

Aula zero

Tássio Naia dos Santos

22 de março de 2009

Finalmente, o primeiro parágrafo do texto! Mais uma frase! E ainda outra¹!

O $\text{\LaTeX}$ não se importa com o número de linhas que você deixa entre os parágrafos no arquivo original. Basta deixar uma, e ele iniciará um novo parágrafo, com espaçamento correto e tudo o mais.

Eis um novo parágrafo. Note que, se não houver uma linha em branco entre as frases (no texto-fonte), elas serão colocadas no mesmo parágrafo.

Uma coisa parecida acontece com os espaços. Deixar um ou muitos espaços entre as palavras, não fará diferença na saída gerada pelo $\text{\LaTeX}$.

1 Mãos à obra—um pouco de filosofia

Até o presente momento, o $\text{\LaTeX}$ só se mostrou um jeito deselegante e bagunçado de escrever documentos. Algumas perguntas (e brados de indignação) são bastante naturais nesse ponto. Provavelmente você já pensou que gastaria muito menos tempo para gerar a mesma saída com o seu velho e conhecido editor de texto, e é bem possível que tenha torcido o nariz para essa complexidade desnecessária que enche o documento de sequências (ai, o trema caiu!) de comandos.

Por outro lado, com alguma sorte você ficou agradavelmente surpreso com a facilidade com que o $\text{\LaTeX}$ gerou uma nota de rodapé, e efetuou a correta divisão silábica de uma palavra ou outra alguns parágrafos atrás.

Pode ser que você esteja um pouco encucado a respeito da quantidade de mudanças que o $\text{\LaTeX}$ fez no texto. Se você, como a grande maioria das pessoas que usam processadores de texto, está familiarizado com a rotina de seleção de estilos, tipos², e configurações de margens e espaçamentos, é bem provável que esteja se perguntando “*mas quem é esse $\text{\LaTeX}$ para dizer o que é melhor para o *meu* texto?*”

Vamos por partes. Se você já deu uma olhadela no arquivo-fonte deste texto, deve ter notado que, excluindo-se os comentários, o documento está recheado de comandos que dão um *significado* à pedaços do texto. Essa abordagem é

¹Certo, nesse passo podemos ir bem longe...

²As famosas *fontes*.

fundamentalmente diferente da visão orientada a formatação, que é a base de editores WYSIWYG³ de texto.

Para tornar uma longa história curta, ao empregarmos L^AT_EX como ferramenta de geração de nossos textos, estamos basicamente separando a *composição* e a *formatação* do texto. E que vantagem nós levamos nisso? Bem, todo documento estruturado carrega um determinado padrão. Na esmagadora maioria dos casos, queremos que todas as seções de nossos relatórios, ou capítulos de nossos livros, ou legendas de nossas tabelas sejam impressas com a mesma formatação. Não importa que sejam da fonte x , tamanho y , desde que sigam consistentemente o padrão.

Ora, quando dotamos os elementos de nosso texto de *significado*, fica muito mais simples fazer isso⁴. Basta dizer que *as notas de rodapé devem ser numeradas sequencialmente, e colocadas nem uma pilha (de altura máxima z) na parte inferior da página; devem ser escritas em uma fonte $w\%$ menor do que a da fonte usada no corpo do texto; e devem apresentar um pequeno traço sobre a primeira delas, para dividi-las do texto*. Sacou?

Mas a história não acaba aí. Imagine que pudéssemos apenas nos preocupar com o conteúdo do texto, e que depois fosse possível entregar o manuscrito a um diagramador profissional—amigo nosso—que cuidaria de todos os detalhes da aparência do texto final: do espaçamento entre as palavras em uma linha até a distribuição das imagens pelo texto e a confecção dos índices, da conferência das inter-referências das seções até a formatação das referências bibliográficas! Bem, você sabe aonde eu quero chegar. . . o L^AT_EX é camarada.

