

Manual de LaTeX/Texto completo

< [Manual de LaTeX](#)

Manual de LaTeX

por es.wikibooks.org

Sumario

Introducción

- Introducción
- Qué es **L^AT_EX** exactamente
- Razones para usar **L^AT_EX**
- Referencias
- La escritura en LaTeX
- Uso con un editor independiente
- Instalación
 - Instalación en Linux
- Actualizando *tetex* o *texlive*
- Editores de texto
 - Instalación en Windows
- Repositorios
- Editores
 - Instalación en OS X
- Programas complementarios
- «Motores»
- Gráficos

La estructura de un documento en LaTeX

- La estructura
 - Preámbulo
 - Paquetes
 - Cuerpo
 - Órdenes o macros
 - Principios básicos de la escritura
- El preámbulo
- La estructura
 - Preámbulo
 - Paquetes
 - Cuerpo
 - Órdenes o macros
 - Principios básicos de la escritura
- Clases de documento
- Diseño del documento
 - Curriculum vitae
 - Presentaciones
 - Cartas
 - Exámenes y apuntes
 - Paquetes comunes
- Marcas de agua: draftwatermarks
 - Modo de uso
- Referencias

El cuerpo
Portada del documento

¿Qué hacemos si queremos una portada diferente?

Secciones de un documento
Estilos de página
Índices
Glosario
Familia
Serie
Forma
Tamaño
ASCII puro
Espacio de no separación
Sangría de primera línea y espacio entre párrafos
Espacios tras punto
Espacios con extensión infinita
Blancos verticales
Referencias
El entorno *tabular*
Tabla de base
Texto en las tablas
El entorno *tabular**, control de la anchura de una tabla
Color
TikZ
Creando Líneas
Símbolos
Acentuación en modo matemático
Uso básico
 Definiendo de la partitura
 Escribiendo la partitura
Uso básico
Compuestos alifáticos
Carbociclos
Heterociclos
Técnicas avanzadas
Referencias

Introducción

En este capítulo estudiaremos algunos detalles fundamentales que el lector debe conocer sobre **L^AT_EX**: Su instalación en diversas plataformas y la compilación de documentos escritos en **L^AT_EX**. También mencionaremos algunos programas útiles que te ayudarán tanto en la escritura de tus archivos **L^AT_EX** como en la compilación de los mismos. Algunos de los programas de los que hablaremos te ayudarán también a crear y convertir gráficos y dejarlos listos para ser incluidos en tu documento.

1. Introducción a **L^AT_EX**
2. La escritura en **L^AT_EX**
3. Instalación
 1. Instalación en Linux
 2. Instalación en OS X
 3. Instalación en Windows
4. Programas complementarios

Este artículo está destinado principalmente a comparar TeX con los procesadores de texto, así como para analizar las ventajas e inconvenientes, pero no es esencial y puede omitirse.

Introducción

En primer lugar hay que señalar que **L^AT_EX** no es un procesador de textos, como lo es *Kword*, *Abiword* o *Writer* de la suite ofimática Openoffice.org. Sino más bien es un lenguaje de programación que en vez de presentarnos en pantalla una interfaz gráfica produce textos para ser leídos o impresos de una gran calidad tipográfica. Para hacernos una idea de lo que puede ser **L^AT_EX** pensemos que este es algo similar a lo que sucede con el código fuente de las páginas web. La página es escrita según unas reglas que entiende el navegador y que nos permite visualizar dichas páginas de un modo intuitivo. Puede comprobarse esto fácilmente pidiéndole al navegador web que muestre el código fuente de esta página. Como se ve lo escrito y lo visualizado no coinciden. **L^AT_EX** procede de modo similar. Lo que se escribe no es lo que se ve en pantalla. ¿Puede esto tener alguna ventaja práctica? ¿Hay alguna razón para abandonar los procesadores de texto que usamos todos los días para realizar nuestros trabajos? Lo que sigue intentará mostrar bajo qué circunstancias **L^AT_EX** puede representar una ventaja frente a otros modos más amigables para trabajar los documentos escritos.

Imagine usted lo siguiente: Tiene que escribir un largo trabajo de investigación. Todo lleno de fórmulas matemáticas, de citas a pie de página y muchos libros de referencia. Imagine que tiene que editar un enorme documento de casi mil páginas y que está dividido en 70 pequeños archivos. ¿Cómo hacer un índice a partir de 70 archivos? ¿Cómo evitar que haya cambios de fuentes o de formato indeseables y que todas las páginas tengan los mismos márgenes? ¿Cómo proceder para que el programa editor no haga cosas extrañas e irreversibles? ¿Cómo saber si todas las comillas que he abierto han sido cerradas? Estos ejemplos no son casos aislados, ni hipotéticos, sino que se refieren al uso cotidiano de los procesadores de textos en entornos académicos y universitarios. Para estos usos los procesadores de textos pensados para las oficinas se quedan un tanto cortos en cuanto al respeto de las más elementales normas tipográficas. Ahora bien, como cada uno de nosotros no es siempre un profesional de la tipografía es conveniente que alguien haga por nosotros dicho trabajo tipográfico. Aquí es donde hace su aparición el entorno **L^AT_EX**.

Para hacerlo más comprensible esta aparición sigamos con los actos de imaginación. Imagine ahora que dispone usted de un programa al cual sólo hay que darle las ordenes y este las realiza por usted. Por ejemplo, le dice usted que esto que escribe es un libro o una obra de teatro o un artículo de astronomía o una carta. Como verá cada una de estas obras tiene una estructura propia que **L^AT_EX** de antemano sabe y maneja según plantillas que respetan la cabalidad las normas tipográficas. Imagine también que dentro de un libro usted le señala a **L^AT_EX** que este es el título, este el autor y la fecha. Con estos elementos **L^AT_EX** compondrá la portada. Además, puedo indicarle que esto es un capítulo (no sé si acabará siendo el primero o el vigésimoquinto), aquí una sección (no sé cuál es su número exacto), esto una nota al margen que siempre debe estar a la altura de tal línea (aunque introduzca cientos de líneas más después), aquí una nota al pie; quiero además que en el encabezado de cada página vaya el número del capítulo y el nombre de la sección, etc. El usuario da las órdenes y **L^AT_EX** se encargará por usted de formatear el documento según las instrucciones que le ha dado. El usuario sólo se dedica a escribir el texto, de la maquetación se encarga **L^AT_EX**. Los resultados finales de esta maquetación son en la mayoría de los casos muy superiores a los que el usuario obtiene manualmente con procesadores de texto WYSIWYG^[1].

En general los usuarios que utilizan los procesadores de textos lo hacen como si fuesen máquinas de escribir sofisticadas. Sin embargo, un texto es un producto que debe respetar un mínimo de normas para su legibilidad. La belleza estética de un texto no es criterio suficiente para establecer su comprensibilidad, lo que importa más bien es su estructura lógica. Un escrito lógicamente estructurado es mucho más legible que un texto solo estéticamente agradable. En este aspecto el uso de **L^AT_EX** representa para el autor una enorme ventaja, ya que al escribir los textos pensando que hay que estructurarlos en vista de su legibilidad, se logra que lo escrito se le presente al lector lo que se quería decir. Desde este punto de vista **L^AT_EX** es un entorno de trabajo casi insuperable por poder de edición de documentos con alta legibilidad.

Otro ejemplo de como se procede con textos estructurados que no coinciden con lo que usted ve en pantalla o en forma impresa. Imagine usted que ha realizado la traducción de una obra medieval y la ha maquetado a dos columnas. En cada página la primera columna contiene el texto latino y enfrente la traducción castellana sincronizada con la versión original. La traducción consta de unas 600 páginas. Usted termina la traducción del texto, pero le dicen que este debe ser presentado ahora no en columnas sino que la página izquierda debe contener el texto original y la página derecha la traducción. Y debe entregarlo maquetado al día siguiente. ¿Qué hacer? ¿Cuántas horas perdidas? ¿Tendrá usted que dejar de dormir para cumplir con los plazos? En esta ocasión **L^AT_EX** presenta todas sus ventajas. Que un texto esté organizado en dos columnas o en páginas enfrentadas no significa ningún problema. La lógica de la traducción es que a cada texto latino corresponda su traducción castellana. Eso se indica en cada párrafo: aquí el texto original y esta es su traducción. Para obtener aquello solicitado lo único que habría que decirle a **L^AT_EX** es que ordene en columnas o en páginas. Para lograr esto solo basta con cambiar una instrucción de tres letras por otra de tres letras. Vale decir, entre dos o tres segundos. **L^AT_EX** es el perfecto sirviente que cumple con las órdenes que le damos. Lo único que hay que aprender es precisamente las órdenes para que **L^AT_EX** las cumpla por nosotros. Además estas instrucciones son las mismas desde hace años. Por lo que no hay que estar siempre aprendiendo nuevos comandos conforme aparezcan nuevas versiones más actualizadas. Lo que aprendí hace diez años sigue tan vigente hoy como dentro de veinte años más.

Qué es \LaTeX exactamente

\LaTeX es un sistema de tipografía que permite crear documentos con un aspecto completamente profesional de una forma sencilla. La idea principal es que el autor se centre en el contenido y no en la forma del documento. Para lograr esto, \LaTeX está provisto de una serie de macros y estilos predefinidos.

Vamos a poner un ejemplo básico: en una herramienta de procesamiento de texto estándar para hacer el título de una sección, la mayoría usa comandos para modificar la forma, por ejemplo, se pone en negrita, subrayado y a tamaño 16. Con \LaTeX , el aspecto del documento es independiente del contenido: el título de la sección se marca con el comando `\section` y \LaTeX se encargará de formatearlo correctamente cuando sea impreso y mostrado según la plantilla de documento que se emplee.

Hay disponible una gran variedad de paquetes para facilitar la representación de fórmulas matemáticas, notación musical, fórmulas químicas, circuitos electrónicos y mucho más. Concretamente, \LaTeX es realmente fantástico escribiendo fórmulas matemáticas, se pueden hacer cosas como $\int_{-N}^N e^x dx$ con gran facilidad, incluso si la fórmula es realmente simple, una vez usado, no querrás hacerlo de otra manera.

En otros términos, \LaTeX es una serie de instrucciones (macros) basado en el sistema de edición de bajo nivel TeX. En su origen fue especialmente diseñado para la composición de textos científicos, sobre todo para aquellos que deben incluir una gran cantidad de fórmulas matemáticas. Sin embargo, las posibilidades que \LaTeX ofrece hacen de él un programa idóneo para componer textos de cualquier índole, y más aún si estos son muy grandes. De hecho, puede afirmarse que si algo puede ser impreso se puede realizar con la combinación \LaTeX y TeX.

\LaTeX combina inigualablemente la sencillez de edición, la calidad tipográfica y la facilidad para garantizar una buena estructura y organización del documento, todo esto de manera casi automática. Así, uno no tiene que hacer más que concentrarse en el contenido del documento y en señalar unas cuantas instrucciones para que \LaTeX haga lo que las instrucciones le indiquen: hacer una portada, un índice de contenidos, poner las notas a pie de página, insertar una imagen, etc. Por ejemplo, en lugar de seleccionar manualmente el tipo de fuente que usaremos para el título de una sección, su tamaño y su estilo (y de recordar todo esto para cuando se llegue el momento de iniciar otra sección), en \LaTeX sólo tenemos que indicarle que dentro del documento que en un determinado lugar se inicia una sección, así basta con escribir `\section{Nombre de la sección}`, y con ello obtendremos un verdadero título de sección con todo y su numeración, con independencia de las secciones que podamos insertar antes o después de esta instrucción de estructura del texto.

No obstante, lo primero con lo que nos vamos a encontrar es que \LaTeX no es, por sí mismo, uno de esos programas que ofrecen una interfaz de usuario intuitiva, como lo hacen los programas de tipo WYSIWYG, sino que nuestra escritura en \LaTeX consiste en una serie de comandos cuyo efecto no será visible hasta después de una compilación. Desde este punto de vista, \LaTeX es menos "amigable" que otros editores de texto. Sin embargo, no cuesta realmente mucho trabajo aprender a usar \LaTeX e identificarse con él, —menos aún considerando la gran cantidad de documentación que a propósito existe—, y tomando en cuenta los resultados que pueden alcanzarse con este programa, \LaTeX termina siendo, en la mayoría de los casos, la mejor opción. Para que el lector se convenza de esto bastará que siga leyendo este libro, lleno de ejemplos de lo que se puede hacer con \LaTeX .

Razones para usar \LaTeX

Uno se puede preguntar qué ventajas conlleva el uso de \LaTeX :

- No hay que recordar que estilo se usó para las secciones previas, las subsecciones, los capítulos y demás.
- Si la fuente del título de sección resulta ser demasiado grande no es necesario cambiarla en todo el documento, se puede hacer con una modificación sencilla.
- Al estar el documento marcado con secciones, subsecciones y demás, resulta más sencillo crear tablas de contenido, de hecho, se crean automáticamente.
- Es software libre, se distribuye bajo la [Licencia Pública del Proyecto LaTeX \(LPPL\)](#).
- Funciona bien en cualquier máquina sea cual sea su sistema operativo o el procesador. Hay versiones de \LaTeX para casi todos los sistemas y arquitecturas, la mayoría gratuitas.
- Automatiza muchos procedimientos mecánicos como la autonumeración de fórmulas, la generación de listas y la creación de índices de contenido, de tablas, figuras y terminológicos. Entre otras muchas cosas más.
- Permite el uso de bases de datos bibliográficas con BibTeX (<http://www.bibtex.org>). Con el consiguiente ahorro de tiempo a la hora de citar textos y hacer listados de publicaciones. Basta con hacer una vez la base (en modo texto) y \LaTeX se encarga de incluir los datos donde corresponde.

