

Программирование в ЛИНГВИСТИКЕ

LaTeX. Система верстки pdf

Для чего нужен LaTeX?

В LaTeX обычно верстаются самые разные **.pdf**: рабочие отчеты, презентации, научные статьи, дипломы и диссертации. Если освоиться в LaTeX, то покажется, что верстать в нем может быть **удобнее**, чем в Microsoft Word: например, Латех поддерживает специальный язык markdown для **математических формул**, а еще он умеет автоматически за вас собирать библиографию (и составлять правильные библиографические ссылки), нумеровать картинки и таблицы и много чего другого.

Entities	bg	cs	pl	ru	sl	uk	be
PER	321	377	414	373	360	298	479
LOC	974	530	716	552	690	948	821
ORG	87	195	195	292	104	138	125

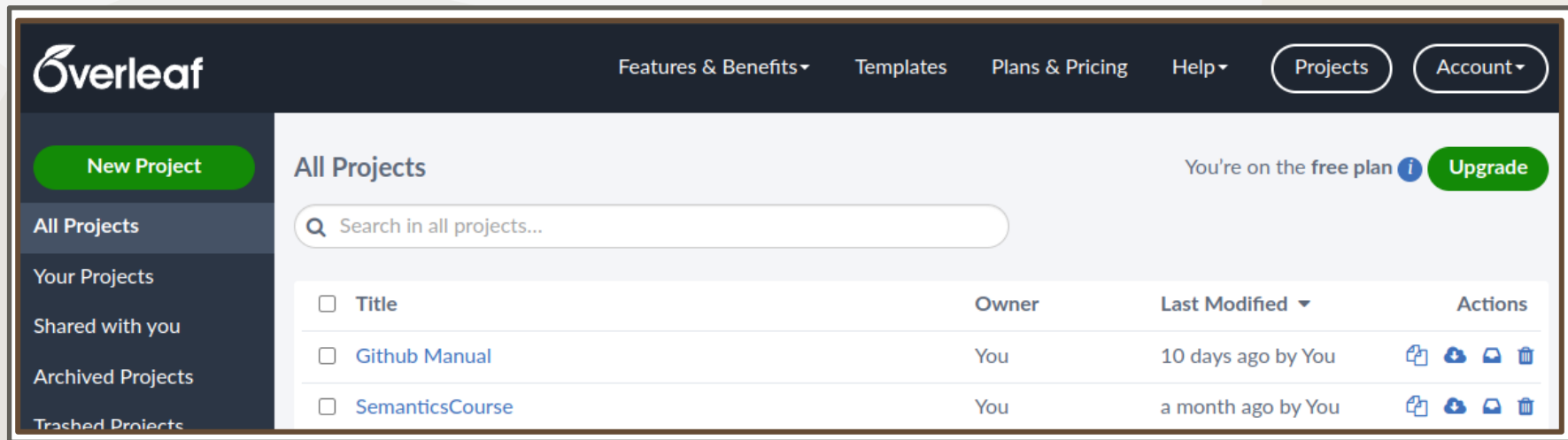
Table 2.1: NER categories in the WikiNER test dataset, number of entities

In our first experiments we also used **WikiMatrix** [Schwenk et al., 2019].

Finally, we created a **synthetic dataset** for all of our languages. The average algorithm of creation is as follows:

Как работать с LaTeX?

Латех - это, по существу, такой же язык (недоязык) программирования, как html или язык верстки Wikipedia. Удобнее всего пользоваться Латехом онлайн через облачный ресурс **Overleaf**. На этом ресурсе нужно зарегистрироваться (достаточно подключить свой аккаунт Google) и можно бесплатно верстать. Там есть платная версия, но и в бесплатной присутствуют все необходимые инструменты.



The screenshot displays the Overleaf web application interface. At the top, the Overleaf logo is on the left, and navigation links for 'Features & Benefits', 'Templates', 'Plans & Pricing', 'Help', 'Projects', and 'Account' are on the right. The 'Projects' tab is active. On the left sidebar, 'New Project' is highlighted in green, followed by 'All Projects', 'Your Projects', 'Shared with you', 'Archived Projects', and 'Trashed Projects'. The main content area is titled 'All Projects' and includes a search bar. Below the search bar is a table of projects. The table has four columns: 'Title', 'Owner', 'Last Modified', and 'Actions'. Two projects are listed: 'Github Manual' and 'SemanticsCourse', both owned by 'You'. The 'Last Modified' column shows '10 days ago by You' and 'a month ago by You' respectively. The 'Actions' column contains icons for cloning, sharing, downloading, and deleting.

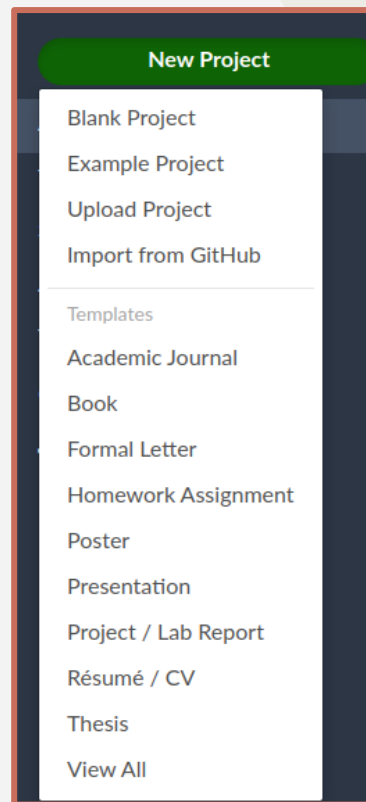
<input type="checkbox"/> Title	Owner	Last Modified ▾	Actions
<input type="checkbox"/> Github Manual	You	10 days ago by You	
<input type="checkbox"/> SemanticsCourse	You	a month ago by You	

Как создать проект LaTeX?

