

BY STULK3

ШАБЛОН ДЛЯ КУРСОВОЙ НА LATEX

с пояснениями

Добавление текста,
изображений, формул,
листинга кода и
специальных символов.

Содержание

Задание	3
Как быстро начать работу с шаблоном	3
Начало работы	4
1 Структура шаблона	4
2 Как писать символы	5
3 Как писать текст	5
4 Как вставлять рисунки	6
5 Как вставлять таблицы	8
6 Как работать с математикой	10
6.1 Математические формулы	10
6.2 Дроби	10
6.3 Интегралы	10
6.4 Сумма и произведение	10
6.5 Пределы	11
7 Как вставлять листинг кода	11
8 Как вставлять специальные символы	16
9 Руководство	17
10 Заключение	18
Источники	19

Задание

Написать курсовую/диплом/отчет и разобраться в работе L^AT_EX .

Как быстро начать работу с шаблоном

Если вы хотите быстро начать, нужно:

1. зайти в `main.tex` и удалить/закомментировать строку с `\include{tutorial}`.

```
%%%%%%%%%%%%% Тutorial %%%%%%%%%%%%%%  
\include{tutorial}  
%%%%%%%%%%%%%
```

Рис. 0.1. include tutorial

2. загрузить свой титульный лист в формате pdf и назвать его `titlepage`, либо если титульный лист не нужен удалить/закомментировать строку с `\includepdf`.

```
%%%%%%%%%%%%% Титульный лист %%%%%%%%%%%%%%  
\includepdf[pages=-]{titlepage.pdf}  
  
% Если нужно вставить свой титульный лист, то загрузите его в  
% формате .pdf и переименуйте на titlepage, он вставится в начало  
% документа  
%%%%%%%%%%%%%
```

Рис. 0.2. include titlepage.pdf

3. Открываете `content.tex` и начинаете работу над отчётом.

На этом все, далее вам будет доступны ещё две страницы - содержание и источники. Содержание будет редактироваться автоматически, источники можно либо убрать, либо в файле `refs.bib` указать их.

Начало работы

Итак, вы скачали этот шаблон (за что вам огромное спасибо), но совершенно не разбираетесь как работать с \LaTeX , что же делать? Давайте разбираться.

1. Структура шаблона

Самый важный файл в шаблоне - **main.tex**. Его нельзя удалять, изменять нужно с осторожностью. Именно он является главным исполняемым файлом, в него включаются все остальные **.tex** файлы и библиография.

Шаблон имеет 3 папки - **Images**, **Listings**, **Settings**.

- В папке **Images** содержатся изображения, которые войдут в документ. Если вам нужно вставить изображение - прежде всего загрузите его сюда.
- В папке **Listings** содержатся файлы с кодом, которые специальной командой можно быстро вставить в документ в качестве листинга.
- В папке **Settings** содержатся технические **.tex** файлы с настройками. Поскольку их довольно много, чтобы не мозолить глаза они были перемещены сюда.

В папке **Settings** содержатся два важных **.tex** файла - **packages** и **format**. Первый отвечает за все пакеты нужные для работы шаблона, второй за формат текста, шрифты, изображения и многие другие вещи.

Если вам нужно добавить какие-то фичи или же изменить положение текста, шрифт и тд - меняйте эти файлы под себя.

Далее идут два опциональных **.tex** файла - **tocpage** и **bibpage**. Страница Содержания и страница Источников. Эти страницы отделены поскольку они

не всегда нужны, поэтому чтобы не загромождать шаблон были вынесены в отдельные файлы.

Никто не мешает вам убрать их и вводить как Содержание, так источники вручную.

Свой отчет нужно вводить в файл **content.tex**.

2. Как писать символы

Чтобы не возникло проблем сразу обрисую ситуацию - на LaTeX нельзя просто вводить символы, которые используются в языке Tex. Это символы:

$\$ \& \% \# _ \{ \} \sim \wedge \backslash$

Чтобы использовать их обязательно нужно вводить их так:

$\backslash \$ \backslash \& \backslash \% \backslash \# \backslash _ \backslash \{ \backslash \} \backslash \sim \backslash \wedge$

Это касается и обычного текста, и названий заголовков.

3. Как писать текст

Писать текст очень легко. Все новые абзацы начинаются с красной строки. Размер текста и поля по краям по ГОСТ'у.

Можно начать с новой строки используя двойную косую черту.

А начать новый абзац можно отступив одну строку между текстом.

...социально-экономическое развитие требует от нас анализа экономической целесообразности принимаемых решений? Кистяковский (2014)

Также можно цитировать командой **cite**.

Чтобы добавлять заголовки - воспользуйтесь командой **section**, подзаголовки - **subsection**. Они сразу добавятся в Содержание.

