syntax.tex
Eine Übersicht

Bernd Worsch

7. Juli 1997

Inhaltsverzeichnis

1 Einleitung

Zur Beschreibung von Zusammenhängen in formal sprachlichen Systemen, findet häufig die sogenannte Bachus–Naur–Form kurz BNF Verwendung. Von dieser (textlichen) Darstellung ausgehend wird gerade im Zusammenhang mit höheren Programmiersprachen gerne eine grafische Beschreibung zur Verdeutlichung herangezogen. Solche *Syntaxdiagramme* sind intuitiv verständlich und verdeutlichen auf anschauliche Weise den formalen Aufbau der jeweiligen Programmiersprache.

Das File syntax.tex ermöglicht das einfache Erstellen solcher Diagramme in LaTeX-Dokumenten. Statt mit geometrischen Objekten (Linien, Kreise, etc.) zu arbeiten, nutzt der Anwender dazu Befehle, die schon im Quelldokument die zu illustrierenden Kontrollstrukturen wiederspiegeln.

2 Bevor es funktioniert ...

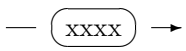
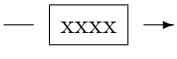
Um die Möglichkeiten von syntax.tex nutzen zu können muß die Datei vor dem ersten zu setzenden Syntaxdiagramm durch `\input{syntax.tex}` in das eigene Dokument eingefügt werden. Da es in der vorliegenden Version 0.02 noch auf einen Befehl des Pakets fancybox.sty zurückgreift, muß dieses am Anfang der eigenen Datei aktiviert werden. Wie für alle Pakete geschieht dies durch den Befehl: `\usepackage{fancybox}`.

3 Grundelemente von syntax.tex

Abgesehen von Hilfskonstrukten, insbesondere Linien und Pfeilen, bestehen Syntaxdiagramme aus Terminalsymbolen, den elementaren Grundbausteinen einer Programmiersprache, und Syntaxvariablen. Letztere stehen als Platzhalter für syntaktische Konstruktionen.

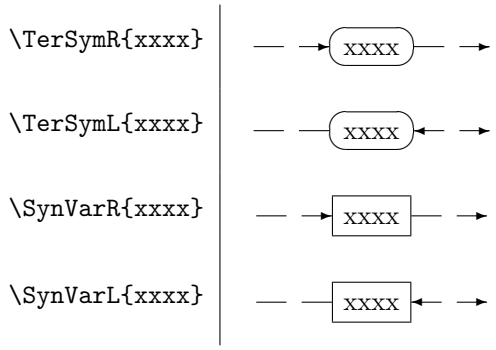
Beide Grundelemente stehen innerhalb der Umgebung `Syntaxdiagramm` mit den Befehlen `\TerSym` und `\SynVar` zur Verfügung.

Die Umgebung `Syntaxdiagramm` erzeugt zunächst ein leeres Syntaxdiagramm, also einen von links nach rechts verlaufenden Pfeil. An inneren Befehlen stehen unter anderem elementare Teilstrukturen von Syntaxdiagrammen bereit. Einige haben keine Parameter und erzeugen beispielsweise Linien oder Pfeile, `\TerSym` und `\SynVar` hingegen erhalten als Parameter einen Text. Da diese Befehle stets innerhalb des umgebenden Pfeils auftreten, wurden in der folgenden Übersicht die inneren Elemente durch Leerzeichen von der äußeren Umgebung abgegrenzt. Diese Zwischenräume sind im endgültigen Diagramm unerwünscht. Befehle und Umgebungen in Syntaxdiagrammen sollten daher mit einem Kommentarzeichen (`%`) beendet werden , falls Zeilenumbrüche im Eingabetext vorgenommen werden.