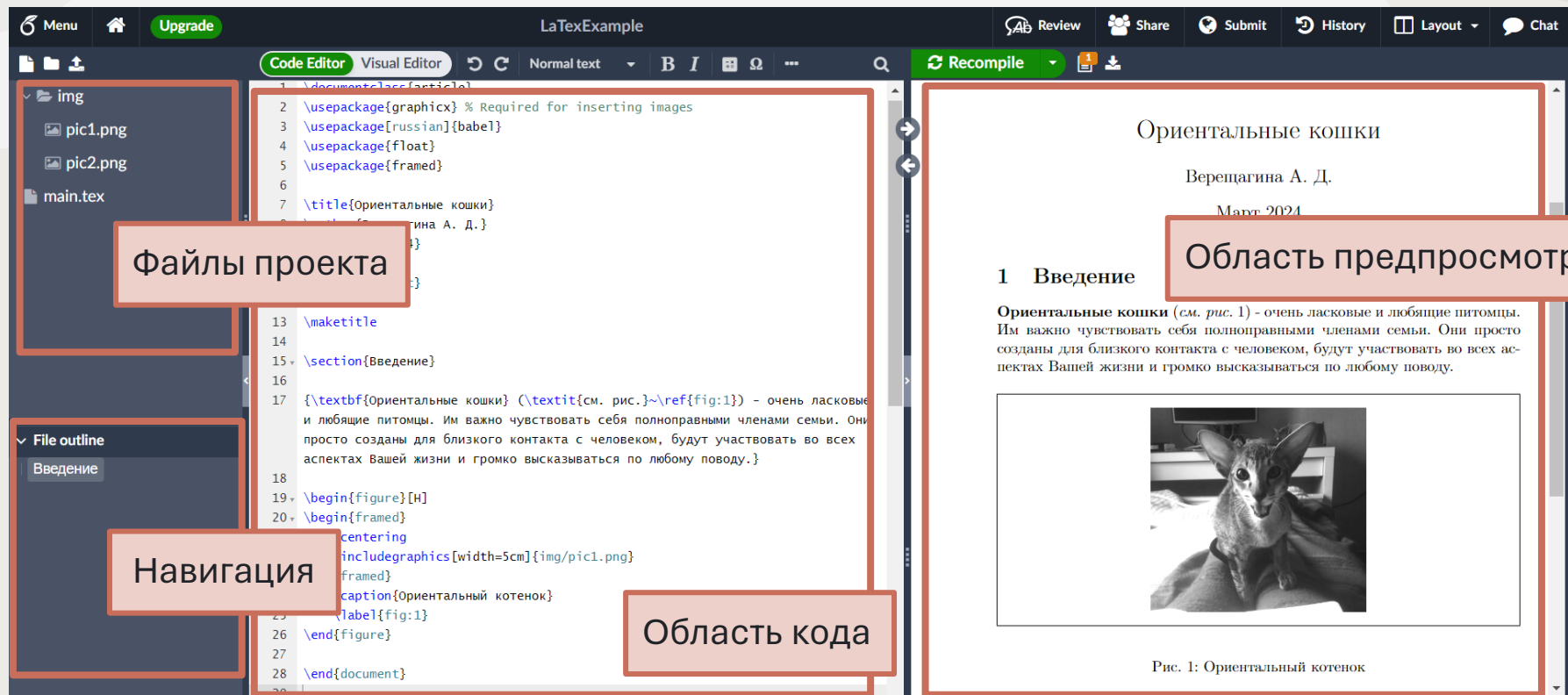
На главной странице Латех показывает все ваши проекты (и те проекты, которыми с вами кто-то поделился). Можно открыть один из существующих или создать новый.

Создать новый проект можно несколькими способами:

1. Создать **пустой новый проект**;
2. Загрузить **готовый проект в .zip** (например, если конференция, статью для которой вы хотите писать, предоставляет шаблон: обычно у них на сайте можно скачать такой .zip);
3. Использовать **шаблоны** (Templates): каждый пункт в меню - это на самом деле огромный выбор из готовых шаблонов соответствующего типа.



Интерфейс LaTeX

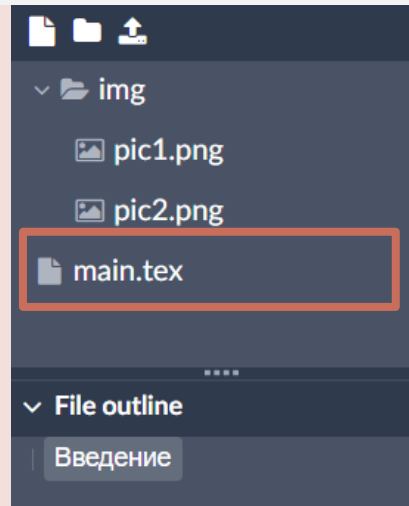


Панель файлов и навигации

В панели слева можно видеть **файлы**, относящиеся к нашему проекту (для пустого проекта это единственный файл **.tex** - тот самый, в котором нужно писать код для компиляции).

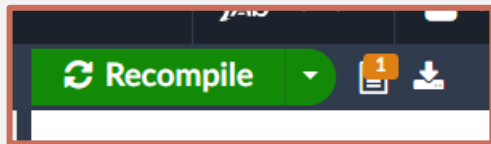
В этой панели можно заводить новые файлы и папки, загружать какие-нибудь файлы (обычно картинки).

Когда в вашем проекте появится какая-то структура, в этой же панели в 'File outline' возникнет **содержание документа**.

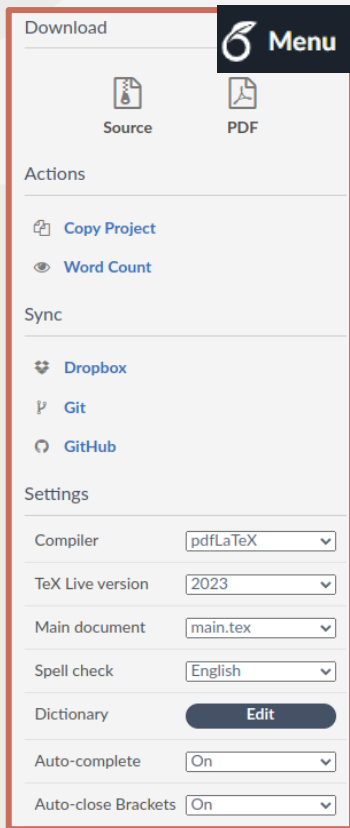


Области кода и предпросмотра

В большом окне в левой части можно собственно писать код; в правой части отображается результирующий .pdf. Имейте в виду, что Латех подгружает Вашу верстку **после компиляции (recompile)**. Также справа можно нажать на кнопку для загрузки pdf, а между кнопками 'recompile' и 'download' находится кнопка **'Logs and output files'**, в которой можно увидеть логи компиляции: если на ней появляются **красные сообщения**, это значит, что скомпилировалось с **ошибками**, если же сообщения желтые - то это просто какие-то предупреждения, которые могут быть критичными, а могут быть вообще несущественными.

