

# **LaTeX-Demo Beispiele**

Stefanie Schmidt

# **Inhaltsverzeichnis**

<b>1 Fahrplan</b>	<b>6</b>
<b>2 Abbildungs- und Tabellenverzeichnis erstellen</b>	<b>7</b>
<b>3 Abbildungen</b>	<b>7</b>
3.1 Bitmap-Grafiken platzieren . . . . .	7
3.2 Diagramme und Vektorgrafiken platzieren . . . . .	8
3.3 Grafiken in LaTeX erstellen mit TikZ . . . . .	11
<b>4 Tabellen</b>	<b>13</b>
<b>5 Code-Listings</b>	<b>14</b>

## **Abbildungsverzeichnis**

1	Unreife Bananen aus Thailand – Originalgröße . . . . .	7
2	Thai-Bananen verkleinert und linksbündig . . . . .	8
3	Thai-Bananen verkleinert und rechtsbündig . . . . .	8
4	OAuth2-PKCE-Flow-Diagramm . . . . .	9
5	OAuth2-PKCE-Flow-Diagramm-beschnitten . . . . .	10

## **Tabellenverzeichnis**

1	Budgetplan für 2026	13
---	---------------------	----

## List of Listings

1 Fläche und Umfang eines Rechtecks in JavaScript berechnen . . . . . 15

# 1 Fahrplan

Diese Demonstration bildet grundlegende Funktionen und Layout-Beispiele ab, wie Code-Listings, Tabellen, Listen und grafische Darstellungen. Außerdem soll gezeigt werden, wie man ein Dokument erstellt, das aus mehreren Dateien besteht, eigene Umgebungen festlegt und Verzeichnisse erstellen kann.

Gleich als Erstes wird diese eher unsortierte und schwer lesbare Aufzählung als ungeordnete Liste gezeigt:

- Verzeichnisse, wie Inhalts- und Abbildungsverzeichnis erstellen
- aus mehreren Dateien ein Dokument erstellen
- Bitmap-Grafiken einbinden und nach Wunsch ausrichten (links-, rechtsbündig, zentriert)
- Diagramme direkt mit LaTeX erstellen oder einbinden
- Code-Listings entsprechend einer spezifischen Programmiersprache erstellen
- Erstellen einer simplen Tabelle

## 2 Abbildungs- und Tabellenverzeichnis erstellen

In einer größeren, komplexen Arbeit, wie ein Fachbuch, hat man wahrscheinlich Tabellen und die eine oder andere Abbildung. Wenn man für diese eigene Verzeichnisse erstellen möchte, gibt es in Latex dafür passende Befehle. Diese Befehle sind `listoffigures` und `listoftables`. Damit die Befehle auch etwas zeigen, werden ein Bild und eine Tabelle gezeigt. Die beiden Befehle schreibt man in die Präambel.

## 3 Abbildungen

### 3.1 Bitmap-Grafiken platzieren

Abbildungen sind ein wesentlicher Bestandteil von Sach- und Fachbüchern, sowie von Sach- und Fachartikeln. Sie lockern lange Textpassagen auf und veranschaulichen Zusammenhänge.

Für dieses Beispiel wird eine Bitmap-Grafik eingebunden und verschieden platziert. Als erstes soll das Bild in Originalgröße zentriert eingebunden werden.