- El resultado final es propio de un texto profesional. Y hay plantillas de **L^AT_EX** que cumplen automáticamente con estándares de publicación científica de muchas revistas.
- Es un programa que con el tiempo mejora la calidad de la salida a pantalla o impresora, pero las instrucciones siguen exactamente iguales, por lo que no es necesario estar aprendiendo cada dos por tres a usarlo. En teoría un texto escrito hoy podría ser procesado exactamente igual dentro de cien años.
- El tamaño de los archivos escritos en **L^AT_EX** son mucho más pequeños que un archivo escrito en un procesador común. Además de ser escritos en modo texto, por lo que pueden ser leídos en cualquier editor con independencia de la arquitectura y sistema operativo.

El usuario no necesita ser un profesional de la tipografía para realizar sus documentos. A modo de ejemplo: ¿cuál es el número máximo de letras que puede contener una línea para que el lector no se canse? La gran mayoría lo ignora. Las razones para usar un sistema de procesador de textos visual es su facilidad de uso. Pero, a la hora de realizar textos elaborados como libros, tesis de grado, ponencias, etc. se muestran sus limitaciones. En definitiva un procesador de textos es una enorme máquina de escribir donde el usuario tiene que introducir manualmente todos los formatos. Y, usualmente, el criterio es más bien estético y no tipográfico, es decir, creemos que un texto bello es sinónimo de legible. Pues bien, eso no es correcto, la tipografía es un arte difícil de manejar. Lo mejor en este caso es dejar en manos de un profesional la maquetación de los documentos.

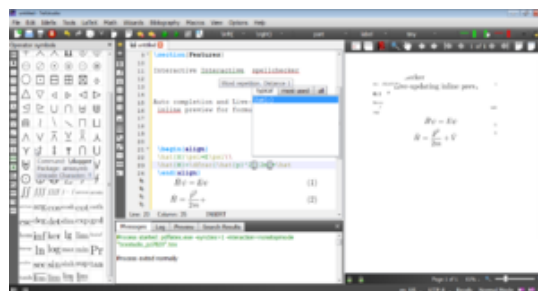
Referencias

1. WYSIWYG significa "What You See Is What You Get", es decir, "Lo que ve es lo que obtiene".

La escritura en LaTeX

Una de las características más importantes de LaTeX es que los autores trabajan con archivos de texto. Dado que LaTeX es un lenguaje de marcas, en estos archivos se combinan órdenes y texto. Para editarlos, puede servir cualquier editor, pero lo recomendable es emplear uno que esté adaptado a las necesidades propias de LaTeX.

De hecho, actualmente hay entornos para la edición de LaTeX que integran la edición, la generación del documento y la visualización en PDF. Ejemplos son TeXshop, TeXworks, TeXnicCenter, TeXmaker,... Con ellos no son necesarios los pasos siguientes ni el uso de la consola o terminal, como tampoco cuando se emplea un servicio en línea como Overleaf (<https://www.overleaf.com/>) o ShareLaTeX (<https://es.sharelatex.com/>).



TeXstudio es uno de los entornos gráficos disponibles

Uso con un editor independiente

Para escribir en **L^AT_EX** es necesario tener un programa para crear ficheros de formato `.tex`, que no es más que un formato de texto de código ASCII.

El archivo `.tex` contendrá todas las instrucciones que serán procesadas por

L^AT_EX para producir un archivo de salida. Por ejemplo, si hemos creado el archivo `ejemplo.tex`, y queremos compilarlo en formato PDF, abrimos una terminal o símbolo del sistema, nos ubicamos en la dirección donde se encuentre nuestro fichero y escribimos

```
pdflatex ejemplo.tex
```

Con esto obtendremos el documento `ejemplo.pdf` creado a partir de todas aquellas instrucciones contenidas en el archivo `ejemplo.tex`

El formato PDF no es el único que podemos producir. A continuación presentamos una lista de algunos otros comandos que producen distintos formatos de salida:

- | | |
|-----------------|--|
| latex | Este comando compilará el fichero <code>.tex</code> en un DVI. Este formato es el original de compilación de L^AT_EX , pero no es muy común ver documentos de este tipo. |
| pdflatex | Como ya mencionamos, este comando produce un archivo PDF. Este es quizá el formato más recomendable, pues aún cuando es de mayor tamaño que un DVI, es un formato mucho más común y de mejor calidad de visualización. Sin embargo, no podremos incluir imágenes PostScript, aunque podemos usar el comando siguiente y después convertir el resultado en PDF. |
| dvips | Este comando no procesará un fichero <code>.tex</code> , sino que a partir de un fichero <code>.dvi</code> , previamente elaborado con el comando <code>latex</code> , produce un documento PostScript (de extensión <code>.ps</code>). En |

general, estos documentos son de mayor tamaño. Pueden ser visualizados con, por ejemplo, Ghostview.

ps2pdf Este comando convierte los archivos `.ps` en archivos `.pdf`. Así es que, si por alguna razón es necesario compilar el fichero `.tex` en `.ps`, siempre es posible obtener al final un archivo `.pdf`.

Instalación

A menos que se recurra un sistema en línea, el primer paso para usar LaTeX es instalarlo en el sistema. Al ser código abierto no hay un único instalador, incluso en un determinado sistema. A cada uno de los instaladores existentes —que difieren en detalles sobre lo que se incluye y la forma como se gestiona y se actualiza— se lo llama *distribución*. La mayoría de las distribuciones actuales instalan todo lo necesario para trabajar con LaTeX, incluyendo entornos de edición.

1. [Instalación en Linux](#)
2. [Instalación en OS X](#)
3. [Instalación en Windows](#)

Alternativamente y como ya se ha señalado, puede recurrirse a un editor en línea, como:

- Overleaf (<https://www.overleaf.com/>)
- ShareLaTeX (<https://es.sharelatex.com/>)

Instalación en Linux

Actualizando *tetex* o *texlive*

Para utilizar **LaTeX** en Linux es muy sencillo. Casi todas las distribuciones la traen por defecto. Basta descargar e instalar el paquete `tetex` (proyecto discontinuado) o bien, el más reciente y actualizado `texlive`. Cada distribución en Linux maneja a su manera el listado y las dependencias de sus paquetes compilados. A modo de ejemplo, en distribuciones basadas en *Debian* bastará abrir un terminal e introducir como *root* la siguiente orden:

```
sudo apt-get install texlive-latex-extra texlive-fonts-recommended lmodern
```

Otro modo aun más sencillo que deja todo listo para ser usado es instalar un editor de **LaTeX** especializado (ver más adelante) con la siguiente orden:

```
sudo apt-get install lyx
```

En otras distribuciones como *SuSe*, *Mandriva*, *Fedora* (basadas en paquetes RPM) hay herramientas gráficas en las cuales es posible seleccionar los mismos paquetes para que sean instalados.

Una vez que lo hagas estará todo listo para comenzar con la creación y compilación de archivos **LaTeX**. Más aún, en la mayoría de las distribuciones de Linux **LaTeX** es un programa que es instalado por defecto y está esperando a que se haga uso de él.

Editores de texto

Una vez instalado **LaTeX** nos encontramos con que este no es un programa con una interfaz gráfica sino que lo que el usuario debe hacer es escribir sus documentos según ciertas reglas que posteriormente **LaTeX** convertirá a un documento legible e imprimible. Para escribir en **LaTeX** necesitarás simplemente un editor de texto. Cualquiera es adecuado para ello. Basta con que el editor de código ASCII ofrezca la posibilidad de guardar el archivo con la extensión `.tex`. Además de editores genéricos existen varios editores muy buenos que no sólo te permitirán guardar tu archivo `.tex`, sino que también ofrecen una serie de herramientas útiles para facilitar la composición. En linux estos editores podemos agruparlos en dos variantes: Aquellos en los que se ingresa directamente los comandos de **LaTeX** y que el programa compila posteriormente, obteniendo diversas salidas: PDF, DVI, PS, HTML, etc. Y un segundo grupo que es capaz de exportar sus archivos a formato **LaTeX**.

En el primer grupo encontramos, a modo de ejemplo, los siguientes programas:

- *Emacs*, muy útil, pues incluye un entorno de edición especial para **LaTeX** (y para algunos otros lenguajes) que, entre otras cosas, colorea los comandos para facilitar la visualización de sintaxis. Poderoso editor de textos, aunque, su instalación ocupa bastante espacio en el disco duro. Aunque incluye un modo de LaTeX, el sistema más completo es AUCTEX con RefTeX.

- *Vim*, similar a emacs en cuanto a coloreado de sintaxis. Posee además un plugin especial para compilar **L^AT_EX**. Su instalación es pequeña y es, además, un muy poderoso editor de textos.
- *Texmaker*, programa muy cómodo que permite insertar código y compilar mediante una interfaz de botones muy intuitiva. Es liviano y consume pocos recursos. Viene acompañado de herramientas que informan de los errores.
- *Kile*, editor especializado en **L^AT_EX** del escritorio KDE. Muy intuitivo y programable. Reconoce los comandos y sugiere autocompletación. Viene con herramientas que indican los errores y abren el archivo en el lugar preciso de dicho error para su corrección.
- *LaTeXila* es un editor LaTeX para el entorno de escritorio GNOME, se caracteriza por su facilidad de uso. Compila los documentos directamente a PDF o PostScript con un botón integrado al programa. Tiene la función de auto completado. Indica los errores que se presenten en el archivo y hace fácil encontrarlos. Permite la rápida inserción de símbolos matemáticos, científicos y caracteres griegos, es liviano en cuando a consumo de recursos del ordenador.

En el segundo grupo, exportadores a formato **L^AT_EX** tenemos a su vez dos variantes. Procesadores de textos que pueden exportar desde sus propios formatos a **L^AT_EX**. Y procesadores especializados en **L^AT_EX** que proporcionan una cómoda interfaz gráfica.

- Procesadores que exportan a **L^AT_EX**:
 - *OpenOffice.org* es capaz de exportar sus archivos (*odt* o *doc*) a **L^AT_EX**, a partir de los cuales es posible la compilación del archivo `.tex`. Además, este programa está disponible en español, por lo que la edición resulta aún más fácil. OpenOffice.org integra un editor de ecuaciones, y si con el insertas una fórmula matemática, ésta será convertida en los comandos de **L^AT_EX** respectivos para producir la fórmula en tu documento `.tex`. OpenOffice.org, es software gratuito. Hay que tener presente que actualmente el código **L^AT_EX** que genera no es muy perfecto.
 - *Abiword*, procesador de textos del escritorio GNOME. Exporta sus archivos a **L^AT_EX**, aunque su exportación no está muy de acuerdo con un archivo escrito directamente en **L^AT_EX**, pues, al igual que Openoffice.org intenta reproducir el formato original mediante comandos. Lo que no es muy propio de **L^AT_EX**, en el que se intenta más bien escribir textos lógicamente estructurados.
 - *Kword*, procesador de textos del escritorio KDE y parte de la suite ofimática Koffice. Realiza más o menos lo mismo que Abiword y Openoffice.org.
- Respecto del segundo grupo contamos en Linux con los siguientes programas:
 - *LyX*, este programa sostiene que es del tipo WYSIWYM (lo que vez es lo que quieres decir). Para aquellos que deseen contar con un editor de textos de estilo más "familiar", pueden pensar en utilizar LyX, un editor de textos que emplea **L^AT_EX** para la creación de sus textos, todo en un ambiente muy parecido a WYSIWYG, salvo que lo que ve en la pantalla no es el documento compilado. Para compilar y visualizar basta apretar un botón. Así, si alguien siente que con **L^AT_EX** se trabaja un poco a ciegas, este programa le permitirá cambiar la situación al mostrarle gráficamente todo lo que va haciendo en su documento, sin necesidad de compilar el texto y esperar hasta entonces para ver los resultados.
 - *Texmacs*, es un programa basado en *emacs* con un entorno casi WYSIWYG para **L^AT_EX**.

Como se señala estos programas harán que el trabajo resulte más sencillo e interesante, y ayudarán a ganar gusto por **L^AT_EX**. Los programas que exportan a **L^AT_EX** le permiten al usuario obtener desde archivos *odt* o *doc* el código fuente para **L^AT_EX** y desde estos archivos ir aprendiendo cual es la lógica interna de este modo de edición.

Es interesante señalar que una vez que se aprende a utilizar **L^AT_EX** se hace muy difícil utilizar algún otro programa para componer tus textos con contenido matemático (o de cualquier otro tipo).

El uso de uno u otro de los programas antes mencionados dependerá de los gustos del usuario. Sin embargo, es recomendable iniciarse con algunos programas más intuitivos como *LyX* y luego pasar a la edición pura en *texmaker* o *kile*.

Instalación en Windows

Para instalar LaTeX en Windows hay dos opciones básicas: MikTeX (<https://miktex.org/download>) y TeXLive (<https://www.tug.org/texlive/acquire-netinstall.html>). La primera es más fácil de instalar y mantener, aunque la segunda es mucho más completa y por lo general más actualizada.

Repositorios

Para instalarlo se debe bajar una aplicación de instalación que contiene los paquetes y programas básicos de **L^AT_EX**. Al correr esta aplicación, se ejecuta un asistente que guiará al usuario durante la instalación de MikTeX. Entre otras cosas, se solicita la confirmación de la dirección donde se guardarán los paquetes de **L^AT_EX**. Por defecto, el asistente creará una carpeta llamada "texmf", en la unidad de disco local, a menos que se indique lo contrario. Es conveniente que se permita la creación de la carpeta tal y como lo propone el asistente para evitar problemas más adelante. Posteriormente, mediante un programa que se instalará junto con

MikTeX, el **MikTeX Package Manager**, se podrán descargar los paquetes de **L^AT_EX** que se necesiten. Existen algunos paquetes menos estándares que no se descargan de forma automática con MikTeX Package Manager, sino que es necesario instalarlos manualmente desde páginas específicas dedicados a **L^AT_EX**.