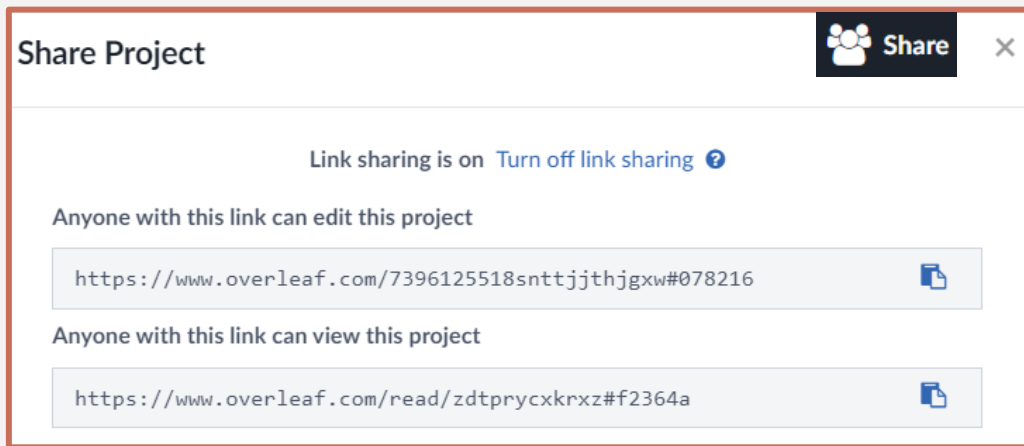


Панели 'Menu' и 'Share'



В меню тоже можно скачать либо весь ваш проект целиком в .zip, либо только pdf, посчитать, сколько в вашем проекте слов (для статей это очень важно), а также поменять какие-нибудь настройки или синхронизировать с гитхабом.

И, наконец, можно **поделиться** этим проектом с коллегами точно так же, как Вы делаете это в Google Drive:



Верстка документа

```
\documentclass{article}
\usepackage{graphicx}

\title{RandomLaTeX}
\author{Anna Vereshchagina}
\date{March 2024}

\begin{document}

\maketitle

\section{Introduction}

\end{document}
```

Если мы заводим проект с нуля, то в рабочем файле появляются эти строки.

Как можно догадаться, все строки, которые начинаются на бэкслеш - это **команды**. Обычная команда в Латехе имеет вид **\command[{}]** (количество скобок может варьироваться или вообще отсутствовать).

Самая первая команда задает **класс** документа: по умолчанию это статья. Список доступных Латеху классов можно посмотреть [тут](#). К этой команде еще можно добавить квадратные скобки, куда вписать какие-нибудь параметры вашего класса: например, дефолтный размер шрифта. Тогда команда будет выглядеть как **\documentclass[12pt]{article}**.

Полезные пакеты

Обычно в Латехе приходится подключать какие-то дополнительные пакеты, точно так же, как в Python, только если в Питоне мы пишем `import`, то в Латехе соответствующая команда выглядит как `\usepackage{name}`. Чаще всего пригождаются следующие пакеты:

<code>\usepackage[utf8]{inputenc}</code>	Кодировка utf-8
<code>\usepackage[T1, T2A]{fontenc}</code>	Подключают кодировки: без этих пакетов может не работать кириллица (если выбран компилятор по умолчанию)
<code>\usepackage[russian]{babel}</code>	Подключает языковую локаль: некоторые стандартные штуки будут зависеть от того, какой язык здесь указан
<code>\usepackage{graphicx}</code>	Для работы с картинками
<code>\usepackage[font=footnotesize]{caption}</code>	Чтобы можно было делать подписи к картинкам. Квадратные скобочки необязательные
<code>\usepackage{todonotes}</code>	Позволяет рисовать себе на полях pdf заметки
<code>\usepackage{wrapfig}</code>	Позволяет делать обтекаемые текстом картинки
<code>\usepackage{tikz}</code>	Позволяет размещать картинки (и не только) в randomном месте страницы

Полезные пакеты

Нам с вами, как лингвистам, еще могут пригодиться такие пакеты:

```
\usepackage{tipa}
```

Для фонетических символов;

```
\usepackage{qtree}
```

```
\usepackage{tikz-qtree}
```

Для рисования деревьев составляющих;

```
\usepackage{tikz-dependency}
```

```
\usetikzlibrary{%
```

```
  shapes,%
```

```
  arrows,%
```

```
  positioning,%
```

```
  calc,%
```

```
  automata%
```

```
}
```

Для рисования деревьев зависимостей;

```
\usepackage{linguex}
```

Для глоссирования.

Структура документа

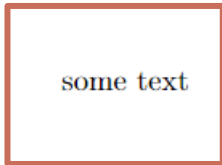
Команды `\title{title}`, `\author{name}`, `\date{date}` **необязательны** и нужны для команды **`\maketitle`**: они просто задают значения, которые потом будут отображены по этой команде.

Вот дальше идет самое важное: пара команд **`\begin{document}`** и **`\end{document}`**. Все, что находится между этими командами, и будет рендериться в pdf: все, что снаружи, обычно является какими-то настройками (импорт пакетов, определение кастомных команд, задание значений и так далее). В Латехе команды обычно вообще бывают двух видов: бывают одиночные, как импорты или задание класса документа, а бывают такие, которые требуют `begin` и `end`: создание картинки, таблицы, дерева зависимости и некоторые другие.

```
\begin{document}
```

some text

```
\end{document}
```



Документ

```
\begin{figure}[h]
```

```
\centering
```

```
\includegraphics[width=5cm]{img/pic1.png}
```

```
\end{figure}
```



Картинка

Текст в LaTeX

Получается, команды `\maketitle` и `\section{...}` - это команды, которые будут что-то рендерить в нашем pdf. На самом деле, внутри тела документа мы уже можем писать **любой текст** - он будет отображаться.

Можно настраивать вид шрифта (жирный, курсив и подчеркнутый): для жирного и курсива работают горячие клавиши **Ctrl+I** и **Ctrl+B**, а вот подчеркнутый придется вручную вписать командой `\underline{}` (все, что внутри фигурных скобок, будет в таком виде).

Для жирного шрифта аналогичная команда `\textbf{}` (на самом деле горячие клавиши просто автоматически вставляют ее), а для курсива - `\textit{}`.