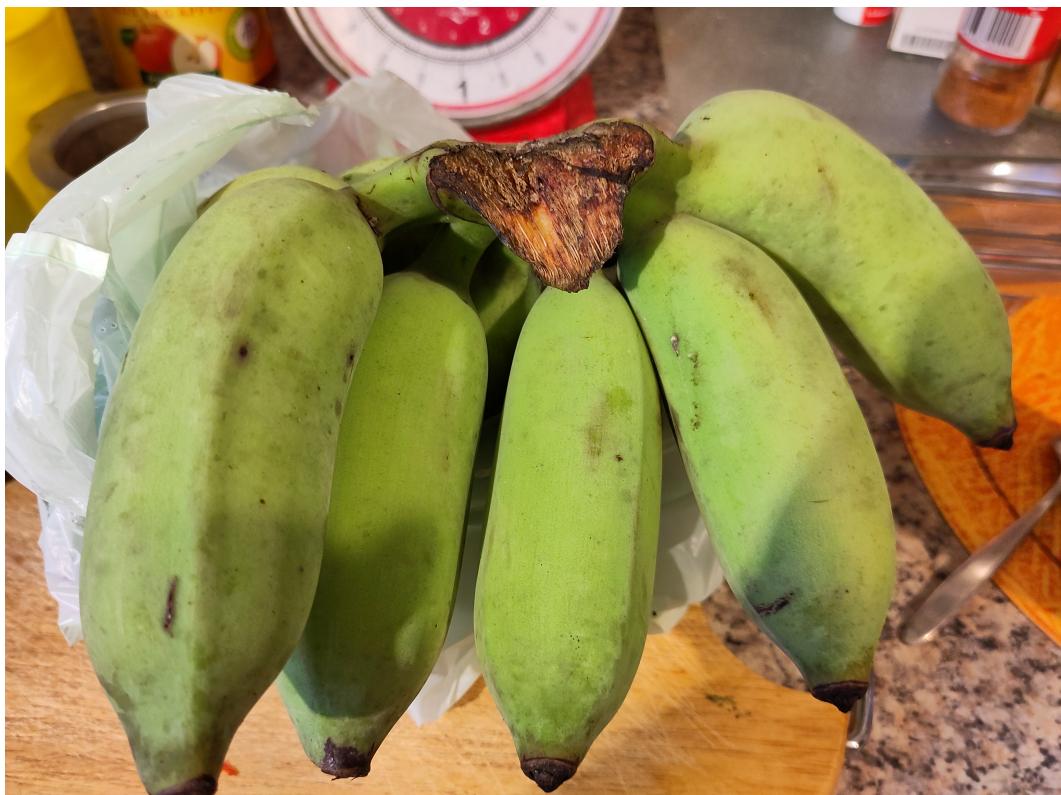


Abbildung 1: Unreife Bananen aus Thailand – Originalgröße

Mit der Angabe der Breite gibt man an, wieviel Prozent des Papiers das Bild in Anspruch nehmen soll, abzüglich der Seitenränder. 1.0 entspricht hierbei 100 %, also die gesamte Breite. In diesem Beispiel wurde eine Breite von 0.95 gewählt, damit das Bild an beiden Rändern ein wenig eingerückt wird.

Im nächsten Beispiel soll das Bild auf der linken Seite eingebunden werden. Hierzu muss man die figure-Umgebung entsprechend anpassen.



Abbildung 2: Thai-Bananen verkleinert und linksbündig

Wenn man Grafiken rechtsbündig platzieren möchte, muss der Bereich links vom Bild ausgefüllt werden. Dies erreicht man mit dem Befehl *hfill* (steht für horizontal fill).



Abbildung 3: Thai-Bananen verkleinert und rechtsbündig

### 3.2 Diagramme und Vektorgrafiken platzieren

In diesem Abschnitt geht es um Diagramme und Vektorgrafiken, denn was ist eine gute Facharbeit ohne Diagramme? Um es simpel zu halten, kann man ein Diagramm, das man mit einer speziellen Software erstellt hat, z.B. Mermaid, in ein PDF umwandeln, da LaTeX von Haus aus keine SVGs unterstützt. Das PDF kann dann wie eine Bitmap-Grafik eingebunden werden, indem man das *graphicx*-Paket verwendet:

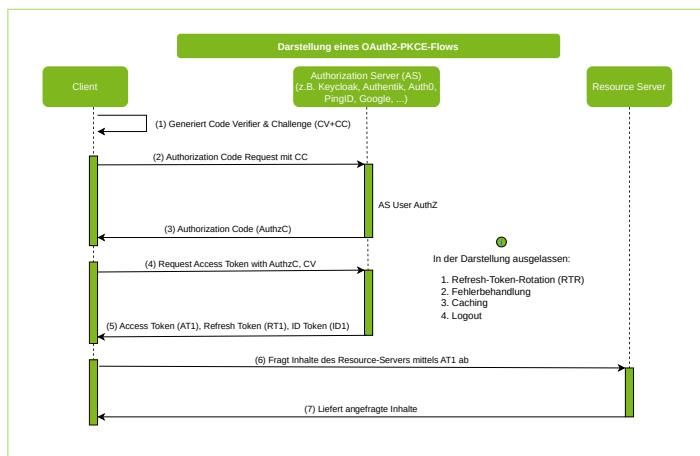


Abbildung 4: OAAUTH2-PKCE-Flow-Diagramm

Nun, das sieht nicht richtig aus. Das Diagramm nimmt eine ganze Seite in Anspruch, obwohl es deutlich kleiner ist und es ist auch nicht zentriert. Was ist passiert?

Beim Export eines Diagramms kann es vorkommen, dass man nicht aufpasst und die Standard-Einstellungen verwendet. In dem Fall wird das Diagramm auf eine DIN-A4-Seite platziert, auch wenn es kleiner ist. LaTeX bindet nun das PDF so ein, wie es ist, mit allem Freiraum. Das heißt, das PDF selbst ist zentriert, das Diagramm ist jedoch immer noch in der linken oberen Ecke des PDFs. Daher ist das Diagramm nicht zentriert.

Bei Drawio kann man beim Export die Option *Zuschneiden* wählen, was dann nur das Diagramm als PDF exportiert, ohne eine ganze Seite zu beanspruchen.

Das überarbeitete Diagramm sieht nun folgendermaßen aus:

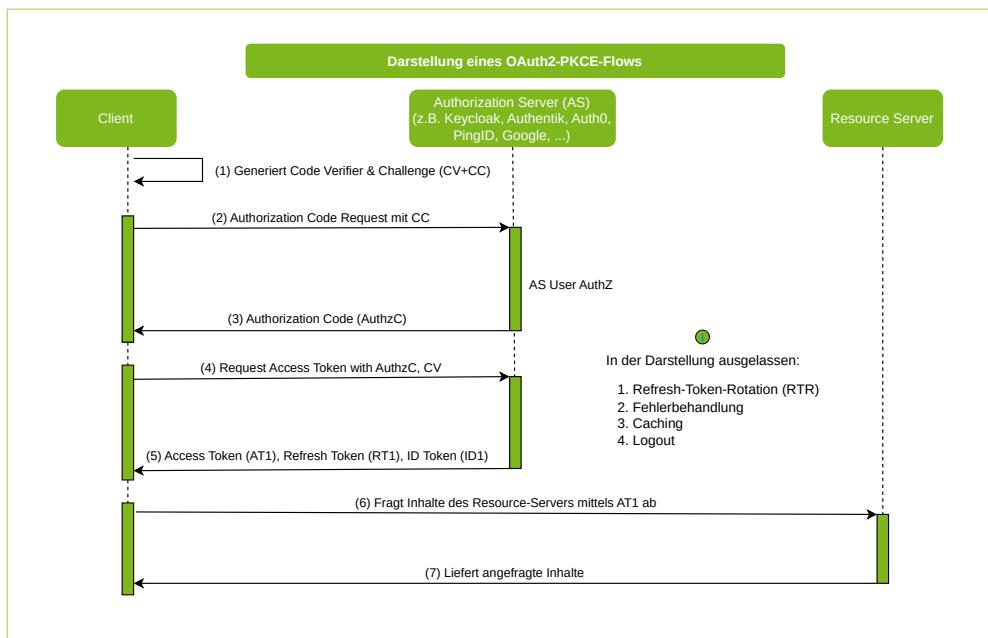


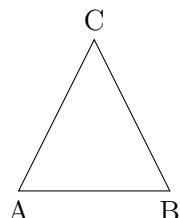
Abbildung 5: OAUTH2-PKCE-Flow-Diagramm-beschnitten

Das Ergebnis sieht deutlich besser aus. Das Diagramm nimmt keine ganze Seite mehr ein und ist korrekt zentriert.

