umgeschaltet wird mit

```
\karolfigur{emma}           % gleichwertig: \karolsetup{figur=emma}
```

Die Einstellung wirkt auf `\karolzeile` und die Umgebung `karolwelt`; in einer Gruppe gesetzt gilt sie nur dort.



Emma ist aus derselben Figur gebaut wie Karol – sie ist ein Nachbau, keine Kopie. Ihre Farben heißen `karolEmmaShirt`, `karolEmmaHaar`, `karolEmmaBluete` und `karolEmmaStiel` und lassen sich mit `\colorlet` ändern.

7.4 Blockfarben

befehlfarbe	116,91,165	Anweisungsblöcke
schleifenfarbe, verzweigungsfarbe	91,165,91	grüne Kontrollblöcke
bedingungsfarbe	91,128,165	Bedingungen
anweisungsfarbe	165,91,98	eigene Anweisungen
kommentarfarbe	165,150,91	Kommentare
hutfarbe	130,123,117	Hauptprogramm-Hut

Die Blockfarben sind aus der Karol-Online-Oberfläche übernommen.

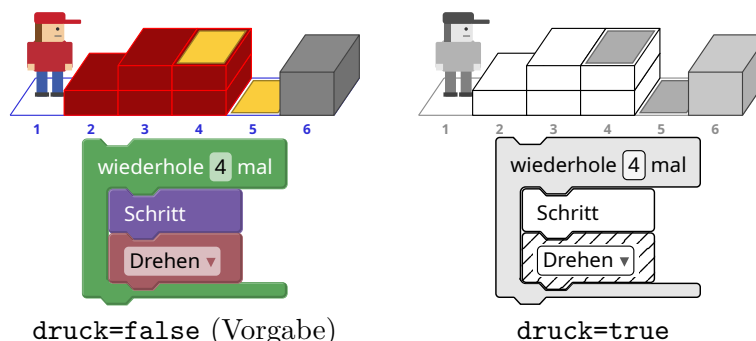
7.5 Druckmodus

Arbeitsblätter werden selten in Farbe vervielfältigt. Der Druckmodus zeichnet deshalb alle Ansichten – Welten, Zeilen-Welten und Blockprogramme – in Graustufen:

```
\usepackage[druck]{robotkarol} % das ganze Dokument
\karolsetup{druck=true}        % ab hier; in einer Gruppe nur dort
\karolsetup{druck=false}       % zurueck zur Farbdarstellung
```

Der Leitgedanke ist *nicht* das absolute Minimum an Toner, sondern: so wenig wie möglich, so viel wie nötig. Was eine Aufgabe trägt, muss erkennbar bleiben. Deshalb tragen die Konturen die Information, und die Flächen bleiben so hell wie möglich:

- Ziegel sind *weiß* mit schwarz umrandeten Lagen – der Stapel bleibt seitlich abzählbar, kostet aber praktisch keine Farbe.
- Der Quader ist die einzige große graue Fläche der Welt: er teilt seine Form mit dem Ziegel und muss sich von ihm abheben.
- Die Marke ist mittelgrau. Sie ist klein, und sie liegt oft auf einem Ziegel, von dem sie sich absetzen muss.
- Bei den Blöcken bekommt der häufigste – `\bbefehl` – die weiße Fläche; die formgleichen `\baufruf` und `\banweisung` bekommen eine **45-Grad-Schraffur** auf weißem Grund. Ein bloß dunkleres Grau trug diesen Unterschied nicht – beide Blöcke haben dieselbe Form, im Original unterscheidet sie allein die Farbe (violett gegen rot). Beschriftung und Namenspille stehen dabei auf weißem Grund; schraffiert ist nur der Hintergrund. Nebenbei braucht die Schraffur weniger Farbe als eine graue Fläche. Alles übrige unterscheidet sich ohnehin schon durch die Form (C-Maul, Hut, Bedingungsplättchen).
- An die Stelle des 3D-Anschliffs tritt eine schwarze Kontur, und die Eingabepillen bekommen einen Rand: auf einem weißen Block wären sie sonst nicht zu sehen.



Die Farbschlüssel aus den vorigen Abschnitten bleiben dabei unangetastet: sie wirken wieder, sobald der Druckmodus aus ist – im Druckmodus selbst haben sie keine Wirkung, denn eigene Farben würden das Graustufenschema gerade aufheben. Eigene *Markenfarben* (`\marke[x=2, y=1, farbe=cyan]`) werden in ihren Grauwert umgesetzt, bleiben also als verschiedene Grautöne unterscheidbar. Die beiden Farben, die es nur im Druck gibt, heißen `karolDruckKontur` (Blockkonturen und Pillenränder) und `karolDruckAkzent` (Auswahldreieck der Pillen); sie lassen sich mit `\definecolor` ändern.

8 Aufgaben verlinken: `\karolonline`

Eine Aufgabe in Robot Karol Online steckt vollständig in ihrer Adresse: auf <https://karol.arrrg.de/> folgt eine Raute und der Aufgabencode. `\karolonline` macht daraus einen Verweis und setzt dabei nur, was man tatsächlich eintippen muss:

Die Aufgabe steht unter `\karolonline{2t7t}`.