В Латехе для комментирования служит символ `%` (если же он вам понадобился сам по себе, придется его заэкранировать: `\%`). Он работает точно так же, как питоновский `#`.

```
\textit{Курсив} \textbf{Полужирный} \underline{Подчеркнутый} % комментарий
```

Разделы

Итак, первая из двух команд в нашем пустом проекте - это создание **заголовка**, а вторая - создание **первого раздела** в нашем документе. Разделы все задаются командами и могут быть нескольких уровней; обычно (это зависит от класса документа) **набор разделов** по возрастанию глубины **примерно такой**:

<code>\chapter</code>
<code>\section</code>
<code>\subsection</code>
<code>\subsubsection</code>

(не будет в статье - это для книг)

```
\chapter{Котики}  
\section{Очень}  
\subsection{Очень}  
\subsubsection{Милые}
```

Глава 1

Котики

1.1 Очень

1.1.1 Очень

Милые

Можно автоматически собрать **оглавление** по разделам командой `\tableofcontents`.

Отступы

Итак, просто текст верстать мы уже можем. Если нам необходимо при этом делать какие-то отступы, то есть две команды `\vspace{}` и `\hspace{}`, которые размещают соответственно вертикальные и горизонтальные отступы. В фигурных скобках пишется размер (можно в pt, а можно в cm): `\vspace{1cm}` сделает вертикальный отступ в один сантиметр.

```
{\textbf{Ориентальные кошки} - очень ласковые и любящие питомцы. Им важно  
чувствовать себя полноправными членами семьи.
```

```
\hspace{1cm}Они просто созданы для близкого контакта с человеком, будут  
участвовать во всех аспектах Вашей жизни и громко высказываться по любому  
поводу.}
```

```
\vspace{1cm}Мяу!
```

Ориентальные кошки - очень ласковые и любящие питомцы. Им важно чувствовать себя полноправными членами семьи.

Они просто созданы для близкого контакта с человеком, будут участвовать во всех аспектах Вашей жизни и громко высказываться по любому поводу.

Мяу!

Картинки

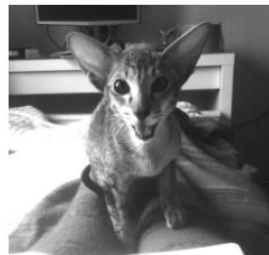
В документе можно, конечно, размещать и картинки любыми способами.

Самый базовый пакет для работы с картинками - это **graphicx**. Для него можно создать в проекте папку с каким-нибудь названием типа **img**, куда складывать все картинки, которые нужно будет отрендерить. Когда вы подключили этот пакет, можно размещать картинки!

Базовая команда для картинок: `\includegraphics{img/mypic.png}`. У нее дополнительно могут быть параметры в квадратных скобках, обычно они касаются размеров картинки:

`\includegraphics[width=13cm]{img/mypic.png}`.

```
\includegraphics[width=5cm]{img/pic.png}
```



Картинки с подписями

Но, скорее всего, мы захотим не просто разместить картинку, но еще и оформить к ней **подпись (Caption)**. Для этого уже понадобится создавать объект типа **Figure**.

Обычно Латех сам подсказывает вам, и если **нажать на Enter** в момент, когда он подсказывает продолжить за вас figure, он **автоматически вставит** этот код и поставит ваш курсор на выборе пути к картинке, а также предложит выбрать путь.

```
\begin{figure}[h]
  \centering % отцентрирует картинку, необязательный
  \includegraphics[width=4cm]{img/pic.png} % найдет нужную картинку
  \caption{Ориентальный котенок}
  \label{fig:cat} % fig показывает латеху, что это картинка, а текст,
  который идет дальше, нужен, чтобы ссылаться на нее в тексте
\end{figure}
```



Рис. 1: Ориентальный котенок

Ссылки на картинки

Теперь на нашу картинку можно сослаться в тексте как `\ref{fig:enter-label}`: тогда в этом месте Латех проставит автоматически номер картинки, и надо будет написать перед ним, например, `см. картинку~\ref{fig:enter-label}` (~ делает неразрывный пробел). Можно сослаться и как `\autoref{fig:enter-label}`: это автоматически разместит и слово Figure, но только если вы подключили пакет `hyperref`.

```
{\textbf{Ориентальные кошки}} (\textit{см. рис.}~\ref{fig:cat}) -  
очень ласковые и любящие питомцы.
```

кошки (*см. рис. 1*) - очень ласковые



Рис. 1: Ориентальный котенок

Картинка на месте

В коде с картинкой был использован такой элемент – `[h]`.

Проблема в том, Латех размещает элементы в pdf так, как **ему кажется** максимально удобно, и поэтому может положить картинку в неожиданное место, например, **в самый конец документа**, если ему показалось, что она по размеру не вписывается.

Можно это поведение изменить, если добавить квадратные скобочки: `\begin{figure}[h]`. Маленькая буква **h** в этих скобочках говорит Латеху "будь добр, размести вот **here**, если можешь". Но Латех и тогда может все равно ее выкинуть в другое место: это **вежливая** просьба. Можно просьбу сделать понастойчивее, если написать **H**. А если и тогда не помогает, можно подключить пакет `\usepackage{float}` и сказать **!h** - это означает "порвись, но размести ее там, где она в коде!". Тогда может образоваться пустой конец страницы, но Латех уже точно вас послушает.

Картинка в тексте

Если Вы хотите, чтобы ваша картинка была **обвернута текстом** (как в Word), нужно использовать пакет **wrapfig**. Тогда код для вставки картинки будет выглядеть так:

```
\begin{wrapfigure}[r]{0.5\textwidth}
% r означает, что размещаем по правой стороне листа.
% 0.5\textwidth означает сделать ширину объекта
% в половину ширины текста
\centering
\includegraphics[width=0.48\textwidth]{img/pic.png} %
специально саму картинку делаем чуточку поменьше
\caption{Ориентальный котенок}
\label{fig:cat}
\end{wrapfigure}
```

Здесь нужно понимать, что сам объект **wrapfigure** - это белое **пространство** вокруг вашей картинки, а картинка - это только то, что вставляется командой `\includegraphics`.

Ориентальные кошки (см. рис. 1) - очень ласковые и любящие питомцы. Им важно чувствовать себя полноправными членами семьи. Они просто созданы для близкого контакта с человеком, будут участвовать во всех аспектах Вашей жизни и громко высказываться по любому поводу.