Editores

Para editar ficheros `.tex` se pueden emplear editores como [TeXmaker](http://www.xm1math.net/texmaker/download.html) (<http://www.xm1math.net/texmaker/download.html>), [TeXnicCenter](http://www.texniccenter.org/download/) (<http://www.texniccenter.org/download/>) u otros similares. Estos cuentan con una gran cantidad de herramientas, dentro de las cuales la más útil es quizá la compilación misma del documento, implementando una serie de instrucciones para compilar el archivo en distintos formatos de salida, como lo son el PDF, el DVI y el PostScript. De este modo no será necesario abrir la consola (o Símbolo del sistema o Command Prompt) para realizar la compilación. Por supuesto, para que esto funcione, es necesario indicar previamente al editor dónde se encuentra el programa de **L^AT_EX** que debe ejecutar para compilar el archivo `.tex`. Esto se hace la primera vez que se inicia el editor, y si se ha permitido que MikTeX se instalara en la carpeta "texmf", no será necesario realizar ningún ajuste extra, ya que el editor asume que los paquetes y programas de **L^AT_EX** se instalaron en la carpeta por defecto y sólo será necesario confirmar las rutas que se señalan.

Instalación en OS X

Para **Mac OSX** (a partir de la versión 10.3) la instalación es sumamente sencilla, sólo hace falta seguir estos pasos:

1. Visite la página <http://www.tug.org/mactex/>. En ella se encuentra un compendio con las mejores, más comunes y más usadas aplicaciones, llamada MacTeX. Soporta TeX, LaTeX, AMSTeX, ConTeXt, XeTeX y muchos otros paquetes.
2. Descargue el archivo MacTeX.dmg y proceda a una instalación común y corriente. i.e. doble click en el archivo descargado y seguir instrucciones.

El paquete MacTeX contiene una distribución Tex Live 2007 completa de Tex, Ghostscript, ImageMagick y los siguientes programas: TeXShop, LaTeXiT, BibDesk, Excalibur y i-Installer. Una instalación detallada permite a los usuarios pasar por alto algunos de estos paquetes si así lo desean.

Una vez instalado el paquete, se recomienda ponerlo todo en una única carpeta (llamada por ejemplo: Tex o LaTeX, etc.) y dejarla en la carpeta Aplicaciones para mayor comodidad. También es recomendable leer el archivo README.rtf.

Programas complementarios

El sistema de TeX sigue el principio de no ofrecer una aplicación monolítica cerrada. En lugar de ello, prefiere ofrecer un conjunto de programas especializados de modo que se pueda optar por uno u otro de las diferentes posibilidades según las necesidades concretas de un determinado flujo documental. Entre los programas que la mayoría de las distribuciones ofrecen están bibtex y makeindex, para gestión de bibliografías e índices alfabéticos; algunas incluyen alternativas que pueden resultar más flexibles, como biber y xindy, respectivamente.

«Motores»

Además de los programas complementarios, hay que tener en cuenta que el propio programa que genera los documentos tiene variantes, que desarrollan el primitivo.

tex

Sin más, es el programa original. El documento generado tiene el formato dvi, que es específico de TeX y necesita un visualizador especial. Por ello, lo normal es que se convierta posteriormente a otros formatos más generales, como PostScript (también de uso limitado) y PDF (que, por el contrario, es universal).

pdftex

Es un programa basado en el anteriores que genera directamente archivos PDF. Añade, además, nuevas funciones.

XeTeX

También está basado en tex, aunque añade algunas funciones de pdftex. Su propósito principal es trabajar con Unicode y con las fuentes tipográficas instaladas en el sistema.

LuaTeX

Esta basado en pdftex y, según sus autores, será su sucesor. No solo trabaja con Unicode y las fuentes instaladas en el sistema, sino que incorpora la posibilidad de extender sus funciones con el lenguaje de programación Lua.

Son estables tex y pdftex. Tanto XeTeX como LuaTeX son bastante estables, aunque no tanto como los anteriores (ambos siguen en desarrollo activo, mientras que pdftex y, sobre todo, tex, se consideran básicamente cerrados con el fin de no introducir posibles incompatibilidades). Este manual presupone que se emplea pdftex, excepto si se indica lo contrario.

Gráficos

Resulta que, al escribir libros con contenido matemático, se nos presenta frecuentemente la necesidad de insertar gráficos, algunos de ellos previamente creados, y otros que aún no sabemos como crear para después incluirlos en nuestro documento. Por ello, conviene, sin entrar en detalles de su uso (esto lo haremos en un capítulo especialmente dedicado a la inserción de gráficos en **L^AT_EX**), mencionar algunos programas, todos gratuitos, con los cuales puedes convertir y crear tus gráficos para después utilizarlos en **L^AT_EX**.

Para la conversión de gráficos a distintos formatos una opción muy popular es *ImageMagick* (<https://www.imagemagick.org/script/index.php>). Este programa convierte gráficos a muy distintos formatos, además de que, en el proceso, puede hacer distintas transformaciones en la imagen misma. Este programa se trabaja mediante comandos en una terminal o símbolo del sistema. Si el lector prefiere un programa más visual, hay diversas opciones, como *Gimp* (<http://gimp.es/descargar/>).

Para crear gráficos, es recomendable usar los paquetes que incluye el propio **L^AT_EX**, como PSTricks y sus derivados (pst-3dplot, pst-labo, pst-func, etc.) o TikZ, dado que de esta forma se dispondrá de más posibilidades de realizar las tareas que se requieren para adaptar los gráficos al documento.

Además, existen programas que funcionan casi como un procesador de textos, pero que son capaces de transformar los textos a formato **L^AT_EX**. Algunos proyectos interesantes son los editores TeXmaker (<http://www.xm1math.net/texmaker/download.html>) LyX (<http://www.lyx.org>), que facilitan la escritura de documentos en LaTeX y es un modo más simple de iniciarse en la escritura de documentos estructurados en **L^AT_EX**.

La estructura de un documento en LaTeX

Ya se ha explicado cómo se compila un documento en **L^AT_EX**, pero no se ha hablado aún de cómo escribir el documento a compilar. En este capítulo se analiza la estructura básica de un documento, y en el siguiente capítulo se expondrán los conceptos básicos sobre la escritura de texto en **L^AT_EX**.

La estructura

La estructura de un documento en **L^AT_EX** se divide en dos grandes partes: el preámbulo y el cuerpo del texto. El siguiente ejemplo muestra un documento mínimo apropiado para el español:

```
\documentclass[spanish]{article}
\usepackage{babel}
\usepackage[T1]{fontenc}
\usepackage{textcomp}
\usepackage[utf8]{inputenc} % Puede depender del sistema o editor
\begin{document}
Aquí iría el texto del documento en sí.
\end{document}
```

Preámbulo

En el *preámbulo* se escriben las instrucciones fundamentales que indican a **L^AT_EX** qué clase de documento se va a escribir y qué características va a tener éste, así como también las que indican a **L^AT_EX** qué paquetes se deben cargar. El preámbulo siempre empezará con la instrucción:

```
\documentclass[<opciones>]{<plantilla_documento>}
```

Para definir la plantilla que se va a emplear en el documento, como por ejemplo `article` o `report`, que determinan diferentes estilos. En general, los argumentos que toma este comando son las llamadas *clases* de documento, y pueden aceptar diferentes opciones. Por ejemplo, la instrucción:

```
\documentclass[12pt, letterpaper]{book}
```

Declara que el documento es un libro, con el tamaño de letra configurado a 12 puntos y utilizando papel tamaño carta. En vez de `letterpaper`, se pueden usar otros tamaños de papel, como lo es A4 (`a4paper`).

Paquetes

Se llama paquete a una extensión del sistema básico que añade nuevas funciones. Hay, literalmente, cientos de paquetes con muy diversas funciones: inserción de imágenes (`graphicx`), paquetes gráficos (`TikZ`), internacionalización (`babel`, `polyglossia`), color (`xcolor`), música, ajedrez, ediciones críticas, secuencias de aminoácidos, etc. Todos estos paquetes deberán ser declarados con:

```
\usepackage[<opciones>]{<paquete>}
```

Donde entre los corchetes estará el nombre del paquete a usar, por ejemplo:

```
\usepackage{amssymb}
```

Para cargar el paquete `amssymb`, que proporciona símbolos matemáticos de la American Mathematical Society. Si una clase de documento o paquete que queremos cargar ofrece opciones y nosotros no especificamos la que queremos, se cargarán las opciones por defecto.

Cuerpo

El cuerpo del documento consiste en prácticamente todo lo que aparecerá en nuestra compilación. Es aquí, pues, donde escribiremos el texto verdadero. Comienza con la instrucción

```
\begin{document}
```

y termina con:

```
\end{document}
```

Todo lo que se escriba con posterioridad a esta instrucción será ignorado por **L^AT_EX** y no se compilará.

Una vez que iniciemos el cuerpo del documento debemos escribir al final de todo lo escrito la instrucción de cierre `\end{document}` aunque no hayamos terminado todo el documento, pues de otra manera tendremos un error en el proceso de la compilación y no podremos ir viendo cómo van quedando nuestros avances.

Órdenes o macros

Hay ciertos caracteres que tienen una función especial, como por ejemplo, el símbolo `\`, por el que comienzan todas las instrucciones, o las llaves y corchetes, que contienen los datos y opciones de las instrucciones. Por ejemplo, en `\title{Un título}` hay una orden (en muchos casos también llamadas macros), que es `\title` y que ajusta el título del documento con el dato (o argumento) que sigue, que es `Un título` (en este punto no se añade realmente el título al documento, sino que LaTeX tan sólo lee el dato y lo guarda para cuando haga falta).

Otro detalle que hay que destacar es que las órdenes que no van seguidas de algún argumento descartan el o los espacios que le siguen (con la excepción de las órdenes que consisten en un símbolo, como `\#`, `\%` o `\$`). Aunque hay pocas órdenes de este tipo —la mayoría tienen algún dato o consisten en un símbolo— es muy importante tener esto presente por si se diera el caso. La solución en tales casos suele pasar por añadir un «dato» vacío, es decir, un par de llaves.

Finalmente, hay que señalar que algunas órdenes no van seguidas de uno o varios argumentos, sino que operan sobre el texto que le sigue hasta que termina el bloque actual delimitado por llaves. En ocasiones, incluso, hay dos variantes que funcionan de cada uno de estos modos. En caso simple es el siguiente:

```
\textbf{texto en negrita}  
{\bfseries texto en negrita}
```

Estas dos formas son equivalentes, pero por lo general se prefiere el primer tipo.

Un tipo especial de orden es la que delimita un bloque del documento. Van siempre por pares:

```
\begin{...}  
...  
\end{...}
```

Una estructura así se llama *entorno* o *ambiente* y la más importante es justamente la que abarca el cuerpo del documento.

Principios básicos de la escritura

Todo bloque de texto separado del resto con líneas en blanco se considera un párrafo. No es el único caso en que LaTeX considera que hay un párrafo, pero sí es el más importante. En este caso, se lee el texto contenido en ese bloque y LaTeX lo procesa con objeto de encontrar las mejores divisiones de línea, los mejores guiones y el mejor espaciado posible para el párrafo. También se preocupa de encontrar el mejor punto para cambiar de página, así como de cuadrar el resultado en la página. Todo ello, naturalmente, sin necesidad de intervención directa de quien escribe.

El texto de cada párrafo se escribe de modo normal, con algunas salvedades importantes. En primer lugar, un espacio entre palabras vale lo mismo que dos, tres o cientos, siempre que no se deje una línea en blanco. De esta forma se evitan espaciados irregulares que en sistemas WYSIWYG aparecen en ocasiones al teclear por error dos espacios seguidos. En segundo lugar, LaTeX proporciona un buen número de caracteres adicionales a menudo inexistentes en los teclados y que se pueden introducir como órdenes; por ejemplo, `\textdagger` inserta una cruz (†) en el punto donde aparece. Véamoslo con un ejemplo de documento completo:

```
\documentclass[spanish]{article}  
\usepackage{babel}  
\usepackage[T1]{fontenc}  
\usepackage{textcomp}  
\usepackage[utf8]{inputenc} % Puede depender del sistema o editor  
  
\title{Un título}  
\author{El autor}  
\date{5 de marzo del 2015}  
  
\begin{document}  
  
Un breve texto introductorio que servirá como  
ejemplo para mostrar qué forma tiene un párrafo.  
Hasta ahora solo tenemos uno, que concluimos con  
una línea en blanco.  
  
Tras la línea en blanco, tenemos otro párrafo. En  
él, además, escribiremos una cruz (\textdagger).  
  
Pero interrogaciones, comillas, etc., se escriben  
normalmente: ¿de verdad?, «comillas», ¡qué bien!  
  
Los estilos de letra también se introducen con  
órdenes, como \textit{cursiva} y \textbf{negrita}.  
  
\end{document}
```

El preámbulo

Ya se ha explicado cómo se compila un documento en **LaTeX**, pero no se ha hablado aún de cómo escribir el documento a compilar. En este capítulo se analiza la estructura básica de un documento, y en el siguiente capítulo se expondrán los conceptos básicos sobre la escritura de texto en **LaTeX**.

La estructura

La estructura de un documento en **LaTeX** se divide en dos grandes partes: el preámbulo y el cuerpo del texto. El siguiente ejemplo muestra un documento mínimo apropiado para el español:

```
\documentclass[spanish]{article}  
\usepackage{babel}  
\usepackage[T1]{fontenc}
```

```
\usepackage{textcomp}
\usepackage[utf8]{inputenc} % Puede depender del sistema o editor
\begin{document}
Aquí iría el texto del documento en sí.
\end{document}
```

Preámbulo

En el *preámbulo* se escriben las instrucciones fundamentales que indican a **L^AT_EX** qué clase de documento se va a escribir y qué características va a tener éste, así como también las que indican a **L^AT_EX** qué paquetes se deben cargar. El preámbulo siempre empezará con la instrucción:

```
\documentclass[<opciones>]{<plantilla_documento>}
```

Para definir la plantilla que se va a emplear en el documento, como por ejemplo `article` o `report`, que determinan diferentes estilos. En general, los argumentos que toma este comando son las llamadas *clases* de documento, y pueden aceptar diferentes opciones. Por ejemplo, la instrucción:

```
\documentclass[12pt, letterpaper]{book}
```

Declara que el documento es un libro, con el tamaño de letra configurado a 12 puntos y utilizando papel tamaño carta. En vez de `letterpaper`, se pueden usar otros tamaños de papel, como lo es `A4` (`a4paper`).