Если вам нужно добавить заголовок или подзаголовок без нумерации - добавьте в конце *. Чтобы они появились в Содержании воспользуйтесь **addcontentsline** как это показано тут.

```
\section*{Начало работы}
\addcontentsline{toc}{section}{Начало работы}
```

В итоге получится полностью рабочий заголовок в содержании.

4. Как вставлять рисунки

Можно вручную вставлять рисунки прописывая каждый параметр:

```
\begin{figure}[!htb]
  \centering
  \includegraphics[width=\textwidth]{image1.jpg}
  \caption{Используйте Rich Text}
  \label{fig:image1}
\end{figure}
```



Рис. 4.1. Используйте Rich Text

Параметр **width** задаёт ширину рисунка. В этом случае она равна ширине текста (**textwidth**). Перед **textwidth** можно указать значение от 0.1 до 1. А можно использовать кастомную команду **image**:

`\image{image2.jpg}{Подпись к рисунку}{0.7}`

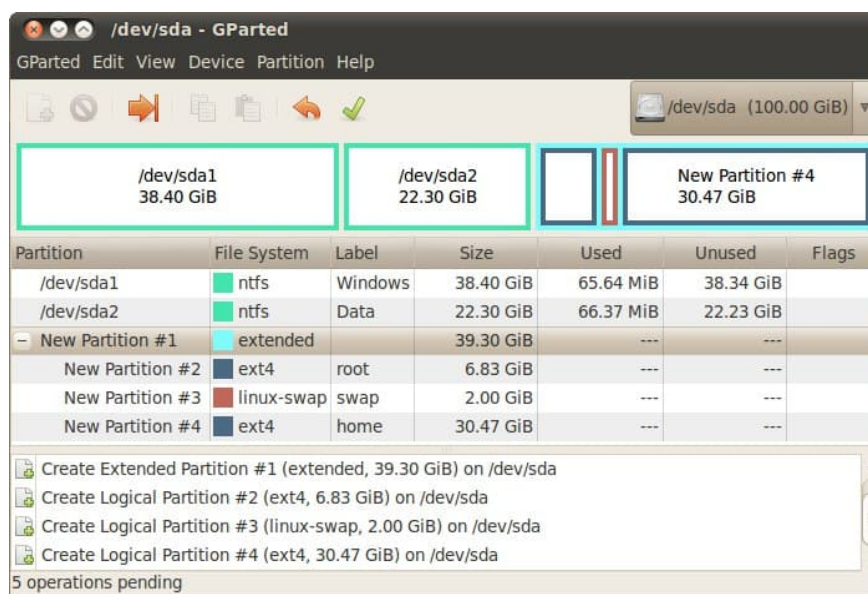


Рис. 4.2. Подпись к рисунку

Она позволяет удобно и быстро вставлять и регулировать размер изображения, для того чтобы исключить проблемы с разметкой изображений на странице. Размер можно менять изменяя второй параметр, который принимает любое число с плавающей точкой, но я рекомендую использовать лишь в диапазоне от 0.1 до 1, поскольку команда автоматически скейлит маленькое изображение, а с большими могут возникать проблемы.

`\image{image2.jpg}{То же изображение, но меньше}{0.35}`

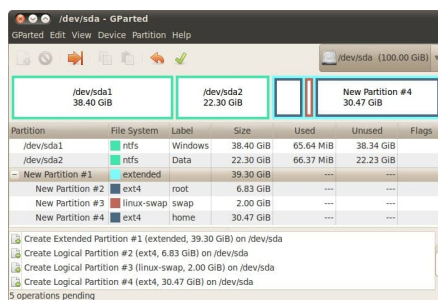


Рис. 4.3. То же изображение, но меньше

К тому же команда автоматически проставляет нумерацию рисунков.

5. Как вставлять таблицы

К сожалению таблицы это комплексная вещь, поэтому придется задавать все параметры вручную.

Однако очень рекомендую [этот сайт](#), тут можно очень легко делать таблицы и потом вставлять сгенерированный код в свой документ.