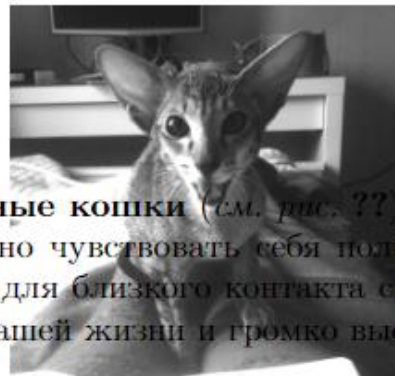


Рис. 1: Ориентальный котенок

Картинка в необычном месте

Если нужно разместить картинку где-то в необычном месте (например, если верстаете **презентацию**), можно использовать **tikz**:

```
\begin{tikzpicture}[overlay,remember picture]  
  \node[anchor=north west,inner sep=0pt]at  
    ([xshift=6.5cm,yshift=-3cm]current page.north west)  
    {\includegraphics[width=4cm]{img/pic.png}};  
\end{tikzpicture}
```



Ориентальные кошки (см. рис. ??) - очень лапмцы. Им важно чувствовать себя полноправными членами семьи. Они созданы для близкого контакта с человеком, в любых аспектах Вашей жизни и громко высказываться

Списки и перечисления

Списки и перечисления делаются с помощью `itemize` и `enumerate`:

Признаки ориентальных кошек:

```
\begin{itemize}
  \item[*] Большие уши
  \item[*] Короткая шерсть
\end{itemize}
```

Чего они больше всего хотят:

```
\begin{enumerate}
  \item Вашего внимания
  \item Достаточно еды и воды
  \item Чистый лоток
\end{enumerate}
```

Признаки ориентальных кошек:

- * Большие уши
- * Короткая шерсть

Чего они больше всего хотят:

1. Вашего внимания
2. Достаточно еды и воды
3. Чистый лоток

Таблицы

Верстка таблиц – непростое занятие. Но возможности LaTeX позволяют делать очень продвинутые таблицы.

```
\begin{table}[h]
\centering
\begin{tabular}{|c|c|} % это сама таблица с заданием ее
внешнего вида
\hline
Мяукает & Общается с Вами \\
\hline
Тянет лапы & Хочет на ручки \\
\hline
Трогает лапой лицо & Выражает особую любовь \\
\hline
\end{tabular}
\caption{Действия ориентального кота}
\label{tab:actions}
\end{table}
```

Ориентальный кот очень общителен
(см. таб. 1).

Мяукает	Общается с Вами
Тянет лапы	Хочет на ручки
Трогает лапой лицо	Выражает особую любовь

Таблица 1: Действия ориентального кота

Таблицы

Внешний вид таблицы (как будет выравниваться содержимое ячеек, сколько их будет, будут ли они разделены просто отступами или видимыми границами) задается в **{tabular}**.

Например, **{c|c}** сделает табличку из двух ячеек, содержимое каждой будет выровнено по центру, а разделять их будет видимая черта. При этом горизонтальных линий автоматически не появится: их нужно вручную расставлять командой **\hline**.

```
\begin{table}[]
  \centering
  \begin{tabular}{|l|c r|}
    \hline
    & center & right \\
    \hline
    left cell & center cell & right cell \\
    \hline
  \end{tabular}
  \caption{Caption}
  \label{tab:my_label}
\end{table}
```

	center	right
left cell	center cell	right cell

Вот, например, как можно задать табличку с выравниванием первой ячейки по левому краю, второй - по центру, а третьей - по правому, причем первую будет отделять видимая линия, а вторую от третьей нет.

Выравнивание

Выравнивать содержимое таблиц можно не только горизонтально, но и вертикально. Доступны такие варианты выравнивания:

l	Left (слева)
c	Center (по центру)
r	Right (справа)
p{width}	(сверху)
m{width}	Middle (посередине)
b{width}	Bottom (снизу)

Чтобы выровнять текст в ячейках **вертикально**, нужно передавать в фигурные скобки параметр width (можно в см). К примеру, `{|p{1cm}|p{1cm}|p{1cm}|}`

Двойную черту можно сделать между ячейками как `||`. А если хочется горизонтальную двойную, можно просто повторить команду `\hline`.

Tabularx

Если нам нужно контролировать ширину таблицы, можно использовать вместо tabular tabularx:

```
\begin{table}[htbp]
  \begin{tabularx}{0.6\textwidth} {
    | >{\raggedright\arraybackslash}X
    | >{\centering\arraybackslash}X
    | >{\raggedleft\arraybackslash}X | } % тоже выравнивание
    по ячейкам
  \hline
  item 11 & item 12 & item 13 \\
  \hline
  item 21 & item 22 & item 23 \\
  \hline
\end{tabularx}

\end{table}
```

item 11	item 12	item 13
item 21	item 22	item 23

...Где 0.6 регулирует ширину нашей таблицы.

Объединение ячеек

Также можно **объединять** ячейки или сливать их, это, правда, уже немножко сложновато:

```
\begin{tabular}{|p{3cm}|p{3cm}|p{3cm}|p{3cm}|} % всего у нас 4 колонки
\hline
\multicolumn{4}{|c|}{Country List} \\ % сделает в самом верху
таблички единую ячейку вместо четырех
\hline
Country Name or Area Name& ISO ALPHA 2 Code &ISO ALPHA 3 Code&ISO
numeric Code\\ % здесь уже четыре
\hline
Afghanistan & AF & AFG& 004\\
Aland Islands& AX & ALA & 248\\
Albania & AL & ALB& 008\\
Algeria & DZ & DZA& 012\\
American Samoa& AS & ASM& 016\\
Andorra& AD & AND & 020\\
Angola& AO & AGO& 024\\
\hline
\end{tabular}
```

Для всего этого необходимо сделать
`\usepackage{multirow}`.

Country List			
Country Name or Area Name	ISO ALPHA 2 Code	ISO ALPHA 3 Code	ISO numeric Code
Afghanistan	AF	AFG	004
Aland Islands	AX	ALA	248
Albania	AL	ALB	008
Algeria	DZ	DZA	012
American Samoa	AS	ASM	016
Andorra	AD	AND	020
Angola	AO	AGO	024

Библиография

Латех работает с так называемыми **bibtex-ссылками**: их можно автоматически получать в scholar.google.com, например. Эти ссылки - это просто записанные в коде данные о цитируемой книге/статье. Обычно все эти ссылки собираются в отдельный файл с расширением **.bib**, который тоже кладется **в папочку проекта** рядом с .tex.