Paquetes

Se llama paquete a una extensión del sistema básico que añade nuevas funciones. Hay, literalmente, cientos de paquetes con muy diversas funciones: inserción de imágenes (`graphicx`), paquetes gráficos (`TikZ`), internacionalización (`babel`, `polyglossia`), color (`xcolor`), música, ajedrez, ediciones críticas, secuencias de aminoácidos, etc. Todos estos paquetes deberán ser declarados con:

```
\usepackage[<opciones>]{<paquete>}
```

Donde entre los corchetes estará el nombre del paquete a usar, por ejemplo:

```
\usepackage{amssymb}
```

Para cargar el paquete `amssymb`, que proporciona símbolos matemáticos de la American Mathematical Society. Si una clase de documento o paquete que queremos cargar ofrece opciones y nosotros no especificamos la que queremos, se cargarán las opciones por defecto.

Cuerpo

El cuerpo del documento consiste en prácticamente todo lo que aparecerá en nuestra compilación. Es aquí, pues, donde escribiremos el texto verdadero. Comienza con la instrucción

```
\begin{document}
```

y termina con:

```
\end{document}
```

Todo lo que se escriba con posterioridad a esta instrucción será ignorado por **L^AT_EX** y no se compilará.

Una vez que iniciemos el cuerpo del documento debemos escribir al final de todo lo escrito la instrucción de cierre `\end{document}` aunque no hayamos terminado todo el documento, pues de otra manera tendremos un error en el proceso de la compilación y no podremos ir viendo cómo van quedando nuestros avances.

Órdenes o macros

Hay ciertos caracteres que tienen una función especial, como por ejemplo, el símbolo `\`, por el que comienzan todas las instrucciones, o las llaves y corchetes, que contienen los datos y opciones de las instrucciones. Por ejemplo, en `\title{Un título}` hay una orden (en muchos casos también llamadas macros), que es `\title` y que ajusta el título del documento con el dato (o argumento) que sigue, que es `Un título` (en este punto no se añade realmente el título al documento, sino que LaTeX tan sólo lee el dato y lo guarda para cuando haga falta).

Otro detalle que hay que destacar es que las órdenes que no van seguidas de algún argumento descartan el o los espacios que le siguen (con la excepción de las órdenes que consisten en un símbolo, como `\#`, `\%` o `\$`). Aunque hay pocas órdenes de este tipo —la mayoría tienen algún dato o consisten en un símbolo— es muy importante tener esto presente por si se diera el caso. La solución en tales casos suele pasar por añadir un «dato» vacío, es decir, un par de llaves.

Finalmente, hay que señalar que algunas órdenes no van seguidas de uno o varios argumentos, sino que operan sobre el texto que le sigue hasta que termina el bloque actual delimitado por llaves. En ocasiones, incluso, hay dos variantes que funcionan de cada uno de estos modos. En caso simple es el siguiente:

```
\textbf{texto en negrita}
{\bfseries texto en negrita}
```

Estas dos formas son equivalentes, pero por lo general se prefiere el primer tipo.

Un tipo especial de orden es la que delimita un bloque del documento. Van siempre por pares:

```
\begin{...}
...
\end{...}
```

Una estructura así se llama *entorno* o *ambiente* y la más importante es justamente la que abarca el cuerpo del documento.

Principios básicos de la escritura

Todo bloque de texto separado del resto con líneas en blanco se considera un párrafo. No es el único caso en que LaTeX considera que hay un párrafo, pero sí es el más importante. En este caso, se lee el texto contenido en ese bloque y LaTeX lo procesa con objeto de encontrar las mejores divisiones de línea, los mejores guiones y el mejor espaciado posible para el párrafo. También se preocupa de encontrar el mejor punto para cambiar de página, así como de cuadrar el resultado en la página. Todo ello, naturalmente, sin necesidad de intervención directa de quien escribe.

El texto de cada párrafo se escribe de modo normal, con algunas salvedades importantes. En primer lugar, un espacio entre palabras vale lo mismo que dos, tres o cientos, siempre que no se deje una línea en blanco. De esta forma se evitan espaciados irregulares que en sistemas WYSIWYG aparecen en ocasiones al teclear por error dos espacios seguidos. En segundo lugar, LaTeX proporciona un buen número de caracteres adicionales a menudo inexistentes en los teclados y que se pueden introducir como órdenes; por ejemplo, `\textdagger` inserta una cruz (†) en el punto donde aparece. Vémoslo con un ejemplo de documento completo:

```
\documentclass[spanish]{article}
\usepackage{babel}
\usepackage[T1]{fontenc}
\usepackage{textcomp}
\usepackage[utf8]{inputenc} % Puede depender del sistema o editor

\title{Un título}
\author{El autor}
\date{5 de marzo del 2015}

\begin{document}

Un breve texto introductorio que servirá como
ejemplo para mostrar qué forma tiene un párrafo.
Hasta ahora solo tenemos uno, que concluimos con
una línea en blanco.

Tras la línea en blanco, tenemos otro párrafo. En
él, además, escribiremos una cruz (\textdagger).

Pero interrogaciones, comillas, etc., se escriben
normalmente: ¿de verdad?, «comillas», ¡qué bien!

Los estilos de letra también se introducen con
órdenes, como \textit{cursiva} y \textbf{negrita}.
```

```
\end{document}
```

Clases de documento

Como mencionábamos, existen diferentes estilos que podemos darle a nuestro documento, y la selección de uno u otro dependerá de qué es lo que necesitamos hacer. Si queremos escribir un documento corto, podemos utilizar la clase `article`. En términos generales, esta clase de documento nos permite dividir el documento en secciones, subsecciones, párrafos y subpárrafos.

A continuación una lista de algunas clases típicas de documento:

- article** Para artículos académicos y otros documentos cortos que no es necesario dividir en capítulos, sino que bastan las secciones y subsecciones y sus párrafos y subpárrafos.
- book** Para libros y otros documentos más largos que deben incluir capítulos, prólogo, apéndices o incluso partes.
- report** Para informes técnicos. Es similar a la clase `book`.
- memoir** Una clase todoterreno con un buen número de funciones adicionales integradas.
- beamer** Otra clase para presentaciones mediante diapositivas.

Las clases `book` y `report` son muy similares, y ambas sirven para documentos grandes, como lo son, naturalmente, los libros y los reportes, entre otros trabajos. Sin embargo, existen ligeras diferencias. Por ejemplo, la clase `book` hace que los capítulos empiecen siempre en una página impar, de modo que si un capítulo anterior termina en una página impar, la página (par) siguiente quedará en blanco y el capítulo nuevo comenzará después de ella. Esto, en cambio, no sucede con la clase `report`, así es que un capítulo simplemente empieza en una página nueva, sea par o impar. Por supuesto, estas opciones pueden ser fácilmente modificadas. Todas las clases de la lista anterior admiten opciones adicionales. Por ello, la sintaxis general para indicar una clase de documento es la siguiente:

```
\documentclass['opción 1, opción 2, ...']{'clase de documento'}
```

Las opciones que podemos dar son:

a4paper, letterpaper, ...	Con esta opción indicamos que el tamaño del papel debe de ser <code>a4paper</code> (tamaño a4), <code>letterpaper</code> (tamaño carta), ... Otras opciones que determinan distintos tamaños de página son: <ul style="list-style-type: none">▪ <code>a5paper</code> (210 mm × 148 mm)▪ <code>b5paper</code> (250 mm × 176 mm)▪ <code>legalpaper</code> (14 in × 8.5 in)▪ <code>executivepaper</code> (10.5 in × 7.25 in) El valor por defecto es <code>letterpaper</code> , de Estados Unidos y México. En los documentos de otros países puede ser necesaria la opción <code>a4paper</code> .
landscape	Apaisado. Pone la página de forma horizontal.
10pt, 11pt, 12pt	Definen el tamaño de la fuente principal del texto.
oneside, twoside	Indican si el documento debe estar adaptado a impresión por un sólo lado de la página o por ambos lados de ella.
titlepage, notitlepage	Determinan si el documento debe o no incluir una página de título, i.e. si va a incluir o no una portada.
openright, openany	<code>openright</code> obliga a los capítulos a iniciar siempre sólo en páginas impares, mientras que con la opción <code>openany</code> permitimos que los capítulos se inicien en cualquier página.
onecolumn, twocolumn	Definen si el documento se va a escribir en una sola columna o a doble columna.
fleqn	Esta opción hace que las ecuaciones queden alineadas por la izquierda en lugar de que sean centradas (como sucede por defecto).
leqno	Con esta opción hacemos que el número de las ecuaciones quede alineado por la izquierda en

	lugar de por la derecha (como sucede por defecto).
draft, final	La opción <code>draft</code> se usa si queremos que la compilación del documento se haga a modo de "borrador". Con <code>draft</code> haremos que las líneas que sean demasiado largas queden marcadas mediante cajas negras. La opción <code>final</code> producirá simplemente que el documento se compile de manera normal.

Cuando no especificamos opciones para una clase de documento, se cargan las opciones por defecto de la clase que estemos utilizando. Por ejemplo, si escribimos

```
\documentclass[letterpaper,10pt,twoside,onecolumn,final,openright]{book}
```

sería lo mismo que si escribiéramos simplemente

```
\documentclass{book}
```

pues la clase `book` tiene como opciones por defecto `letterpaper,10pt,twoside,onecolumn,final,openright`. Además, la clase `book` producirá automáticamente una página para el título del documento. Con la opción `notitlepage` haremos que esto no suceda así, de manera que el título del documento no quedará en una página aparte.

La clase `article` carga automáticamente las opciones `letterpaper,10pt,oneside,onecolumn,final`. Puesto que en la clase `article` no existen capítulos, las opciones `openright` y `openany` no están permitidas.

Las opciones por defecto de la clase `report` son `letterpaper,10pt,oneside,final,openany`.

Diseño del documento

Tanto los márgenes como el tamaño del papel se pueden cambiar a los valores que se deseen con el paquete `geometry`. Un ejemplo simple, que ajusta todos los márgenes a 1 cm en una hoja DIN A5, es:

```
\usepackage[a5paper,margin=1cm]{geometry}
```

Un paquete alternativo es `zwpagelayout`, con menos opciones, pero que ajusta internamente los parámetros necesarios en un PDF (y que tiene otras funciones como marcas de corte, por ejemplo).

Curriculum vitae

[Manual de LaTeX/La estructura de un documento en LaTeX/Preámbulo/Curriculum vitae](#)

Presentaciones

[Manual de LaTeX/La estructura de un documento en LaTeX/Preámbulo/Presentaciones](#)

Cartas

[Manual de LaTeX/La estructura de un documento en LaTeX/Preámbulo/Cartas](#)

Exámenes y apuntes

[Manual de LaTeX/La estructura de un documento en LaTeX/Preámbulo/Exámenes y apuntes](#)

Paquetes comunes

Además de las clases estándar de documento de **L^AT_EX** descritas en el apartado anterior, hay algunos paquetes que normalmente vienen incluidos en cualquier distribución de **L^AT_EX** (más específicamente, en distribuciones de **L^AT_EX 2_ε**). Algunos de ellos los describiremos más detalladamente en capítulos posteriores, y nos limitaremos a dar aquí una tabla que los compendie:

alltt	Este paquete provee el entorno <code>alltt</code> , muy similar al entorno <code>verbatim</code> salvo que <code>"</code> , <code>{</code> y <code>}</code> tienen
--------------	--

	su significado usual, por lo que pueden ser introducidos comandos.
doc	Este es un paquete básico para la escritura de documentación de programas de L^AT_EX .
exscale	Este paquete proporciona versiones escaladas de las fuentes matemáticas de extensión.
fontenc	Este paquete se usa para especificar la codificación de fuente que debe usar L^AT_EX .
graphpap	Este comando permite el uso del comando <code>\graphpaper</code> , que se usa para dibujar cuadrículados o mallas.
ifthen	Para comandos de la forma if... then... else...
inputenc	Este paquete se usa para especificar la codificación de caracteres para los documentos de entrada de L^AT_EX . Si vamos a escribir en español, conviene usar la instrucción <code>\usepackage[latin1]{inputenc}</code> ^[1] , que nos permitirá escribir con acentos en nuestro archivo de entrada, y con ello no tendremos que escribir cosas como <code>pr\'actico</code> para obtener "práctico" en nuestro documento compilado. Lo mismo sucede con la tilde que lleva la eñe.
latexsym	Puesto que en L^AT_EX 2_ε los caracteres símbolos ya no son cargados automáticamente, es necesario cargar el paquete <code>latexsym</code> para poder disponer de ellos.
makeidx	Este paquete proporciona comandos para la construcción de índices alfabético.
syntonly	Este paquete causará que el documento sea procesado sin producir ningún documento compilado de salida, sino que lo único que nos permitirá hacer es verificar que la sintaxis de cada comando es correcta.

No obstante, los paquetes de la tabla anterior cumplen propósitos muy específicos y es probable que el lector no tenga interés en todos ellos. Existen paquetes muy comunes que son de gran utilidad:

amsmath	Este paquete, realizado por la American Mathematical Society, proporciona comandos para la escritura de fórmulas matemáticas de mayor complejidad.
babel	Este paquete está hecho para soportar lenguajes diversos, entre ellos el español. Para indicar que nuestro documento se escribirá en español, y con ello que éste se adapte a dicho idioma, hemos de escribir <code>usepackage[spanish]{babel}</code> .
graphics	Con este paquete podrás incluir y transformar imágenes en tu documento, incluyendo las que hayas creado con otros programas.