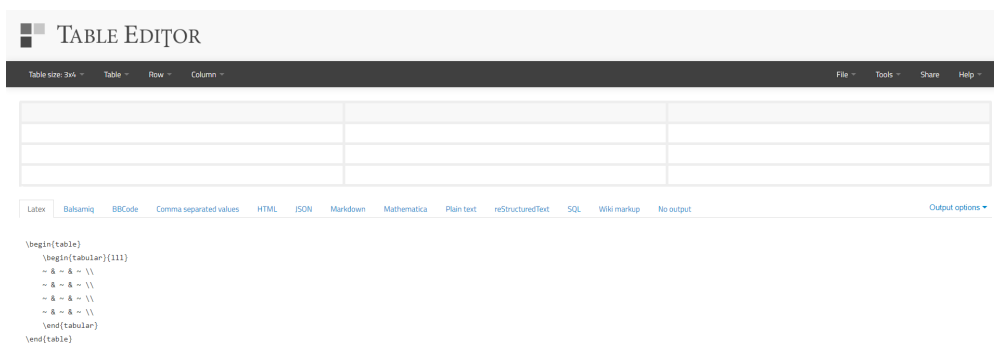


Рис. 5.1. Table Editor


```

\begin{table}[h!]
\centering
\begin{tabular}{|c|c|c|c|}
\hline
Столбец1 & Столбец2 & Столбец3 & Столбец4 \\ \hline
\multirow{3}{5em}{Несколько строк} & 6 & 87837 & 787 \\
& 7 & 78 & 5415 \\
& 545 & 778 & 7507 \\
& 545 & 18744 & 7560 \\
& 88 & 788 & 6344 \\ \hline
\end{tabular}
\caption{Пример работы с таблицей}
\label{table:1}
\end{table}

```

Вот пример таблицы:

Столбец1	Столбец2	Столбец3	Столбец4
Несколько строк	6	87837	787
	7	78	5415
	545	778	7507
	545	18744	7560
	88	788	6344

Таблица 5.1. Пример работы с таблицей

Больше о таблицах [тут](#).

6. Как работать с математикой

6.1. Математические формулы

Хорошо известная теорема Пифагора $x^2 + y^2 = z^2$ была доказана недействительной для других показателей. Это означает, что следующее уравнение не имеет целочисленных решений:

$$x^n + y^n = z^n$$

Другой способ вставить уравнение в текст такой: $x^2 + y^2 = z^2$. То есть уравнение нужно поместить между двумя знаками "доллара".

6.2. Дроби

При отображении дробей в строке, например $\frac{3x}{2}$, вы можете установить другой стиль отображения: $\frac{3x}{2}$. Это также верно и в обратном направлении

$$f(x) = \frac{P(x)}{Q(x)} \text{ и } f(x) = \frac{P(x)}{Q(x)}$$

6.3. Интегралы

Интеграл $\int_a^b x^2 dx$ внутри текста. Тот же интеграл на дисплее:

$$\int_a^b x^2 dx$$

Официальный туториал по интегралам можно посмотреть по этой [ссылке](#).

6.4. Сумма и произведение

Тоже оставлю [ссылку](#).

6.5. Пределы

Предел $\lim_{x \rightarrow \infty} f(x)$ внутри текста. Тот же предел на дисплее:

$$\lim_{x \rightarrow \infty} f(x)$$

7. Как вставлять листинг кода

Для листинга можно воспользоваться командой `codefromfile`. Эта команда позволяет указать файл и вставить его в документ. Может пригодиться если лень копировать и вставлять огромный код. Для того чтобы файл открывался - нужно поместить его в папку Listings.

Файл main.java

```
package com.pract4;

import javax.swing.*;
import java.awt.*;

public class Main extends JFrame {

    private static final String Team1 = "Team Spirit";
    private static final String Team2 = "PSG";

    JButton buttonMilan = new JButton(Team1);
    JButton buttonRealMadrid = new JButton(Team2);
    JLabel jlResult = new JLabel("Result: 0 X 0");
    JLabel jlLastScorer = new JLabel("Last Scorer: N/A");
    Label lWinner = new Label("Winner: DRAW");
    int team1Score = 0;
    int team2Score = 0;
    String lastScorer = "N/A";
    String winner = "N/A";

    public Main() {
        super("International 10");
```

```

setSize(500, 500);
setLayout(new FlowLayout());
add(buttonMilan);
add(buttonRealMadrid);
add(jlResult);
add(jlLastScorer);
add(lWinner);
buttonMilan.addActionListener(ae -> {
    team1Score++;
    jlLastScorer.setText("Last Scorer: " + Team1);
    onClick();
});
buttonRealMadrid.addActionListener(ae -> {
    team2Score++;
    jlLastScorer.setText("Last Scorer: " + Team2);
    onClick();
});
}

private void onClick() {
    jlResult.setText("Result: " + team1Score + " X " + team2Score);
    lWinner.setText("Winner: DRAW");
    if (team1Score > team2Score) {
        lWinner.setText("Winner: " + Team1);
    } else if (team2Score > team1Score) {
        lWinner.setText("Winner: " + Team2);
    }
}

public static void main(String[] args) {
    new Main().setVisible(true);
}
}

```

Можно воспользоваться командой `codefromfile`, а можно забить код вручную через среду `minted`:

Файл `def.py`

```
1 import numpy as np
2
3 def incmatrix(genl1,genl2):
4     m = len(genl1)
5     n = len(genl2)
6     M = None #to become the incidence matrix
7     VT = np.zeros((n*m,1), int) #dummy variable
8
9     #compute the bitwise xor matrix
10    M1 = bitxormatrix(genl1)
11    M2 = np.triu(bitxormatrix(genl2),1)
12
13    for i in range(m-1):
14        for j in range(i+1, m):
15            [r,c] = np.where(M2 == M1[i,j])
16            for k in range(len(r)):
17                VT[(i)*n + r[k]] = 1;
18                VT[(i)*n + c[k]] = 1;
19                VT[(j)*n + r[k]] = 1;
20                VT[(j)*n + c[k]] = 1;
21
22            if M is None:
23                M = np.copy(VT)
24            else:
25                M = np.concatenate((M, VT), 1)
26
27            VT = np.zeros((n*m,1), int)
28
29    return M
```

При этом добавив к нему кучу красивого оформления - как в этом листинге.

Обратите внимание, что если во вторых скобках указать вместо `text` любой другой язык, то код будет подсвечиваться:

```

// quot_rem.cpp

#include <iostream>
#include <cstdlib>

int main()
{
    using namespace std;

    int a = 0, b = 0; // Целые числа.

    cout << "a = ";
    cin >> a;

    cout << "b = ";
    cin >> b;

    cout << "quotient a:b = " << a / b << endl;
    cout << "remainder a:b = " << a % b << endl;

    return EXIT_SUCCESS;
}

```

В данном случае я указал C++. Тоже самое можно будет сделать если указать в команде codefromfile во второй скобке язык:

```

programmers = ["I'm an expert Python Programmer",
               "I'm an expert Javascript Programmer",
               "I'm a professional Python Programmer"
               "I'm a beginner C++ Programmer"
]
#method 1
for p in programmers:
    if p.find("Python"):
        print(p)
#method 2
for p in programmers:
    if "Python" in p:
        print(p)

```

Помимо команд выше можно воспользоваться **lstlisting**, которая позволяет более точно настроить параметры листинга а также указать подпись к нему. К тому же она автоматически проставляет нумерацию листингов.

```

\begin{lstlisting}[language=Java, caption=Кириллица в листинге не имеет подсветки]
class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello, мир!"); // Комментарий на кириллице
        for (int i = 0; i < 100; ++i) {
            System.out.println(i); // Latin comment
        }
    }
}
\end{lstlisting}

class HelloWorldApp {
    public static void main(String[] args) {
        System.out.println("Hello, мир!"); // Комментарий на кириллице
        for (int i = 0; i < 100; ++i) {
            System.out.println(i); // Latin comment

```

```

    }
  }
}

```

Листинг 1. Кириллица в листинге не имеет подсветки

```

SELECT Orders.OrderID, Customers.CustomerName
FROM Orders
INNER JOIN Customers ON Orders.CustomerID = Customers.CustomerID;

```

Листинг 2. Пример на SQL

```

quicksort :: (Ord a) => [a] -> [a]
quicksort [] = []
quicksort (x:xs) =
    let smallerSorted = quicksort [a | a <- xs, a <= x]
        biggerSorted = quicksort [a | a <- xs, a > x]
    in smallerSorted ++ [x] ++ biggerSorted

```

Листинг 3. Пример на Haskell

```

programmers = ["I'm an expert Python Programmer",
               "I'm an expert Javascript Programmer",
               "I'm a professional Python Programmer",
               "I'm a beginner C++ Programmer"
]

#method 1
for p in programmers:
    if p.find("Python"):
        print(p)

#method 2
for p in programmers:
    if "Python" in p:
        print(p)

```

Листинг 4. Листинг из файла

8. Как вставлять специальные символы

αA - греческие символы, λ ; Λ - физические величины, \exists ; \forall - логические символы

По этой [ссылке](#) можно посмотреть остальные символы. [Здесь](#) - математические операторы.

9. Руководство

[Ссылка](#) на полное введение в Latex на русском языке.

10. Заключение

Вот вы и закончили написание курсовой/диплома/отчета на \LaTeX , вы великолепны!

Источники

Богдан Александрович Кистяковский. Социальные науки и право. Directmedia, 2014.

Евгений Михайлович Ландау, Лев Давидович и Лифшиц. Теоретическая физика. Рипол Классик, 1958.

Usage statistics of content languages for websites, 2017. URL http://w3techs.com/technologies/overview/content_language/all. Last accessed 16 September 2017.

Donald E. Knuth. Literate programming. The Computer Journal, 27(2):97--111, 1984.