`\begin{Syntaxdiagramm}` `\end{Syntaxdiagramm}`	
`\Linie`	
`\PfeilR`	
`\PfeilL`	
`\Ellipse`	
`\Leer`	
`\TerSym{xxxx}`	
`\SynVar{xxxx}`	

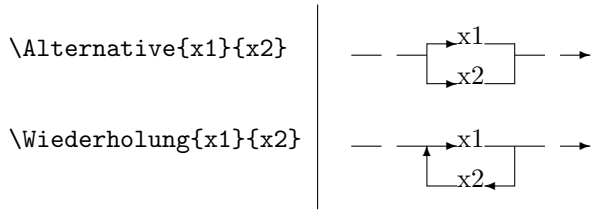
Aus obigen Grundelementen und vertikalen Linien werden alle Syntaxdia-

gramme aufgebaut. Die vertikalen Linien und Pfeile sind prinzipiell unter den Namen `\LinieO{x}`,`\LinieU{x}`,`\PfeilO{x}` und `\PfeilU{x}` ansprechbar, wobei `x` die jeweilige Länge festlegt, sie sind jedoch nur intern notwendig. Um die LaTeX–Beschreibung eines Syntaxdiagramms kompakt zu halten existieren schließlich noch vier abkürzende Schreibweisen. Es sind die zusammengesetzten Grundelemente:



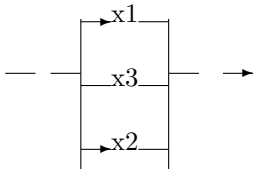
4 Strukturelemente von syntax.tex

Aus den Grundelementen allein lassen sich noch keine (bzw. sehr wenige) Syntaxdiagramme aufbauen, es fehlt die Möglichkeit diese Elemente zu verbinden. Zu diesem Zweck stellt syntax.tex Alternativen und Wiederholungen bereit. Für beide Strukturen existieren zunächst Befehle, die je zwei Parameter übernehmen. Im Fall des Befehls `\Alternative{x1}{x2}` steht `x1` für den Teil des Strukturdiagramms der auf dem ersten Pfad, `x2` für den Teil der auf dem zweiten Pfad durchlaufen wird. Bei `\Wiederholung{x1}{x2}` entspricht `x1` dem wiederholten Teil und `x2` allem, was im zurücklaufenden Pfad auftritt. Die erzeugten Bilder wieder in einer Tabelle:



Für Wiederholungen ist diese Konstruktion ausreichend, bei Alternativen sollten jedoch mehr als zwei Pfade möglich sein, dies wird durch die Umgebung `Alternativen` realisiert. Sie übernimmt ähnlich obigen Befehlen zwei Parameter. `x1` enthält wie gehabt den ersten Pfad, `x2` hingegen den letzten der möglichen Pfade. Alle weiteren Pfade werden innerhalb der Umgebung getrennt durch `\\` angegeben, also wie in einer einspaltigen `\tabular`–Umgebung, wobei auch auf die letzte Zeile ein abschließendes `\\` folgt.

```
\begin{Alternativen}
  {x1}{x2}
  x3\\
\end{Alternativen}
```



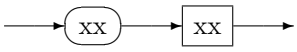
5 Setzen von Syntaxdiagrammen

Praktisch geht man nun so vor, daß innerhalb der äußeren Umgebung im einfachsten Fall eine Folge von Syntax–Variablen und Terminal–Symbolen aufgelistet wird. Hierzu verwendet man die nach rechts gerichteten Varianten.

```
\begin{Syntaxdiagramm}%
  \TerSymR{xx}%
\end{Syntaxdiagramm}
```

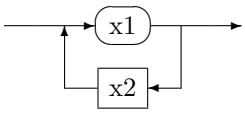

```
\begin{Syntaxdiagramm}%
  \SynVarR{xx}%
\end{Syntaxdiagramm}
```


```
\begin{Syntaxdiagramm}%
  \TerSymR{xx}%
  \SynVarR{xx}%
\end{Syntaxdiagramm}
```

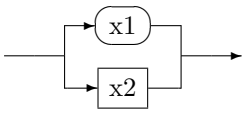


Etwas anspruchsvoller können Syntax–Variablen und Terminal–Symbole in Strukturen auftreten. In diesem Fall sorgt die umgebende Struktur für die notwendigen Pfeile, man verwendet also die ungerichteten Befehle `\TerSym{}` und `\SynVar{}`:

```
\begin{Syntaxdiagramm}%
  \Wiederholung%
    {\TerSym{x1}}%
    {\SynVar{x2}}%
\end{Syntaxdiagramm}
```

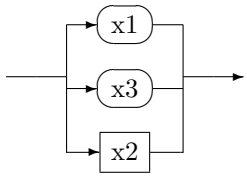



```
\begin{Syntaxdiagramm}%
  \Alternative%
    {\TerSym{x1}}%
    {\SynVar{x2}}%
\end{Syntaxdiagramm}
```