Marcas de agua: `draftwatermarks`

El paquete `draftwatermarks`^[2] permite introducir en los documentos marcas de agua, importa automáticamente el paquete `color`. Para usarlo en un documento **L^AT_EX** basta con importar en el preámbulo el paquete:

```
\usepackage{draftwatermarks}
```

Modo de uso

Con sólo importar el paquete, se introducirá la marca de agua por defecto, consistente en el texto "DRAFT", en gris al 80% con un tamaño de fuente de 5 cm y un factor de escala de 1'2, con un ángulo de inclinación de 45°. Para modificar cualquiera de estas cinco propiedades se emplean las siguientes instrucciones:

- Para el texto: `\SetWatermarkText{NuevoTexto}`
- Para un tono de gris: `\SetWatermarkColor[gray]{0.5}`, con el rango [0,1].
- Para un color diferente: `\SetWatermarkColor[rgb]{1,0,0}`, con el rango [0,1] para cada color primario.
- Para el tamaño de la fuente: `\SetWatermarkFontSize{3cm}`
- Para el factor de escala: `\SetWatermarkScale{5}`
- Para el ángulo de inclinación: `\SetWatermarkAngle{30}`, en grados.

Referencias

1. En algunas distribuciones es mejor escribir:


```
\usepackage[utf8]{inputenc}
```

2. Documentación del paquete `draftwatermark` en CTAN (<http://mirrors.ctan.org/macros/latex/contrib/draftwatermark/draftwatermark.pdf>)

El cuerpo

Manual de LaTeX/La estructura de un documento en LaTeX/Cuerpo

Portada del documento

En todas las clases estándar de documento de **LaTeX** podemos introducir el título y autor del mismo, así como también la fecha. Para el título se usa la instrucción `\title{'título del documento'}` y para el nombre del autor se usa `\author{'nombre del autor'}`. La fecha se escribe con el comando `\date{'fecha'}`. Si omitimos esta orden (o escribimos `\date{\today}`) aparecerá la fecha en que se compila el documento. Todas estas instrucciones se escriben en el preámbulo, y para que aparezcan impresos en el documento compilado debemos escribir

```
\maketitle
```

inmediatamente después de iniciar el cuerpo del documento (i.e. justo después de `\begin{document}`).

Si usamos la clase `book` o la clase `report` el título aparecerá al principio y en una página aparte. En cambio, con la clase `article` el título aparecerá en la parte superior de la primera página del documento. Si queremos que en la clase `article` aparezca el título en una página aparte, debemos especificar la opción `titlepage`, que está desactivada por defecto.

El tamaño de letra del título es `\LARGE`, pero puede ser cambiada dentro de `\title{}`. Por ejemplo, con las líneas

```
\documentclass{book}
\title{\Huge Composición de textos con LaTeX}
\author{Los Wikiescritores}
\date{}
\begin{document}
\maketitle
\end{document}
```

obtenemos la página de título siguiente:

Composición de textos con
 \LaTeX 2_ε

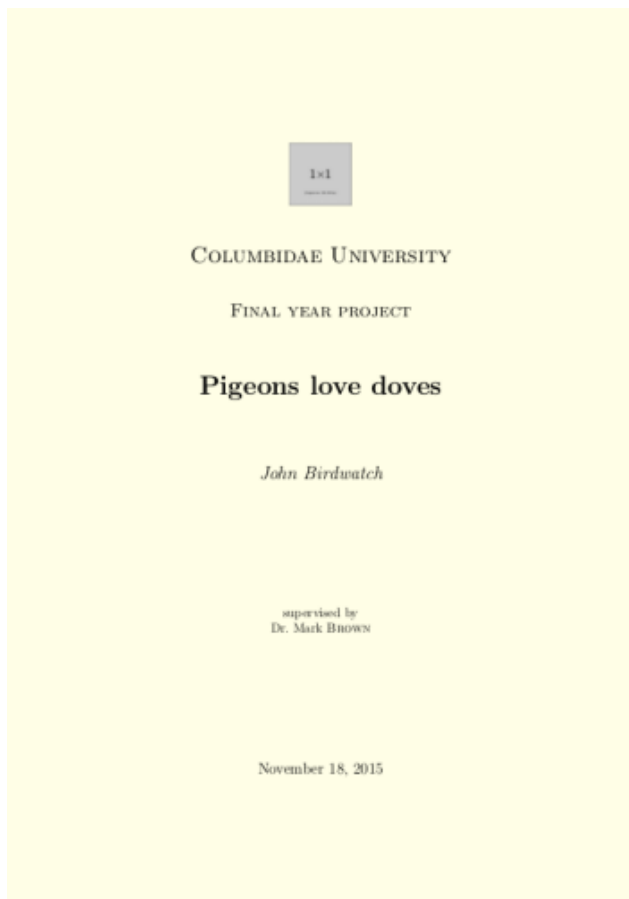
Los Wikiscribers

¿Qué hacemos si queremos una portada diferente?

Una opción es sustituir `\maketitle` por el entorno `titlepage`. Un ejemplo práctico:

```
\documentclass[12pt,a4paper]{report}
\usepackage{graphicx}
\begin{document}
\begin{titlepage}
  \centering
  \includegraphics[width=0.15\textwidth]{example-image-1x1}\par\vspace{1cm}
  {\scshape\LARGE Columbidæ University \par}
  \vspace{1cm}
  {\scshape\Large Final year project\par}
  \vspace{1.5cm}
  {\huge\bfseries Pigeons love doves\par}
  \vspace{2cm}
  {\Large\itshape John Birdwatch\par}
  \vfill
  supervised by\par
  Dr.~Mark \textsc{Brown}

  \vfill
% Bottom of the page
  {\large \today\par}
\end{titlepage}
\end{document}
```



Secciones de un documento

Aunque un documento en LaTeX puede tener una estructura por secciones casi arbitraria, normalmente se siguen las divisiones de las clases estándar. Con `article` puede dividirse en secciones, subsecciones, párrafos y subpárrafos. Con las clases `book` y `report` podemos incluir también capítulos.

Para iniciar un capítulo usamos el comando

```
\chapter{'nombre del capítulo'}
```

Similarmente se usan los comandos `\section{}`, `\subsection{}`, `\paragraph{}` y `\subparagraph{}` para secciones, subsecciones, etc. Cabe mencionar que la numeración de cada una de estas partes del documento la realiza **LaTeX** por sí solo.

Hay ocasiones en que el título de una sección es muy largo. En estos casos, el encabezado con el nombre de la sección sobrepasa el tamaño de la página, por lo que es conveniente contar con un método para que el nombre de la sección aparezca abreviado en el encabezado de la página. Por ello, la forma general del comando para secciones que provee **LaTeX** es el siguiente:

```
\section['nombre corto de la sección']{nombre de la sección}
```

En particular, la clase `book` permite dividir el documento en partes, cada una de ellas obtenidas con el comando `\part{}`. Además, esta clase de documento incluye nuevas características en **LaTeX 2_ε**, como lo son los comandos

```
\frontmatter
```

```
\mainmatter
```

```
\backmatter
```

que dan estructura al documento. Con `\frontmatter` damos el estilo que deben tener los principios, es decir, la portada, la tabla de contenidos, los prólogos..., con `\mainmatter` damos el estilo que debe tener el texto principal del documento, y finalmente `\backmatter` se usa para el estilo de las finales, es decir, parte final del libro (la bibliografía, los índices alfabético, colofón...).

Todo lo que quede contenido entre `\frontmatter` y `\mainmatter` (que se supone debe de ser la parte frontal del libro), tendrá un estilo en el que la numeración de página es con números romanos, y ningún capítulo, ni ningún otro título de nivel inferior, será numerado. Las páginas después de `\mainmatter` serán numeradas con números arábigos y los capítulos y títulos de nivel inferior sí serán numerados. Con `\backmatter` hacemos que los capítulos y títulos nivel inferior no aparezcan numerados (lo que es ideal para conclusiones o notas finales). Estos ajustes se pueden modificar por diversas vías.

Estilos de página

La numeración de páginas y la impresión de encabezados en las mismas constituyen el estilo de la página. Cambios en el estilo de página pueden realizarse con el comando

```
\pagestyle{'estilo'}
```

Los estilos que ofrecen las clases de documento estándar de **L^AT_EX** son `empty`, `plain` y `headings`. Con `\pagestyle{empty}` hacemos que las páginas queden sin número de página ni encabezado; con `\pagestyle{plain}`, que es el estilo por defecto, obtenemos páginas numeradas, pero sin encabezado; con `\pagestyle{headings}` obtenemos páginas numeradas y con encabezado. Más específicamente, `\pagestyle{headings}` produce efectos distintos según la clase de documento y las opciones que para ella se especifiquen. Por ejemplo, con la clase `article`, `\pagestyle{headings}` nos dará el número de página al pie y un encabezado con el nombre de la sección, y si hemos elegido la opción `twoside`, el encabezado será el nombre de la sección en las páginas pares y el nombre de la subsección en las páginas impares. Para el caso de la clase `book`, `\pagestyle{headings}` pondrá el número de página en la parte exterior de la cabecera (lado izquierdo en páginas pares y lado derecho en páginas impares) y el encabezado (que será el nombre del capítulo en páginas pares y el nombre de la sección en páginas impares) en la parte interior de la cabecera.

Si queremos cambiar el estilo de una página en particular, usamos

```
\thispagestyle{'estilo'}
```

que toma los mismos valores que `\pagestyle{}`.

Para especificar por nuestra propia cuenta que es lo que aparecerá en la cabecera, podemos usar la instrucción

```
\pagestyle{myheadings}
```

que pondrá los encabezados según estos estén indicados con los comandos

```
\markboth{'encabezado izquierdo'}{'encabezado derecho'}
```

y

```
\markright{'encabezado derecho'}
```

Notar que con la opción de clase `oneside`, los encabezados sólo pueden ser los derechos (pues no hay páginas que estén a la izquierda).

Al utilizar el estilo `\pagestyle{headings}`, vemos que la letra del encabezado aparece en caracteres inclinados y en mayúsculas. Esto se debe a que las clases de documento estándar de **L^AT_EX** así lo definen. Para reajustar los encabezados y los pies, en lo que

respecta a las mayúsculas y en otros detalles, hay dos paquetes útiles: fancyhdr y titleps. Un ejemplo simple de este último es:

```
\newpagestyle{main}{
  \sethead[\thepage][\chaptertitle][(\thesection) % pares
    {\thesection}]{\sectiontitle}{\thepage}} % impares
\pagestyle{main}
```

Índices

[Manual de LaTeX/La estructura de un documento en LaTeX/Cuerpo/Índices](#)

Glosario

[Manual de LaTeX/La estructura de un documento en LaTeX/Cuerpo/Glosario](#)

Finalmente este es el capítulo en el que hablaremos realmente de cómo escribir en **LaTeX**. En particular, estudiaremos la escritura de texto, i.e. la escritura en modo no matemático. La estructura de este capítulo esta ideada de tal modo que pueda servir también como referencia. Así, un lector con necesidades de consulta específicas podrá dirigirse exactamente al tema que le interesa, sin necesidad de conocer nada de lo que se ha expuesto anteriormente.

- [1. Espacios](#)
- [2. Caracteres especiales](#)
- [3. Tamaños, estilos y tipos de letra](#)
- [4. Alineación del texto](#)
- [5. Partición de palabras](#)
- [6. Signos ortográficos](#)
- [7. Portada del documento](#)
- [8. Capítulos y secciones](#)
- [9. Estilos de página](#)
- [10. Referencias](#)
- [11. Notas al pie](#)
- [12. Entornos de enumeración](#)
- [13. Citas](#)
- [14. Tablas](#)

En **LaTeX** las letras (o fuentes) tienen en general 5 atributos, aunque sólo mencionaremos cuatro de ellos, que son los que determinan el aspecto del carácter en el texto compilado.


Para los cambios breves de la fuente, lo recomendable es:

```
Este es un texto que puede tener \textit{cursiva} y también \textbf{negrita}.
También puede ser \textit{\textbf{cursiva con negrita}}. Otra posibilidad
es la \textsc{versalita}.
```

La conversión de caja se obtiene con `\MakeUppercase{texto}` (que convierte a mayúscula) y `\MakeLowercase{TEXTO}` (a minúscula). (No deben usarse en LaTeX las órdenes `\uppercase` y `\lowercase`, aunque las admita, porque no siempre dan el resultado correcto.)

Familia

La **familia** es el nombre de una colección de fuentes. **LaTeX** organiza las fuentes en tres familias, que son [Archivo:Roman.svg](#),

[Archivo:Sans Serif.svg](#), y . Para conseguir cada una de estas familias se usan, respectivamente, los comandos `\rmfamily`

(letras con remates), `\sffamily` (letras sin remates) y `\ttfamily` (letras mecanográficas). Estos comandos son en realidad *declaraciones*, por lo que su efecto se limita de manera distinta a la de los comandos comunes. Por ejemplo, si queremos conseguir

un texto con caracteres , debemos escribir

```
{\ttfamily ''texto''}
```

y así el efecto de `\ttfamily` afectará sólo al texto que se encuentre entre llaves.

Las fuentes preferminadas de LaTeX perteneces a la familia Computer Modern, pero podrían cambiarse a otras como, respectivamente, Times, Helveticas y Courier, por ejemplo. Para ello lo recomendado es cargar algún paquete, pero para emplear una fuente arbitraria instalada en el sistema es necesario recurrir a dos variantes de TeX llamadas XeTeX y LuaTeX. X

Serie

La **serie** de una fuente determina que tan gruesa o expandida será ésta. Con **LaTeX** tenemos la opción Medium (media) y la opción **Bold** (negrita). Caracteres con este tipo de series se consiguen, respectivamente, con las declaraciones `\mdseries` y `\bfseries`. Como éstas también son declaraciones, para obtener, por ejemplo, un texto en negritas hemos de escribir `{\bfseries ''texto''}`.