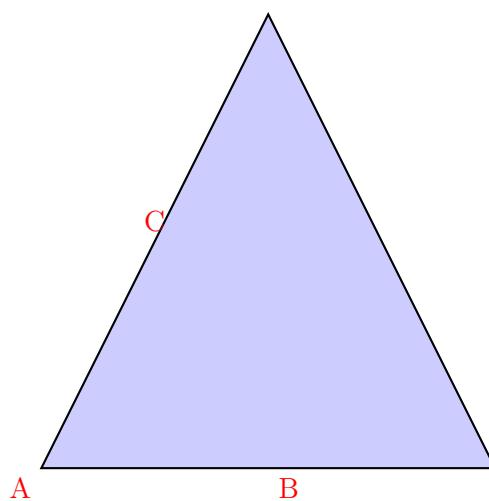
### 3.3 Grafiken in LaTeX erstellen mit TikZ

Um Grafiken direkt in LaTeX zu erstellen, benötigt man spezielle Pakete, nämlich TikZ oder PSTricks, wobei TikZ das weiter verbreitete sein dürfte. Hat man das volle LaTeX-Paket installiert, sollte TikZ bereits enthalten sein, so dass kein extra Schritt notwendig ist.

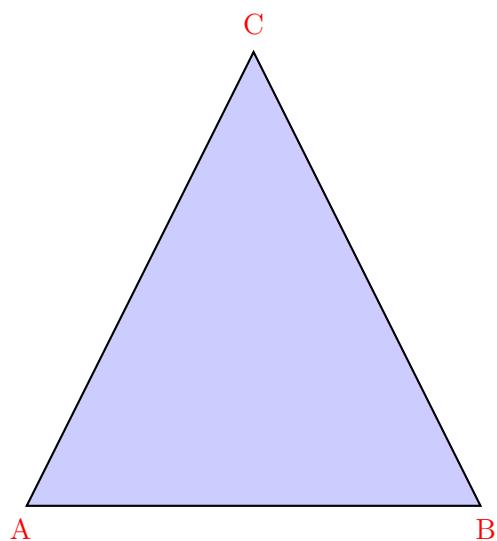
Wie gewohnt bindet man das Paket in der Präambel mit *usepackage* ein. Ein einfaches Dreieck kann man wie folgt zeichnen:



Das ist super, aber ein bisschen zu simpel. Wie wäre es, das Dreieck in die Mitte zu schieben, zu vergrößern und Farbe zu geben?



Das sieht schon besser aus, ist aber noch nicht rund – die Labels sind nicht so platziert, dass es ein stimmiges Bild gibt. Hier sind also noch einige Anpassungen nötig.



Betrachtet man die Ergebnisse und welcher Aufwand nötig ist, sieht man schon, dass man bei komplexeren Diagrammen und Grafiken schnell ein recht komplizierten und umfangreichen Quelltext bekommt. Wer experimentieren möchte, kann noch 3D-Effekte hinzufügen oder dem Dreieck einen Schatten geben. Das ist aber eindeutig über dem hinaus, was der Artikel zeigen kann.

## 4 Tabellen

In diesem Abschnitt dreht sich alles um Tabellen und wie man sie mit LaTeX ansprechend und informativ gestalten kann.

Angenommen man möchte als Teamleiter seine Budgetplanung einreichen (macht man wahrscheinlich nicht mit LateX, aber vielleicht ist die Budgetplanung Teil von etwas Größerem), dann möchte man eine Überschrift, mindestens drei Spalten und mehrere Zeilen haben. Die Spalten sollen sprechende Bezeichnungen bekommen, damit man als Leser weiß, welcher Inhalt zu erwarten ist.