[HTML] Progress in machin

H Wang, H Wu, Z He, L Huang, K

... the emergence of neural machi

machine translation to example-t

☆ Сохранить Цитировать

1. Находим
интересующую статью в
Scholar

ГОСТ Wang H. et al. Progress in r
2022. – T. 18. – С. 143-153.

MLA Wang, Haifeng, et al. "Progr
Engineering 18 (2022): 143-

APA Wang, H., Wu, H., He, Z., H
Progress in machine transla

BibTeX EndNote

2. Запрашиваем BibTeX

```
@article{wang2022progress,  
  title={Progress in machine translation},  
  author={Wang, Haifeng and Wu, Hua and He,  
  journal={Engineering},  
  volume={18},  
  pages={143--153},  
  year={2022},  
  publisher={Elsevier}  
}
```

3. Сохраняем код в файл .bib

Библиография

Когда вы создали такой .bib-файл и накопили в него нужных вам ссылок, можно их цитировать в тексте pdf с помощью `\cite{}`. Латех сам будет подсказывать по ярлыкам те ссылки, которые у него есть.

Для всего этого нужно **создать список литературы** в любом месте документа.

```
У кошек поведение разнится от породы  
\cite{salonen2019breed}. Ориенталы привлекают особенностями  
своего поведения, но, к сожалению, часто болеют  
\cite{esders2023single}.  
  
\bibliographystyle{apalike}  
\bibliography{refs.bib}
```

У кошек поведение разнится от породы [Salonen et al., 2019]. Ориенталы привлекают особенностями своего поведения, но, к сожалению, часто болеют [Esders et al., 2023].

Литература

[Esders et al., 2023] Esders, S. L., Hülskötter, K., Schreiner, T., Wohlsein, P., Schmitz, J., Bräsen, J. H., and Distl, O. (2023). Single nucleotide polymorphisms associated with aa-amyloidosis in siamese and oriental shorthair cats. *Genes*, 14(12):2126.

[Salonen et al., 2019] Salonen, M., Vapalahti, K., Tiira, K., Mäki-Tanila, A., and Lohi, H. (2019). Breed differences of heritable behaviour traits in cats. *Scientific reports*, 9(1):7949.

Фонетические обозначения

Пакет **tipa** нужен просто для того, чтобы можно было вставлять любые **фонетические обозначения**: полный список команд для таблицы IPA можно посмотреть [тут](#). Соответственно, с ним никаких сложностей, скорее всего, не будет.

IPA L^AT_EX Codes Within `\textipa{...}`¹

Consonants

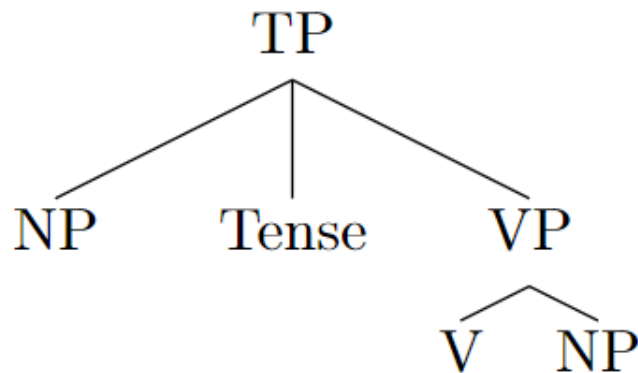
	<i>Bilabial</i>	<i>Labiodental</i>	<i>Dental</i>	<i>Alveolar</i>	<i>Postalveolar</i>	<i>Retroflex</i>	<i>Palatal</i>	
<i>Plosive</i>	p b			t d		\:t \:d	ɕ ɟ	
<i>Nasal</i>	m	ɱ		n		\:n	\textbardotlessj	
<i>Trill</i>	\;B	ʙ		r			\textltailn	
<i>Tap or Flap</i>				R	ɾ	\:r	ɽ	
<i>Fricative</i>	F Φ B β	f v	T θ D ð	s S z Z	ʃ ʒ	\:s \:z	\c{c} J	ç j
<i>Lateral Fricative</i>				\textbeltl \textlyoghlig	ɬ ɮ			
<i>Approximant</i>		V U		*r	ɹ	\:R \:r	j	\texttt
<i>Lateral Approximant</i>				l		\:l	ɭ ʟ	

Деревья составляющих

Деревья составляющих можно довольно легко рисовать с помощью пакета **qtreet**. Как только вы его подключите, можно пользоваться командой **\Tree**. Например:

```
\Tree [.TP NP Tense [.VP V NP ] ]
```

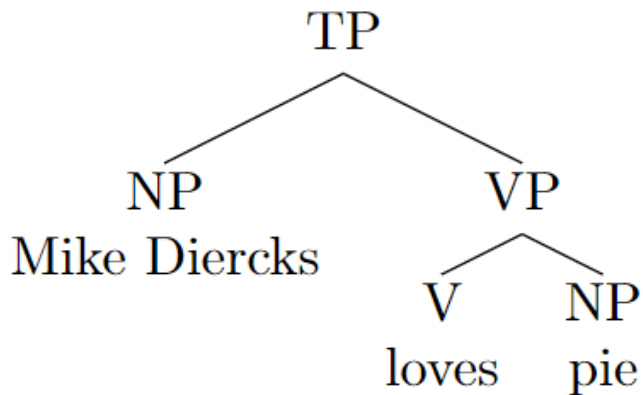
После самой команды нужно описать желаемое дерево.
Квадратные скобки обозначают **границы составляющих**. Точка впереди ставится у нетерминала, то есть, у узла дерева (в отличие от листа): в примере выше стрелки будут рисоваться от TP и VP с точкой.