Lorem Ipsum
Lorem Ipsum
Lorem Ipsum
Lorem Ipsum
Lorem Ipsum

Muestras de la familia Computer Modern

Forma

La **forma** que puede tener un caracter dentro de una familia puede ser: `\Upright`.svg (vertical o recta), `\Italic`.svg (itálica), `\Slanted`.svg (inclinada) o `\Small Caps`.svg (Mayúsculas y mayúsculas pequeñas). Estas formas se consiguen con las declaraciones `\upshape`, `\itshape`, `\slshape` y `\scshape`, respectivamente. Además, tenemos los comandos

`\textbf{''texto''}`: para texto en negritas

`\textit{''texto''}`: para texto en itálicas

`\textsl{''texto''}`: para texto inclinado

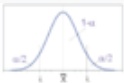
`\texttt{''texto''}`: para texto en estilo 

`\textsc{''texto''}`: para texto en mayúsculas y minúsculas pequeñas

Nota. Aunque LaTeX no dé error con las órdenes de Plain TeX `\bf`, `\it`, `\sf`, etc., no deberían usarse en lugar de las recién descritas.

Tamaño

El tamaño de una letra puede ser

<code>{\tiny tiny}</code>	que se consigue con la declaración <code>\tiny</code>
<code>{\scriptsize scriptsize}</code>	que se consigue con la declaración <code>\scriptsize</code>
<code>{\small small}</code>	que se consigue con la declaración <code>\small</code>
	que se consigue con la declaración <code>\normalsize</code>
	que se consigue con la declaración <code>\large</code>
<code>{\Large larger}</code>	que se consigue con la declaración <code>\Large</code>
<code>{\LARGE LARGE}</code>	que se consigue con la declaración <code>\LARGE</code>
<code>{\huge huge}</code>	que se consigue con la declaración <code>\huge</code>
<code>{\Huge Huge}</code>	que se consigue con la declaración <code>\Huge</code>

Estas órdenes no solo ajustan el tamaño de la letra, sino también la interlínea y en ocasiones también otros parámetros relacionados con listas y ecuaciones. Un error habitual es escribir un párrafo del siguiente modo:

```
{\small  
  Texto texto texto  
  texto texto.}
```

Aunque con ello se cambia el tamaño de la letra, la interlínea sigue igual. Es necesario señalar un párrafo con, por ejemplo:

```
{\small  
  Texto texto texto  
  texto texto.\par}
```

Para obtener los signos ortográficos del castellano (y otros más) es necesario usar el paquete `inputenc`. Si en el preámbulo escribimos

```
\usepackage[latin1]{inputenc}
```

cambiamos la codificación de la entrada de **L^AT_EX** a `latin1` (usada por ejemplo en Unix) y podremos escribir las palabras acentuadas tales y cuales, es decir, á, Á, ñ, Ñ nos dará á, Á, ñ, Ñ en el documento compilado. También se pueden escribir otros caracteres como «», “”, ¿, ¡, etc. Otros valores característicos son:

- En Windows:

```
\usepackage[cp1252]{inputenc}
```

- En Mac:

```
\usepackage[applemac]{inputenc}
```

Aunque cada vez es más frecuente el uso de editores de Unicode que guardan en UTF-8:

```
\usepackage[utf8]{inputenc}
```

Además, para que el resultado sea óptimo, es necesario establecer también una codificación apropiada de las fuentes con `fontenc`. Para el español, el valor recomendado es `T1`:

```
\usepackage[T1]{fontenc}
```

Cierto número de signos y símbolos requieren un paquete adicional:

```
\usepackage{textcomp}
```

No son raros los errores por un ajuste indebido de la codificación. Un mensaje como el siguiente suele indicar un error de este tipo:

```
Package inputenc Error: Unicode char \u8:a not set up for use with LaTeX
```

Un sistema más seguro para seleccionar la codificación, que da resultados correctos incluso si se recodifica un archivo, es el siguiente:

```
\documentclass{article}  
\usepackage{selinput}  
\SelectInputMappings{  
  aacute={á},  
  ntilde={ñ},  
  Euro={€}  
}  
\usepackage[T1]{fontenc}  
\begin{document}  
áéíóú  
\end{document}
```

Con XeTeX y LuaTeX no es necesario, en principio, ningún paquete, siempre que el documento fuente esté en UTF-8. Es posible que tu editor utilice por defecto la familia de fuentes Computer Modern (que tiene un conjunto de caracteres limitado y no apto para el español), por lo cual querrás usar otro conjunto de fuentes, como Latin Modern. Una forma fácil de hacerlo es agregar el paquete `fontspec` que por defecto define como fuente a Latin Modern:

```
\usepackage{fontspec}
```

En LaTeX también es posible un marcado lógico: así, `spanish` para `babel` proporciona las abreviaciones `<<` `>>` para comillas genéricas (similar al elemento `q` de HTML). El tipo de comilla («»“”’) es configurable (en España puede ser «» y en México “”, por ejemplo) y dependerá del contexto. Otra opción para las comillas es el paquete `csquotes`.

ASCII puro

LaTeX originalmente solo leía texto de código ASCII, por lo que para usar palabras acentuadas, se necesitaban algunas instrucciones. Desde hace muchos años (`inputenc` y `fontenc` datan de 1994) no es necesario y el empleo de los métodos descritos a continuación no resultan por lo general convenientes, pero se dan como referencia para trabajar con documentos antiguos y con ciertas tareas internas y especiales.

He aquí la forma de conseguir acentuaciones y signos ortográficos que no tienen un ASCII:

<code>\'</code>	´ (acento agudo)	<code>\`</code>	` (acento grave)
<code>\~</code>	~ (tilde)	<code>\"</code>	¨ (diéresis)
<code>\^</code>	^ (circunflejo)	<code>\c</code>	ç (c con cedilla)
<code>?`</code>	¿ (signo izquierdo de interrogación)	<code>!\`</code>	¡ (signo izquierdo de exclamación)

Así, para obtener palabras como conexión escribimos `conexi\'on`. La acentuación de la letra `i` requiere de un paso previo, que consiste en eliminar el puntito que ha de ser remplazado por el acento. Pare esto escribimos `\i`. Así, al escribir `\'{\i}` obtenemos `í`.

Para obtener comillas se usan el acento grave (```), que nos da `‘`, y el apóstrofo (`'`), que nos da `’`. Así, por ejemplo, con

```
Él dijo que había dicho la ``verdad''
```

obtenemos `Él dijo que había dicho la “verdad”`.

Hay símbolos que no están definidos de manera inmediata en **LaTeX**, pero que podemos conseguir muy fácilmente. Por ejemplo, el símbolo de grado, `°`, podemos obtenerlo con `^\{circ}`. Sería aún mejor si definimos nuestro propio comando para obtener el símbolo que hemos creado. Por ejemplo, podemos escribir (de preferencia en el preámbulo),

```
\newcommand{\grad}{^\{circ\}}
```

y así, al escribir `La temperatura era de 47 \grad C` obtendremos `La temperatura era de 47°C`

También podemos escribir `La temperatura era de 47^\{circ\}C` Obtendremos el mismo resultado.

En las últimas versiones del paquete `babel` es posible escribir ordinales con `1"o`, `2"a`, etc. para conseguir `1.º`, `2.º`, etc.

Referencias

<http://www.tex-tipografia.com/archive/spanish.pdf> En **LaTeX**, un espacio en blanco en el texto fuente produce un espacio en blanco en el documento compilado. Más de un espacio en blanco en el texto fuente no producen más que un espacio en blanco en el texto compilado.

Por tanto, si escribimos:

```
Uno o más espacios equivalen a un solo espacio en blanco
```

```
Uno o más  espacios equivalen a un  solo espacio  en blanco
```

obtenemos en ambos casos: `Uno o más espacios equivalen a un solo espacio en blanco`.

Podemos usar también el comando `\hspace{' 'valor' '}` para obtener un espacio horizontal igual al *valor* que especifiquemos. Por ejemplo,

```
Hola\hspace{4cm}adiós
```


dejará un espacio horizontal de 4 centímetros entre las palabras "Hola" y "adiós" en el texto compilado.

Espacio de no separación

L^AT_EX justifica de manera automática los párrafos, por lo que una vez que se llene una línea mandará lo que sigue a la línea de abajo, separando palabras que, en ocasiones, sería mejor mantener juntas. Para conseguir que **L^AT_EX** no separe dos palabras con un cambio de línea debemos usar el comando `\~`. Por ejemplo, debemos escribir

```
O~Wilde escribió obras como...
```

para que no haya un salto entre "O." y "Wilde".

Sangría de primera línea y espacio entre párrafos

Para ajustar la sangría de la primera línea, hay que modificar el parámetro `\parindent`. Por ejemplo, se deja un cuadratín con:

```
\setlength{\parindent}{1em}
```

Y se suprime con:

```
\setlength{\parindent}{0pt}
```

El espacio entre párrafos se ajusta con `\parskip`:

```
\setlength{\parskip}{10pt}
```

Espacios tras punto

A menos que se use babel con la opción spanish, de manera automática **L^AT_EX** deja un espacio adicional después de un punto, según la tradición tipográfica anglosajona,^[1] a menos que éste esté precedido por una mayúscula, caso en el cual **L^AT_EX** interpreta el punto como el de una abreviatura y no deja ningún espacio adicional. Si una abreviatura termina con una letra minúscula, como por ejemplo la abreviatura latina "e. g.", entonces hemos de evitar el espacio adicional que dejará **L^AT_EX**. Esto se consigue con el comando `\.`. Por ejemplo, debemos escribir

```
...existen clases (e.g.\ la clase de todos los conjuntos) que no son conjuntos
```

Si una frase termina con mayúscula, **L^AT_EX**, como ya hemos dicho, no dejará un espacio adicional después del punto que termina dicha frase por considerarlo el de una abreviatura. Para indicarle a **L^AT_EX** que se trata efectivamente del punto que termina una frase debemos escribir el comando `\@`. Por ejemplo, debemos escribir

```
Podemos compilar nuestros documentos en formato PDF\@. Además, están los formatos...
```

Espacios con extensión infinita

Otra opción más para espacios horizontales son los comandos que "empujan" el texto hasta el final de la página. Por ejemplo, el comando `\hfill` empuja el texto dejando espacios en blanco, como en el siguiente ejemplo: Desde este punto\hfill hasta este punto produce:

Desde este punto\hfill hasta este otro

Si en lugar de `\hfill` escribimos `\hrulefill` o `\dotfill` obtenemos, respectivamente:

Desde este punto\hrulefill hasta este otro

y

Desde este punto\dotfill hasta este otro

Blancos verticales

Por otra parte, una o más líneas en blanco en el texto fuente producen una sólo línea en blanco en el texto compilado.

Así, si escribimos:

Primera línea.

Aún estamos en la misma línea. Esta es la segunda línea Esta es la tercera línea

obtenemos:

Primera línea. Aún estamos en la misma línea.

Esta es la segunda línea

Esta es la tercera línea.

Podemos usar también el comando `\vspace{'valor'}`, con un efecto similar al de `\hspace{}` salvo que el espacio es vertical. Para espacios verticales predefinidos, podemos usar los comandos

`\smallskip` `\medskip` `\bigskip`

Es posible también dejar cierto blanco cuando se hace un salto de línea con `\\` añadiendo una dimensión entre paréntesis: `\\['valor']` produce un espacio entre líneas igual al *valor* especificado. Recuérdese que esta orden no es para crear espacios entre párrafos, sino solo cuando para los casos, no muy habituales, en los que hay un salto de línea dentro de un párrafo. De hecho, su uso para aumentar el espacio entre párrafos es un error (habitual).

Referencias

1. Sin embargo, el paquete `babel` en su opción castellana sigue la tradición tipográfica europea de no usar espacios extra después de un punto. Por tanto, ahorra todos estos problemas con las abreviaturas.

Al justificar el texto, **L^AT_EX** partirá las palabras que ya no quepan completas en una línea. Para ello, basta con cargar el paquete `babel` con la opción `spanish`:

```
\usepackage[spanish]{babel}
```

En algunas ocasiones, raras, no siempre lo hace del modo correcto (en especial en voces compuestas y prefijadas). Para indicarle a **L^AT_EX** la forma en que debe partir una palabra en particular usamos el comando `\hyphenation`. Por ejemplo, para indicar a **L^AT_EX** como debe partir las palabras "neoortodoxia" y "bioaerosol" escribimos:

```
\hyphenation{neo-or-to-do-xia bio-ae-ro-sol}
```

Si escribimos esto en el preámbulo del documento, las reglas de partición de esas palabras quedarán grabadas en las reglas de partición del lenguaje que estemos usando. Como mencionábamos, una especificación de partición para una palabra con el comando `\hyphenation` en el preámbulo hará que las reglas de partición que has dado para esa palabra se guarden en las reglas de partición del lenguaje "Spanish".

Otra forma de indicar cómo debe partirse una palabra es mediante el comando `\-`. Por ejemplo, si te das cuenta que en la compilación de tu documento la palabra "incomible" está mal particionada, ve hacia ella en el texto fuente e indica la forma correcta, reemplazando la palabra `incomible` por `in-co-mi-ble`. Pero es un último recurso que conviene evitar en general.

Para que **L^AT_EX** no parta una palabra en algún punto, se usa `\mbox{}`. Por ejemplo, si se escribe `Dr.\mbox{Knuth}`, **L^AT_EX** siempre escribirá `Dr. Knuth` sin partir esta última palabra (no debería haber espacios dentro de esta orden, pues no se ajustaría al tamaño apropiado cuando se justifica el texto). Aquí, `~` es un espacio de no división. Ya hemos visto que **L^AT_EX** reserva ciertos símbolos de código ASCII para funciones especiales (e.g. la barra invertida `\` para iniciar un comando, la tilde `~` para evitar particiones de palabras, etc). He aquí todos los caracteres que tienen una función especial en **L^AT_EX**:

`$` `#` `%` `&` `^` `_` `{` `}` `~` `\`

Si queremos que aparezcan como simples símbolos en el texto compilado, hemos de escribir

`\$` `\#` `\%` `\&` `\^` `_` `\{` `\}` `\~` `\textbackslash`

Algunos paquetes también interpretan de modo especial ciertos caracteres. El caso más significativo es babel, en el que las comillas rectas " introducen lo que se llaman shorthands o abreviaciones. Obsérvese que este carácter no debe usarse para introducir comillas en ningún caso, aunque no se haya cargado babel. Así, con spanish para babel "n equivale a ñ, "o es el ordinal, "+-- es una raya, etc. Manual de LaTeX/Escribiendo texto/Internacionalización **L^AT_EX** automáticamente justificado los párrafos (es decir, los alinea por ambos lados), aunque podemos alinear el texto solo por la izquierda con la declaración `\raggedright` (bandera por la derecha) y solo por la derecha con la declaración `\raggedleft` (bandera por la izquierda). Para centrar el texto hemos de usar la declaración `\centering`.