Tabelle 1: Budgetplan für 2026

Datum	Art	Betrag
03. Jan.	KI-Maschine	10.000 €
15. März	2 neue Arbeitsplätze	20.000 €
30. Sept.	Erneuerung Software-Lizenzen	55.000 €
21. Dez.	Weihnachtsfeier	5.000 €
<b>Summe:</b>		<b>90.000 €</b>

Dies ist ein sehr vereinfachtes Beispiel, wie man mit LaTeX Tabellen erstellen kann. Z.B. können die Spaltenüberschriften mehrzeilig sein, oder es können Zellen und Spalten mit einander verschmolzen werden. Das kennt man vielleicht aus Tabellenkalkulationsprogrammen. Im Gegensatz zu diesen muss man das allerdings mit den richtigen Kommandos von Hand vornehmen.

## 5 Code-Listings

LaTeX unterstützt selbstverständlich auch die Darstellung von Code-Listings. Dafür bietet LaTeX mehrere Möglichkeiten, und zwar die Pakete Verbatim, Listings und Minted. Minted ist ein Zusatz-Paket und dürfte bei texlive-core nicht enthalten sein. Hat man hingegen texlive-full installiert, sollte es kein Problem sein. Warum nun drei Pakete zur Auswahl?

Alle drei Pakete haben ihre Berechtigungen, wobei Minted das mächtigste Werkzeug ist. Verbatim stellt Text so dar, wie er eingegeben wurde und kann hilfreich sein, wenn man beispielsweise Text mit vielen Sonderzeichen hat, die man sonst alle mit einem Backslash escapen müsste. Für Code-Listings kann man es zwar benutzen, hat aber keine Möglichkeiten für Syntax-Hervorhebung oder das Anzeigen von Zeilennummern.

Das Listings-Paket kann da schon mehr. Es unterstützt allerdings nur einige Programmiersprachen, z.B. ist JavaScript nicht dabei. Wer Unterstützung für JavaScript benötigt, braucht das Minted-Paket.

Bevor man jedoch das Paket installiert, sollte man sich versichern, dass man die aktuelle Version Python (2.6 oder neuer) und Pygments installiert hat. Beides wird von Minted direkt gebraucht.

Hier ist ein kurzes Beispiel für einen Verbatim-Text.

```
\begin{verbatim}
Hier steht alles so drin, wie es eingegeben wurde, auch die nicht druck-
baren Zeichen.
Auch Sonderzeichen, wie \ werden ohne Probleme dargestellt. Es sieht
aber nicht unbedingt schön aus.
\verb[end{verbatim}]
```

Wie bereits auffällt, muss man sich um die Zeilenumbrüche selbst kümmern. Verbatim "überschreibt" quasi alles, was LaTeX normalerweise automatisch macht. Auch die Einrückung muss man selbst machen. Bei langen Listings wird das schnell lästig.

Deshalb ist hier nun ein kurzes Beispiel JavaScript-Beispiel mit dem Minted-Paket.

---

```
1 // Define a Rectangle object
2 function Rectangle(length, width) {
3     this.length = length; // Integer type
4     this.width = width; // Integer type
5
6     // Method to calculate the area
7     this.calculateArea = function() {
8         return this.length * this.width;
9     };
10
11    // Method to calculate the circumference
12    this.calculateCircumference = function() {
13        return 2 * (this.length + this.width);
14    };
15 }
16
17 // Instantiate an object called myRect
18 const myRect = new Rectangle(10, 5); // Example: length = 10, width = 5
19
20 // Call the calculation functions
21 const area = myRect.calculateArea();
22 const circumference = myRect.calculateCircumference();
23
24 // Log the results
25 console.log("Area: " + area); // Area: 50
26 console.log("Circumference: " + circumference); // Circumference: 30
```

---

Listing 1: Fläche und Umfang eines Rechtecks in JavaScript berechnen