Деревья составляющих

Если Вам нужно нарисовать дерево **со словами**, то можно их добавить с помощью **двойных бэкслешей**. А если нужно отобразить **более одного** слова, используются **фигурные скобки**:

```
\Tree [ .TP NP\\{Mike Diercks} [ .VP V\\loves NP\\pie ] ]
```

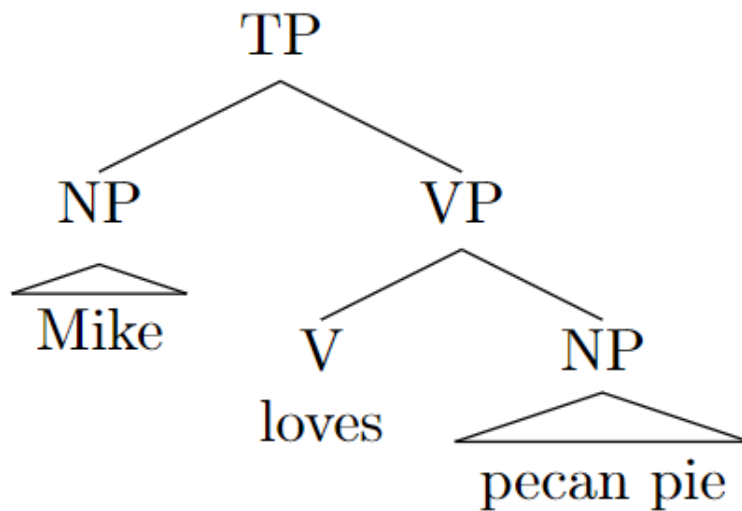


Деревья составляющих

Для того, чтобы нарисовать **треугольник**, нужно использовать команду `\qroof`.

Эти штуки могут быть только **терминалами**.

```
\Tree [.TP \qroof{Mike}.NP [.VP V\\loves \qroof{pecan pie}.NP ] ]
```

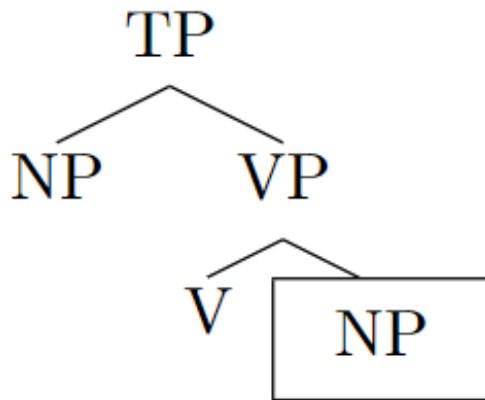


Деревья составляющих

Если хочется обвести какую-то часть дерева **в квадрат**, можно использовать **такую команду**.

Она будет применяться к тому, что стоит непосредственно от нее **слева** (то есть, если слева закрывающая квадратная скобка (]), в квадрат попадет все, что внутри нее).

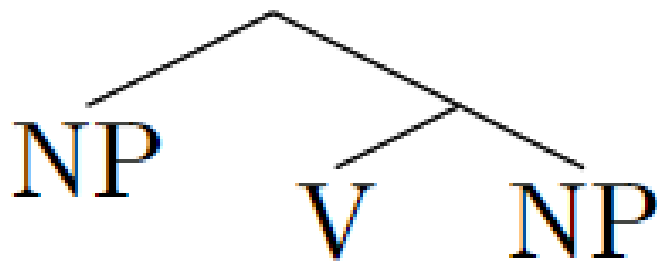
```
\Tree [ .TP NP [ .VP V NP !{\qframesubtree} ] ]
```



Деревья составляющих

Наконец, если вам не нужны нетерминалы, можно просто их не писать, тогда в дереве их не будет:

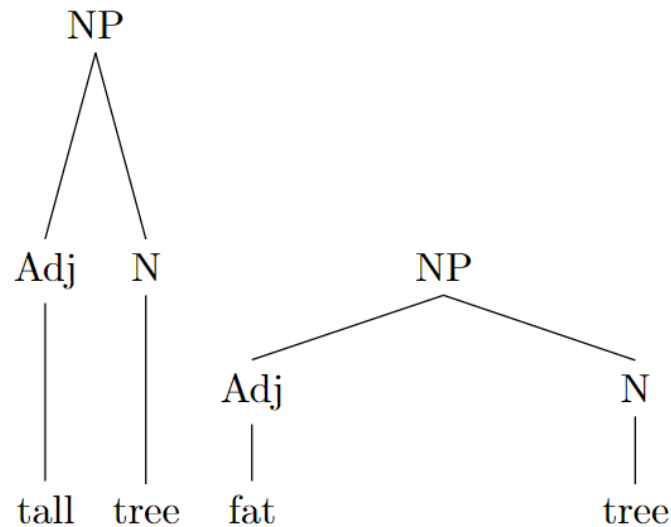
```
\Tree [ NP [ V NP ] ]
```



Деревья составляющих

Также есть пакет `\usepackage{tikz-qtree}`, который позволяет тоже рисовать деревья составляющих. Для его использования нужно подключать и `tikz` тоже. У него расширенные по сравнению с `qtree` возможности. Базовый синтаксис у них совпадает.

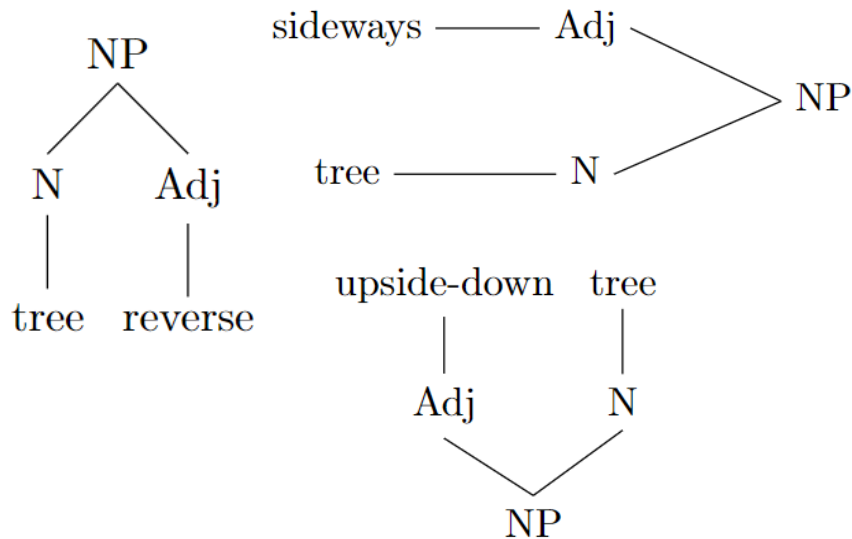
```
\begin{tikzpicture}
\tikzset{level distance=60pt}
\Tree [.NP [.Adj tall ] [.N tree ] ]
\end{tikzpicture}
%
\begin{tikzpicture}[sibling distance=72pt]
\Tree [.NP [.Adj fat ] [.N tree ] ]
\end{tikzpicture}
```



Деревья составляющих

Деревья можно отрисовывать как специальные объекты tikzpicture и указывать, например, расстояние между уровнями и сестрами. Можно даже менять направление "роста" дерева:

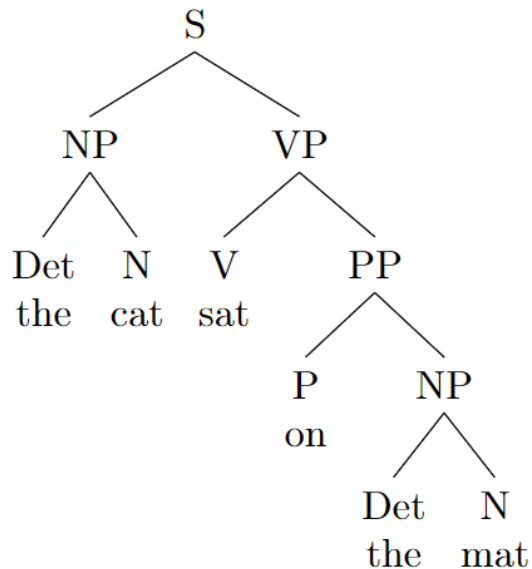
```
\begin{tikzpicture}[grow'=down]
\Tree [.NP [.Adj reverse ] [.N tree ] ]
\end{tikzpicture}
%
\begin{tikzpicture}[grow'=up]
\Tree [.NP [.Adj upside-down ] [.N tree ] ]
\end{tikzpicture}
%
\begin{tikzpicture}[grow=left]
\tikzset{level distance=60pt,sibling distance=18pt}
\tikzset{execute at begin node=\strut}
\Tree [.NP [.Adj sideways ] [.N tree ] ]
\end{tikzpicture}
```



Деревья составляющих

Еще можно настраивать **стиль отображения** для разных частей дерева:

```
\begin{tikzpicture}
\tikzset{every tree node/.style=
{align=center,anchor=north}} % позволит
отцентрировать и прикрепить кверху каждый узел в
дереве
\Tree [.S [.NP Det\\the N\\cat ]
[.VP V\\sat
[.PP P\\on
[.NP Det\\the N\\mat ] ] ] ] ]
\end{tikzpicture}
```



Деревья составляющих

Настраивать стиль можно для следующих частей:

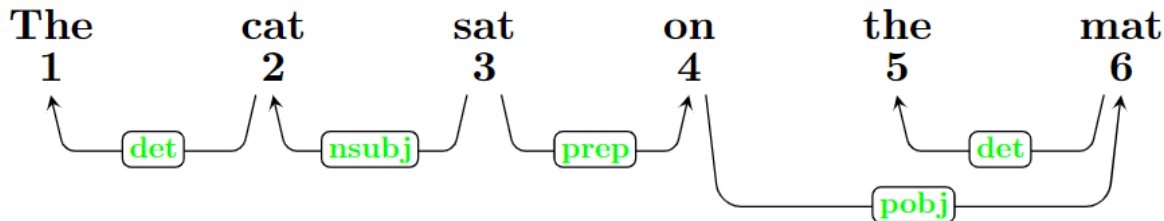
- every tree node
- every internal node
- every leaf node
- every level n node ($n=0$ - вершина дерева)

Также можно рисовать стрелки, треугольники и еще много чего.
Подробнее можно посмотреть в [мануале](#): оттуда можно просто копировать код.

Деревья зависимостей

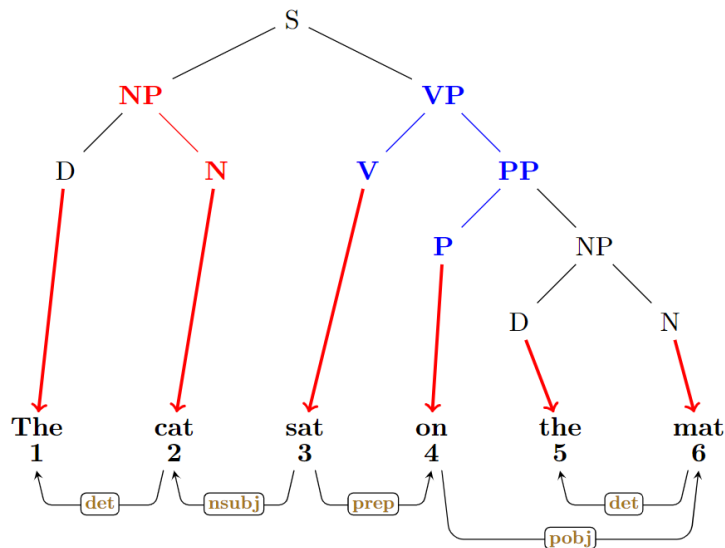
Деревья зависимостей можно рисовать с помощью библиотеки **tikz-dependency**.

```
\begin{tikzpicture}
\begin{scope}[shift={(1cm,-2.2in)}] % это у нас объект scope, которому можно задать смещение (положение в документе), но необязательно
\begin{deptext}[column sep=3em] % тут будет написан исходный текст предложения
    \textbf{The} \& \textbf{cat} \& \textbf{sat} \& \textbf{on} \& \textbf{the} \& \textbf{mat} \\
    \textbf{1} \& \textbf{2} \& \textbf{3} \& \textbf{4} \& \textbf{5} \& \textbf{6} \\
\end{deptext}
\end{scope}
\depedge[edge below]{3}{2}{\bf\textcolor{green}{nsubj}} % а теперь отрисуем связь между вершиной 3 и зависимым 2. Ее имя -
nsubj, цвет - зеленый.
\depedge[edge below]{4}{6}{\bf\textcolor{green}{pobj}}
\depedge[edge below]{3}{4}{\bf\textcolor{green}{prep}}
\depedge[edge below]{2}{1}{\bf\textcolor{green}{det}}
\depedge[edge below]{6}{5}{\bf\textcolor{green}{det}}
\end{tikzpicture}
```



Деревья зависимостей

Можно **совместить** дерево зависимостей и составляющих (можете просто скопировать этот огромный код в свой документ и попробовать разобраться, что там где):

[illegible]

Глоссирование

Наконец, есть и даже несколько библиотек для **глоссирования**. Очень простая библиотека - `\usepackage{linguex}`. Чтобы написать глосс с ее помощью, достаточно:

```
\exg. ni- c- chihui -lia in no- piltzin ce calli \\  
I it make for to-the my son a house \\  
'I made my son a house.'
```

ni- c- chihui -lia in no- piltzin ce calli
I it make for to-the my son a house
'I made my son a house.'