Se essa carreira de argumentos não foi o suficiente para impressionar, chegou a hora de pegar pesado—estou falando da escrita de fórmulas e expressões matemáticas. A necessidade de um sistema que permitisse a geração automática de equações de forma decente foi uma das motivações da criação do T_EX, base sobre a qual o L^AT_EX se apoia⁵. O resultado disso é que podemos exclamar⁶, sem a menor dificuldade:

$$\oint \vec{B} \cdot d\vec{s} = \mu_0 \epsilon_0 \frac{d\phi_E}{dt} + \mu_0 i_{enc}.$$

Mas chega de enrolação—vejamos um pouco de L^AT_EX pra valer. As próximas duas seções introduzem algumas ideias básicas, fundamentais para o uso dessa ferramenta poderosa.

Um último ponto antes de passarmos à técnica. Gosto de fazer a seguinte comparação: aprender a usar o L^AT_EX é como aprender a usar uma calculadora científica (ou uma planilha). Não é trivial, mas faz toda a diferença.

³What You See Is What You Get.

⁴Tão simples que até um comutador pode fazer!

⁵(Ai essa reforma ortográfica!) O L^AT_EX é um conjunto de macros T_EX que facilita seu uso e estende sua funcionalidade.

⁶Após uma rápida consulta ao livro-texto de física III. . .

2 Ambientes

Ambientes são uma parte importante de um documento. Não é incomum identificarmos partes do texto que apresentam características peculiares, que as diferenciam do texto que as rodeia. Ambientes (*environments*) em L^AT_EX são sempre iniciados por “\begin” e terminados por “\end”. Alguns exemplos de ambientes são listas, tabelas, citações e figuras. Em geral, ambientes

- envolvem pequenos trechos do documento (que devem receber tratamento especial);
- são aninháveis (podemos ter um ambiente dentro de outro ambiente⁷); e
- similarmente a boa parte dos comandos, como
 1. \emph;
 2. \footnote;
 3. \section; etc.

podem receber parâmetros.

3 Alguns comandos

Antes de dar a aula por terminada, há ainda alguns comandos a apresentar. Usamos o L^AT_EX para formatação de texto, mas, por vezes, há trechos do texto que não queremos processados por ele. Um típico caso ocorre quando queremos inserir o trecho de um programa em nosso documento. Programas geralmente são escritos em uma fonte monoespaçada⁸, e frequentemente usamos caracteres em branco para estruturar (endentar) o código, tornando mais fácil a sua compreensão. Mas se apenas escrevermos o código, digamos, de uma função escrita em C:

```
int fatorial (int n) if (n == 1) return 1; return n * fat (n - 1);
```

não é nenhuma surpresa que o resultado não seja o desejado.

Quando queremos que parte do texto seja mantida sem formatação, ou seja, quando queremos que o L^AT_EX não o processe, recorremos ao ambiente *verbatim*.

```
int fatorial (int n) {  
    if (n == 1)  
        return 1;  
    return n * fat (n - 1);  
}
```

Existe ainda outro modo de inserir texto não-formatado no documento, sem que para isso tenhamos que iniciar um ambiente. É o comando `\verb`. Para

⁷Basta observar que o próprio documento é um ambiente: *document*.

⁸Um tipo monoespaçado é aquele em que todos os caracteres têm a mesma largura.

usá-lo, basta encerrar a sequência de caracteres que queremos “*verbatimizar*” entre duas ocorrências de um caractere delimitador.

Outro exemplo: `SOMA    MOVE.B    #5,D2`.

Outro comando que é usado com frequência é a ênfase (`\emph`). Esse comando põe em destaque o texto que lhe é passado como argumento.

4 Resumo da ópera

Segue uma lista dos comandos apresentados nessa “aula”, com uma pequena descrição.

- `%` permite inserir comentários que se estendem até o fim da linha;
- `\documentclass{}` especifica as configurações básicas do documento que vamos escrever;
- `\begin{}` (`document`, `itemize`, `enumerate`, `verbatim`) e `\end{}` delimitam ambientes;
- `\section{}` marca o início de uma seção;
- `\emph{}` enfatiza;
- `\footnote{}` insere uma nota de rodapé;
- `\dots`, `\textbackslash`, `\TeX`, `\LaTeX`, `\%` peritem imprimir ... \ `\TeX` `\LaTeX` (respectivamente); e
- `\verb` impede que um certo texto seja processado.

E por hora basta. Até a próxima!