Das ergibt: Die Aufgabe steht unter karol.arrrg.de/#2t7t. Der Verweis führt auf <https://karol.arrrg.de/#2t7t>. Der Code wird dabei unverändert übernommen – auf Groß- und Kleinschreibung kommt es bei den Aufgabencodes nicht an.

8.1 Mit oder ohne `hyperref`

Der Verweis braucht `hyperref`. Ob `robotkarol` es lädt, entscheidet eine Paketoption:

`hyperref=true` `hyperref` wird geladen (Vorgabe)
`hyperref=false` `hyperref` wird nicht geladen

Ohne `hyperref` setzt `\karolonline` denselben Text, nur ohne Verweis; das Dokument übersetzt in beiden Fällen unverändert.

Geladen wird erst am Ende der Präambel (im Haken `begindocument/before`) – `hyperref` möchte als letztes Paket geladen werden. Ein Dokument darf es deshalb auch selbst laden, mit eigenen Optionen und an beliebiger Stelle, ohne Optionskonflikt:

```
\usepackage{robotkarol}  
\usepackage[colorlinks]{hyperref}
```

Geladen wird dabei ohne Optionen: wie Verweise aussehen – farbig, umrahmt oder unsichtbar –, stellt das Dokument selbst ein, etwa mit `\usepackage[hidelinks]{hyperref}` oder `\hypersetup{...}`.

Und weil `\karolonline` nicht die Paketoption abfragt, sondern ob es `\href` beim Setzen wirklich gibt, verlinkt es auch in einem Dokument, das `hyperref` trotz `hyperref=false` selbst geladen hat.

Ein Verweis ist ein unteilbarer Block. Steht er in einer schmalen Spalte, braucht der Absatz unter Umständen ein `\sloppy` (wie die Inline-Blöcke auch).

9 Hinweise und Migration

- Das Paket benötigt **LuaLaTeX**.
- Bis v2.x hieß das Zeichenketten-Kommando `\karolwelt`; ab v3.0 heißt es `\karolzeile`, der Name `karolwelt` bezeichnet die neue Umgebung. Bestehende Dokumente: einfaches Suchen/Ersetzen.

- Die kurzen Objektnamen `\karol`, `\ziegel`, `\marke`, `\quader` und `\wand` sind nur *innerhalb* der Umgebung `karolwelt` belegt. Ein Dokument darf sie also selbst verwenden – außerhalb der Umgebung behalten sie ihre eigene Bedeutung.
- In v3.10 hat sich dreierlei geändert, ohne Rücksicht auf ältere Dokumente. Der Schlüssel `aufraeumen` heißt jetzt `auftrag=aufraeumen`, neu dazu kam `auftrag=aufbauen`. Die Feldnummern von `\karolzeile` zählen ab 1 statt ab 0 – wie die Koordinaten der Umgebung. Damit gilt auch `hervorheben` ab 1: aus `hervorheben={2,4}` wird `hervorheben={3,5}`. Und `karolwelt` beginnt keinen neuen Absatz mehr: zwei Welten mit `\quad` dazwischen stehen jetzt nebeneinander statt untereinander. Wer sie untereinander will, trennt sie durch eine Leerzeile.
- Vorgegeben ist die Figur `karol`. In v3.5 war für kurze Zeit `emma` die Vorgabe; wer sie behalten will, schreibt in die Präambel ein `\karolfigur{emma}`.
- Neu in v3.7 ist außerdem `\karolonline` für Verweise auf Aufgaben der Online-Umgebung (Abschnitt 8). **Dafür lädt das Paket jetzt `hyperref`** – wer das nicht will, lädt es mit `hyperref=false` (Abschnitt 8.1).
- Neu in v3.7 ist der Druckmodus (`\usepackage[druck]{robotkarol}`, Abschnitt 7.5). Ohne ihn ist die Ausgabe dieselbe wie in v3.6 – mit einer Ausnahme: die *Marke* sitzt jetzt in beiden Modi mittig auf ihrem Feld. Bis v3.6 fehlte in ihren Bildkoordinaten der Tiefenanteil, sie saß rund 0,3 mm zu weit links und berührte auf einem Ziegel dessen linke Kante.
- Seit v3.6 setzt das Paket seine Beschriftungen in den Schriften von Robot Karol Online und lädt dafür `fontspec`. Wer die bisherige Ausgabe will, lädt es mit der Paketoption `schrift=dokument` – siehe Abschnitt 7.1.

Lizenz

`robotkarol` steht unter der L^AT_EX Project Public License, Version 1.3c oder später (<https://www.latex-project.org/lppl.txt>), mit dem Wartungsstatus *maintained*. Copyright © 2026 Anselm Wagner.

Farben und Proportionen sind aus der Oberfläche von *Robot Karol Online* gemessen und mit TikZ nachgebaut; es wurden keine Grafiken übernommen.