Alternativamente podemos usar los entornos de alineación `flushleft`, `flushright` y `center`. Puesto que estos son entornos, para, por ejemplo, centrar texto con `center` debemos escribir

```
\begin{center}
```

Texto centrado

```
\end{center} 12.1. LISTAS NO NUMERADAS
```

Este tipo de listas son simplemente un conjunto de elementos como el siguiente:

- leche
- pan y cereales
- legumbres

Una lista de este tipo se consigue con el entorno **itemize**, donde cada elemento a especificar irá precedido del comando `\item`, de la siguiente manera:

```
\begin{itemize}
\item leche
\item pan y cereales
\item legumbres
\end{itemize}
```

Este tipo de listas pueden anidarse. LATEX se encarga de la gestión de la apariencia de los distintos niveles de profundidad:

```
\begin{itemize}
\item leche
\item pan y cereales
```

- leche `\begin{itemize}`
- pan y cereales `\item trigo`

```
.trigo \begin{itemize}
  ° harina \item harina
\end{itemize}
.cebada \item cebada
.centeno \item centeno
.maiz \item maiz
\end{itemize}
```

- legumbres `\item legumbres`

```
.lentejas \begin{itemize}
\item lentejas
.garbanzos \item garbanzos
\end{itemize}
\end{itemize}
```

En los libros escolares, las tablas son normalmente utilizadas para recapitular los resultados de una investigación. En general es necesario manejarlas bien para realizar documentos de buena calidad.

La gestión de tablas no es muy intuitiva. Las tablas de base son fáciles y presentables, utilizando la misma lógica que en HTML, pero una tabla un poco más elaborada requiere de cierto aprendizaje ya que no es muy intuitiva su construcción.

El entorno *tabular*

Recordemos algunos conceptos ya explícitos.

Entorno

Un entorno es una declaración particular destinada a la composición de texto en un estilo específico. Todos los entornos empiezan y terminan de la misma manera:

```
\begin{nombre-entorno}  
...  
...  
\end{nombre-entorno}
```

Entorno `tabular`

El entorno `tabular` es otro tipo de entorno, concebido para colocar los datos en las tablas. Ciertos parámetros son necesarios después de la declaración del entorno para describir la alineación de cada columna. No es necesario indicar el número de columnas porque se deduce a partir de los parámetros introducidos. De la misma manera, se pueden introducir líneas verticales entre columnas. Los símbolos siguientes están disponibles para describir las columnas de una tabla.

- `l` : Columna alineada a la izquierda
- `c` : Columna centrada
- `r` : Columna alineada a la derecha
- `p{anchura}` : Columna de anchura fija, justificada y con sangría; El texto está posicionado en lo alto de la celda.
- `m{anchura}` : Como en el caso anterior pero el texto está centrado verticalmente.
- `b{anchura}` : Como en el caso anterior, pero el texto está posicionado en la parte baja de la celda.

<cite id="endnote_ los parámetros m y b necesitan la utilización de la extensión array" style="font-style: normal;">[[#ref_ los parámetros m y b necesitan la utilización de la extensión array|^]]

- `|` : línea vertical
- `||` : doble línea vertical

Una vez en el entorno,

- `&` : Separador de columna.
- `\\` : Principio de una nueva línea.
- `\hline` : Línea horizontal.

A tener en cuenta, que los espacios insertados entre estos comandos son inútiles, pero facilitan la lectura.

Tabla de base

Este ejemplo muestra como crear una simple tabla en LaTeX. Es una tabla tres por tres, pero sin ninguna línea.

```
\begin{tabular}{ l c r }  
1 & 2 & 3 \\  
4 & 5 & 6 \\  
7 & 8 & 9 \\  
\end{tabular}
```

1	2	3
4	5	6
7	8	9

Modificando el ejemplo anterior añadiendo algunas líneas verticales:

```
\begin{tabular}{ l | c || r | }  
1 & 2 & 3 \\  
4 & 5 & 6 \\  
7 & 8 & 9 \\  
\end{tabular}
```

1	2	3
4	5	6
7	8	9

Para añadir las líneas horizontales superiores e inferiores:

```

\begin{tabular}{1 | c | | r | }
\hline
1 & 2 & 3 & \\
4 & 5 & 6 & \\
7 & 8 & 9 & \\
\hline
\end{tabular}

```

1	2	3
4	5	6
7	8	9

Y finalmente, para añadir líneas centradas entre todas las filas (ver la utilización del entorno center):

```

\begin{center}
\begin{tabular}{1 | c | | r | }
\hline
1 & 2 & 3 & \\
4 & 5 & 6 & \\
7 & 8 & 9 & \\
\hline
\end{tabular}
\end{center}

```

1	2	3
4	5	6
7	8	9

Texto en las tablas

Los algoritmos de LaTeX para generar las tablas tienen ciertas imperfecciones. Una de ellas es que no hará un salto de línea dentro de una celda, aunque se desborde la anchura de la página. Para las columnas que contendrán una cierta cantidad de texto, se recomienda emplear el atributo `p` e indicar la anchura deseada de la columna (aunque esto pueda obligar a efectuar varios ajustes antes de obtener el resultado previsto).

Antes de continuar, tenemos que presentar el sistema de medidas que LaTeX emplea. Es muy flexible para que se pueda elegir entre toda una variedad de unidades de medida

- `pt` : punto anglosajón, 1/72 de pulgada ;
- `mm` : milímetro ;
- `cm` : centímetro ;
- `in` : pulgada (2,54 cm) ;
- `ex` : altura d'x , altura de una letra sin el trazo vertical ni el palo inferior de la fuente utilizada;
- `em` : cuadratín, grosso modo la anchura de una **M** (capital) en la fuente utilizada.

Existen comandos conocidos con el nombre de *commandos de longitud*, que juegan el rol de variable, que no tienen valores fijos porque dependen de la configuración de la clase y/o del preámbulo normal del documento. Los mas útiles son:

- `\parindent` : El tamaño del desplazamiento a la derecha ;
- `\baselineskip` : Distancia vertical entre las líneas ;
- `\parskip` : Espacio suplementario entre los párrafos ;
- `\textwidth` : La anchura de una línea de texto en el entorno local (por ejemplo, las líneas son generalmente mas estrechas en el resumen que en el texto normal);
- `\textheight` : La altura del texto en la página;

Los ejemplos que se dan a continuación son bastante largos debido a que se ilustran lo que se produce cuando hay un fragmento de texto en las celdas de una tabla. Así, en lugar de reproducirlo en la página, id a ([1] (<http://www.andy-roberts.net/misc/latex/latextutorial4.html>)) para poder consultar directamente el fichero LaTeX de ejemplo, [tutorial4/wrapped.tex wrapped.tex] y luego mirar el [tutorial4/wrapped.pdf resultado].

El entorno `tabular*`, control de la anchura de una tabla

Es fundamental una pequeña extensión de la versión básica de la tabla, ya que exige un parámetro suplementario (antes de las descripciones de columnas) para indicar la anchura deseada para la tabla.

```

\begin{tabular*}{0.75\textwidth}{| c | c | c | r | }
\hline
label 1 & label 2 & label 3 & label 4 & \\
\hline
item 1 & item 2 & item 3 & item 4 & \\
\hline
\end{tabular*}

```

```
\hline
\end{tabular*}
```

Sin embargo, esto no se parece a lo que se espera. Las columnas tienen siempre su anchura normal (justo lo suficientemente larga para adaptar su contenido mientras que las líneas son tan anchas como la anchura deseada de la tabla) La tabla no tiene una buena apariencia. La razón de este desorden es debido a que se tiene que insertar un espacio suplementario en la columna. Latex, tiene una longitud en caucho, que a diferencia de otras, no son fijas y Latex puede dinámicamente decidir el momento en el que deben ser fijas. Así, la solución al problema propuesto es:

```
\begin{tabular*}{0.75\textwidth}{@{\extracolsep{\fill}} | c | c | c | r | }
\hline
label 1 & label 2 & label 3 & label 4 & \\
\hline
item 1 & item 2 & item 3 & item 4 & \\
\hline
\end{tabular*}
```

En el código se ha introducido la construcción `@{...}` que se coloca al principio de la columna. Mas tarde se darán los detalles de este elemento. En el interior de estas construcciones, el comando `\extracolsep`, exige una anchura como parámetro. Se hubria podido utilizar una anchura fija, sin embargo, utilizando una longitud elástica, es decir, `\fill`, las columnas se espacian automáticamente de manera uniforme.

Los gráficos en LaTeX se puede generar internamente, con paquetes como [PSTricks](#) y [TikZ](#), o pueden [incluirse](#) archivos externos (PDF, JPEG, EPS).

Color

Para trabajar con colores la opción recomendada es el paquete ``xcolor``, que extiende las funciones básicas de ``color``. Con él es posible trabajar con CMYK y RGB entre otros espacios de colores. Con ``colorspace`` es posible también crear colores directos, es decir, que se imprimen con su propia tinta en lugar de con las tramas de cuatricromía.

Una vez cargado xcolor, se puede establecer el color del texto con

```
\textcolor{color}{texto}
```

donde 'color' es algún color previamente definido. Una sintaxis alternativa es:

```
{\color{color} texto}
```

Los colores predefinidos son:

```
white, black, red, green, blue, cyan, magenta, yellow.
```

Además, 'xcolor' proporciona opciones para cargar un buen número de colores predefinidos adicionales.

TikZ

[TikZ](#) (TikZ ist kein Ziechenprogram, TikZ no es un programa de dibujo), es un paquete desarrollado por Till Tantau, que se ejecuta en LaTeX, pero también en PlainTeX y en ConTeX. El modelo de LaTeX para la inserción de figuras en un documento se basa en el concepto de elemento flotante (que no tienen relación alguna con la función del mismo nombre de Word). En libros de texto es frecuente que las ilustraciones vayan numeradas y se coloquen bien a la cabeza o bien al pie. LaTeX coloca estas figuras de modo automático para que queden cerca del texto en el que se mencionan las figuras. No hace falta, por tanto, ir colocándolas a mano. Si el texto se redistribuyera, con la adición o la supresión de texto, las figuras se recolocarían a un nuevo sitio.

Hay dos elementos flotantes predefinidos: las figuras (con el entorno figure) y los cuadros o tablas (con el entorno table). [Manual de LaTeX/Inclusión de gráficos/Ubicación de un gráfico](#) [Manual de LaTeX/Inclusión de gráficos/Inclusión de gráficos EPS](#) [Manual de LaTeX/Inclusión de gráficos/Conversión de formatos gráficos](#) [Manual de LaTeX/Inclusión de gráficos/Dibujo de gráficos en LaTeX](#) [Manual de LaTeX/Inclusión de gráficos/El paquete XY-Pic](#) [PSTricks](#) es una colección de macros TEX basados en PostScript, soporta color, gráficas, movilidad, árboles y otros. Para llamarlo es necesario llamar al paquete antes de iniciar el documento:

```
\usepackage{pstricks}
```

No funciona con pdftex directamente, aunque hay utilidades que permiten ciertas conversiones automáticas.

Dentro del documento se pone el ambiente

```
\begin{pspicture}([Xmin],[Ymin])([Xmax],[Ymax])
...[COMANDOS]
\end{pspicture}
```

Se debe especificar el tamaño de lo que será la figura, por ejemplo desde la posición (-2,-2) hasta (2,2) es decir, de tamaño 4x4.

Creando Líneas

Para crear líneas tiene que estar dentro del ambiente pspicture:

```
\begin{pspicture}(-2,-2)(2,2)
\psline(0,0)(2,2)
\end{pspicture}
```

La forma general para crear una línea es:

```
\psline[OPCIONES]{TIPO DE FLECHA}(X0,Y0)(X1,Y1)
```

Comando	Descripción	Ejemplo
OPCIONES	Características de la línea: grosor, color, continuidad, etc.	<p>punteada:</p> <pre>{[linestyle=dashed]}</pre> <p>grosor:</p> <pre>{[linewidth=2pt]}</pre> <p>color:</p> <pre>{[linecolor=blue]}</pre>
TIPO DE FLECHA	Características de la flecha	<p>Simple</p> <pre>{-}</pre> <p>Bidireccional</p> <pre>{<->}</pre> <p>Direccional(predeterminado)</p> <pre>{->}</pre>
(X0,Y0)(X1,Y1)...	Puntos iniciales y finales	<p>Una línea</p> <pre>{(0,0)(2,2)}</pre> <p>Dos líneas</p> <pre>{(0,0)(2,2)(1,1)}</pre>

Ejemplo si se quiere crear una línea que tenga un sentido a hacia arriba, un color azul, grosor más ancho y punteada.

```
\begin{pspicture}(-2,-2)(2,2)
\psline[linewidth=2pt,linecolor=blue,linestyle=dotted]{->}(0,0)(2,2)
\end{pspicture}
```

 100px

Es posible crear también polígonos de la forma

```
\pscurve[OPCIONES]{TIPO DE FLECHA}(X0,Y0)(X1,Y1) ... (Xn,Yn)
```

[Manual de LaTeX/Inclusión de gráficos/Gráficos con MetaPost](#) [Manual de LaTeX/Inclusión de gráficos/Gráficos con xfig](#)

[Manual de LaTeX/Textos técnicos](#) [Manual de LaTeX/Textos técnicos/Lingüística](#) [Manual de LaTeX/Textos técnicos/Matemáticas](#)
Para insertar una fórmula en el texto escrito, debes usar encerrarla entre $\langle \rangle$. Esta es la forma canónica de LaTeX, aunque también es muy frecuente emplear $\$$, del siguiente modo: $\$fórmula\$,$ pero se considera más propia de otros formatos (como Plain TeX).