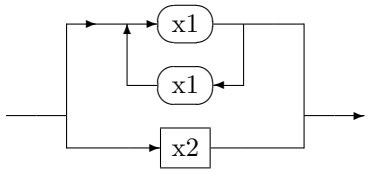
In der `Alternativen`–Umgebung müßen die inneren Pfade selbst für ihre Pfeile sorgen (hier: Terminal–Symbol x3).

```
\begin{Syntaxdiagramm}%
  \begin{Alternativen}%
    {\TerSym{x1}}%
    {\SynVar{x2}}%
    \TerSymR{x3}\\%
  \end{Alternativen}%
\end{Syntaxdiagramm}
```



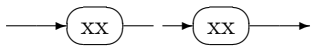
Schließlich können auch Strukturen beliebig kombiniert und verschachtelt werden:

```
\begin{Syntaxdiagramm}%
  \Alternative%
    {\Wiederholung%
      {\TerSym{x1}}%
      {\TerSym{x1}}}%
    {\SynVar{x2}}%
\end{Syntaxdiagramm}
```



Die vielen Kommentarzeichen (%) in der Eingabe sind nicht an allen Stellen notwendig. Es ist aber in jedem Fall einfacher alle Zeilen mit einem % zu beenden, als jeweils zu überlegen, ob es eventuell weggelassen werden könnte. Sollte ein notwendiges %-Zeichen vergessen werden, so erscheint an entsprechender Stelle im Diagramm ein Leerzeichen. Umgekehrt weisen solche Brüche im Diagramm auf vergessene Kommentarzeichen hin.

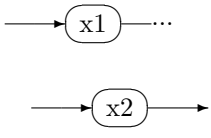
```
\begin{Syntaxdiagramm}%
  \TerSymR{xx}
  \TerSymR{xx}
\end{Syntaxdiagramm}
```



Sollte ein Syntaxdiagramm zu lang für eine einzelne Zeile werden, so kann ein Teildiagramm zu einer Syntax-Variable zusammengezogen werden und diese in einem weiteren Diagramm aufgeschlüsselt werden.

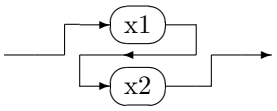
Ist dies nicht erwünscht, so kann die `Syntaxdiagramm`-Umgebung mit einem optionalen Parameter aufgerufen werden. Der Inhalt des Parameters ersetzt dann als Ende–Symbol den abschließenden Pfeil. Verwendet man den Befehl `\Dots` so kann man das Diagramm mit einem weiteren in der nächsten Zeile fortsetzen. Dieses kann durch eine voranstehendes `\quad` etwas eingerückt werden.

```
\begin{Syntaxdiagramm}[\Dots]%
  \TerSymR{x1}
\end{Syntaxdiagramm}
\quad
\begin{Syntaxdiagramm}%
  \TerSymR{x2}
\end{Syntaxdiagramm}
```



Schließlich gibt es den Stapel Befehl der zwei Teile eines Diagramms untereinander setzt und sie durch einen rücklaufenden Pfeil verknüpft. Oft ist das Aufteilen ohne rücklaufenden Pfeil allerdings sowohl einfacher als auch übersichtlicher.

```
\begin{Syntaxdiagramm}%
  \Stapel{\TerSym{x1}}%
         {\TerSym{x2}}%
\end{Syntaxdiagramm}
```



6 Weitere Hinweise

- Einige Konstruktionen erzeugen nur dann Verbindungslinien korrekter Länge, wenn auf den Pfaden Einträge mit der richtigen Höhe bzw. Tiefe auftreten. Im ersten Beispiel zur Umgebung `Alternativen` ist dies nicht der Fall.
- In der Datei syntax.tex sind einige Anmerkungen und eine kleine Liste möglicher Erweiterungen. Wer Zeit und Lust hat darf sich gerne austoben und das eine oder andere erledigen.
- Kritik, Anregungen und eigene Beiträge sind in jedem Fall erwünscht. Vielleicht finde ich bei ausreichendem Interesse sogar die Zeit noch ein wenig an diesem Paket zu basteln.

7 Kontakt


```
-------------------------------------
            Bernd Worsch
         Doberanerstrasse 99
           18057 Rostock
  bernd.worsch@stud.uni-rostock.de
          email preferred
-------------------------biichi-ji---
```