Si por el contrario deseas que la fórmula se muestre aparte del texto, y por lo tanto en un mayor tamaño, puedes conseguirlo usando el entorno `equation`, o encerrando la ecuación entre $\langle \rangle$ y \backslash ; ambos son equivalentes.

Veamos unos ejemplos:

Este es un texto mezclado con una ecuación $\frac{\sqrt{b-a^2}}{c}.$

Este es un texto mezclado con una ecuación $\frac{\sqrt{b-a^2}}{c}.$

$$D\Psi(u)[v] = p \int_{\Omega} |u|^{p-2} uv \, dx.$$

```
\begin{equation}
D\Psi(u)[v]=p\int_{\Omega}|u|^{p-2}uv\,dx.
\end{equation}
```

o:

```
\[
D\Psi(u)[v]=p\int_{\Omega}|u|^{p-2}uv\,dx.
\]
```

Una de las primeras cosas que nos gustará saber cómo especificar son las potencias (o superíndices) y los subíndices. La forma de hacerlo se detalla a continuación:

$$E = mc^2$$

```
\begin{displaymath}
E= m \, c^2
\end{displaymath}
```

También podremos especificar los subíndices mediante la siguiente expresión:

$$a = a_1 + a_2$$

```
\begin{displaymath}
a=a_1+a_2
\end{displaymath}
```

En caso de tener expresiones más complejas, es posible, encerrar dicha expresión entre llaves para definirla. Por ejemplo:

$$a = a_i + a_{i+1}$$


```
\begin{displaymath}
a=a_{i}+a_{i+1}
\end{displaymath}
```

Símbolos

Las matemáticas tienen muchos símbolos, por tanto una de las grandes dificultades que se tienen en LaTeX es intentar recordar el comando para cada uno de ellos.

Llenar este manual con todos ellos sobrecargaría la página, para eso dejo este link donde se pueden encontrar todos y cada uno de los símbolos matemáticos tales como letras griegas, operadores, flechas, delimitadores, etc.

- LaTeX maths symbols (<http://www.artofproblemsolving.com/Wiki/index.php/LaTeX:Symbols>)

Algunos ejemplos sobre las letras griegas se presentan a continuación:

```
\[
\alpha, \beta, \gamma, \delta, \epsilon, \zeta, \eta, \theta, \vartheta, \phi, \varphi, \omega, \sigma, \varsigma, \Gamma, \Delta, \Theta, \Phi, \Omega
\]
```

$\alpha, \beta, \gamma, \delta, \epsilon, \zeta, \eta, \theta, \vartheta, \phi, \varphi, \omega, \sigma, \varsigma, \Gamma, \Delta, \Theta, \Phi, \Omega$

Por tanto incluir símbolos matemáticos de diferentes maneras resulta ser muy sencillo, un ejemplo sería el siguiente:

```

\frac{\Gamma\mapsto\Pi}{\Psi\rightarrow\Upsilon}

```

$$\frac{\Gamma \mapsto \Pi}{\Psi \rightarrow \Upsilon}$$

Acentuación en modo matemático

¿Qué hacer cuando te quedas sin símbolos y fuentes? Bueno, el siguiente paso es recurrir a los acentos.

a'	\hat{a}	\bar{a}	\overline{aaa}	\check{a}	\vec{a}	\tilde{a}
\grave{a}	\acute{a}	\breve{a}	\check{a}	\vec{a}	\tilde{a}	
\dot{a}	\ddot{a}	\dddot{a}		\ddddot{a}		
\not{a}	\mathring{a}	\widehat{AAA}	\widetilde{AAA}			

Manual de LaTeX/Textos técnicos/Matemáticas/Texto entre matemáticas Manual de LaTeX/Textos técnicos/Matemáticas/Arreglos matemáticos Manual de LaTeX/Textos técnicos/Matemáticas/Entornos para teoremas Manual de LaTeX/Textos técnicos/Música Manual de LaTeX/Textos técnicos/Música/Lilypond El paquete (<http://www.ctan.org/pkg/musixtex>) MusiX_{TeX}, de Daniel Taupin, Ross Mitchel y Andreas Egler, permite representar partituras musicales en **L^AT_EX**. Es suficiente con incluir un único paquete en el preámbulo del documento:

```
\usepackage{musixtex}
```

Uso básico

El entorno básico para describir una partitura musical es `music`:

```
\begin{music}
\end{music}
```

Dentro de este entorno se pueden definir las características del pentagrama, así como la secuencia de notas y otros detalles.

Definiendo de la partitura

El paquete MusiX_{TeX} permite seleccionar cuatro tamaños diferentes para las piezas musicales, de menor a mayor:

- `\smallmusicsize`, aproximadamente el 80% del tamaño por defecto.
- `\normalmusicsize`, que es la opción por defecto.
- `\largemusicsize`, aproximadamente el 120% del tamaño por defecto.
- `\Largemusicsize`, aproximadamente el 150% del tamaño por defecto.

Escribiendo la partitura

El pentagrama se genera mediante los pares de comandos:

- `\startextract` y `\endextract`, para piezas breves.
- `\startpiece` y `\endpiece`, para piezas más largas.

El espaciado entre las notas musicales se indica mediante los comandos:

- `\znotes`: sin espaciado entre notas, para indicar acordes.
- `\notes`: espaciado recomendado para semicorcheas.
- `\notesp`: espaciado recomendado para semicorcheas con puntillo.
- `\Notes`: espaciado recomendado para corcheas.
- `\notesp`: espaciado recomendado para corcheas con puntillo.
- `\NOTes`: espaciado recomendado para negras.
- `\NOTesp`: espaciado recomendado para negras con puntillo.
- `\NOTes`: espaciado recomendado para blancas.
- `\NOTesp`: espaciado recomendado para blancas con puntillo.
- `\NOTEs`: espaciado recomendado para redondas.

Estos comandos se cierran con `\enotes` o `\en`, y entre ellos se escriben las notas, que siguen el formato `\<duración> <altura>`.

Para indicar la duración de las notas se emplean los comandos:

- `\wh`: redonda
- `\ha`: blanca (1/2)
- `\qa`: negra (1/4)
- `\ca`: corchea (1/8)
- `\cca`: semicorchea (1/16)
- `\ccca`: fusa (1/32)
- `\cccca`: semifusa (1/64)
- `\cccca`: garrapatea (1/128)

La `a` final indica a MusiX_{TeX} que escoja automáticamente la dirección de la nota, sustituyéndola por la `u` se fuerza a que la plica vaya hacia arriba, y por la `l` se fuerza a que la plica vaya hacia abajo.

Los silencios se indican mediante los comandos:

- `\pause`: silencio de redonda
- `\hpause`: silencio de blanca
- `\soupon` o `\qp`: silencio de negra
- `\ds`: silencio de corchea
- `\qs`: silencio de semicorchea
- `\hs`: silencio de fusa
- `\qqs`: silencio de semifusa

Para indicar la altura se emplea la notación anglosajona, de modo que para el do central se emplea la letra `c` y de forma correlativa el resto de notas, las letras mayúsculas se emplean para indicar notas más graves, tal y como se muestra en la tabla siguiente:

do	re	mi	fa	sol	la	si	
					A	B	
C	D	E	F	G	H	I	
J	K	L	M	N	a	b	
c	d	e	f	g	h	i	Octava central
j	k	l	m	n	o	p	
q	r	s	t	u	v	w	
x	y	z					

Además, se puede emplear el apóstrofe (') para elevar una octava todas las notas que se escriban a continuación, así como un acento grave (`) para bajar una octava. Se pueden emplear de forma acumulativa ambos signos.

Los dos siguientes ejemplos generan la misma partitura:

```
\notes \qa{CDEFGHI JKLMNab cdefghi jklmnop} \enotes
\notes \qa{``cdefghi 'cdefghi 'cdefghi 'cdefghi} \enotes
```

```
\begin{music}
\end{music}
```

Manual de LaTeX/Textos técnicos/Química El paquete (<http://www.ctan.org/pkg/xymtex>) ~~X~~^MTeX, de Shinsaku Fujita, permite representar estructuras químicas en 2D. Permite trabajar con tres modos de compatibilidad según el proceso de compilado del documento que se vaya a emplear, a saber:

- Compatibilidad TeX/~~L~~^M~~T~~_EX, mediante el paquete `xymtex`.
- Compatibilidad PostScript, mediante el paquete `.`
- Compatibilidad PDF, mediante el paquete `.`

Es necesario incluir tres paquetes en el preámbulo del documento, aunque el primero dependerá del modo de compatibilidad escogido:

```
\usepackage{xymtex}
\usepackage{xcolor}
\usepackage{graphicx}
```

Uso básico

- `yl-functions`
- Listas de substitución `<subslst>`.
- Listas atómicas `<atomlist>`.
- Listas ligadas `<bondlist>`.
- Listas ligadas esqueléticas `<skelbdlist>`.
- Listas ligadas eliminadas `<delbdlist>`.
- Fusionando unidades
- Reduciendo el tamaño

Compuestos alifáticos

- Compuestos planares
- Compuestos no planares
- Cadenas en zigzag

Carbociclos

- Carbociclos de 6 miembros
- Carbociclos de menos de 6 miembros
- Carbociclos fusionados
- Otros carbociclos

Heterociclos

- Heterociclos de 6 miembros
- Heterociclos de menos de 6 miembros
- Heterociclos fusionados
- Otros heterociclos

Técnicas avanzadas

- Polímeros
- Combinación de estructuras
- Estereoquímica
- Fórmulas moleculares
- Reacciones químicas
- Coloreado
- Generación de archivos EPS
- Generación de archivos PDF

Referencias

- Paquete XyMTeX en CTAN (<http://www.ctan.org/pkg/xymtex>)
- Manual de XyMTeX (<http://xymtex.com/fujitas3/xymtex/xym501/manual/xymtex-manualPS.pdf>)
- Web del creador (<http://xymtex.com/fujitas/fujitae.html>)

Manual de LaTeX/Gestionando la bibliografía [Manual de LaTeX/Gestionando la bibliografía/Referencias](#) La forma más sencilla de incluir notas a pie de página en **LaTeX** es usando:

```
\footnote{'nota al pie'}
```

Por ejemplo, si escribimos

```
... y, de hecho, el mal entendimiento respecto del carácter puramente existencial del axioma de elección
ha llevado a muchas discusiones estériles durante algunas décadas\footnote{cf. F. P. Ramsey. \textit{The
foundations of Mathematics}}, London Mathematical Society.}
```

En realidad, la instrucción general para poner notas al pie de página es la siguiente:

```
\footnote[marca]{'nota al pie'}
```

El valor opcional determina el tipo de marca de la nota al pie. Por ejemplo, si queremos que la marca de nuestra nota al pie sea 3, entonces escribimos `\footnote[3]{'nota al pie'}`.

Más aún, podemos cambiar el tipo de numeración de las notas al pie como sigue:

<code>\renewcommand{\thefootnote}{\arabic{footnote}}</code>	Numeración arábica: 1, 2, 3...
<code>\renewcommand{\thefootnote}{\roman{footnote}}</code>	Numeración romana en minúsculas: i, ii, iii...
<code>\renewcommand{\thefootnote}{\Roman{footnote}}</code>	Numeración romana en mayúsculas: I, II, III...
<code>\renewcommand{\thefootnote}{\alph{footnote}}</code>	Numeración alfabética en minúsculas a, b, c...
<code>\renewcommand{\thefootnote}{\Alph{footnote}}</code>	Numeración alfabética en mayúsculas: A, B, C...
<code>\renewcommand{\thefootnote}{\fnsymbol{footnote}}</code>	No números, sino símbolos diversos

El largo y el ancho de la línea de la nota al pie está determinada por la definición del comando `\footnoterule`. La definición original que da **L^AT_EX** de este comando equivale a la siguiente:

```
\newcommand{\footnoterule}{\vspace*{-3pt}
\noindent\rule{2in}{0.4pt}\vspace*{2.6pt}}
```

Así, si queremos que la línea de la nota al pie tenga un largo de 5cm y un ancho de 1pt, escribimos en el preámbulo

```
\renewcommand{\footnoterule}{\vspace*{-3pt}
\noindent\rule{5cm}{1pt}\vspace*{2.6pt}}
```

Con esta definición, el aspecto de la página del ejemplo anterior sería la siguiente:

Archivo:Footnote2.svg

Las notas puede adoptar multitud de disposiciones, y no solo puede ir al pie sino también al margen o a final del capítulo (o del libro). Hay diversos paquetes que extienden las funciones básicas:

bigfoot

Múltiples series de notas, en espacial para aparatos críticos

footmisc

Numeración por páginas, notas en un párrafo (sin salto de línea entre ellas)

yafoot

Más utilidades para notas

endnotes

Notas al final de capítulo

Para hacer las citas, sólo debemos escribir:

```
\begin{quote}
```

texto citado

```
\end{quote}
```

Recordemos que en ocasiones el tamaño de la fuente de la cita suele ser menor que el del texto normal, para esto sólo debemos:

```
\begin{quote}
```

```
\small texto citado
```

```
\end{quote}
```

Sobre programación en **L^AT_EX** [Manual de LaTeX/Programación/Macros](#) [Manual de LaTeX/Programación/Creación de paquetes](#) [Manual de LaTeX/Programación/Plain TeX](#)

Obtenido de «https://es.wikibooks.org/w/index.php?title=Manual_de_LaTeX/Texto_completo&oldid=215831»

Esta página se editó por última vez el 25 dic 2013 a las 16:51.

El texto está disponible bajo la [Licencia Creative Commons Atribución-CompartirIgual 3.0](#); pueden aplicarse términos adicionales. Véase [Términos de uso](#) para más detalles.