



UNIVERSIDADE DA CORUÑA
FACULTAD DE INFORMÁTICA
AULA DE FORMACIÓN INFORMÁTICA

*Introducción
a la edición de documentos
con $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$*

MANUAL DEL CURSO

Laura M. Castro Souto

Última revisión: *Julio de 2006*

Presentación del curso

Objetivos

\LaTeX es un sistema para la elaboración de documentos electrónicos de alta calidad. El principal objetivo de este curso es ilustrar los conceptos básicos y la manera de trabajar con \LaTeX . Partiendo desde cero, se pretende proporcionar la base suficiente para poder crear todo tipo de documentos, desde simples informes o cartas hasta artículos de investigación y memorias de proyectos.

Requisitos

Es necesario el dominio a nivel de usuario de algún sistema operativo (conocimientos de informática básica para manejo de archivos y programas), puesto que \LaTeX está disponible para los más comunes. No obstante, en el curso se trabajará bajo entorno Linux, de modo que se recomienda familiaridad con éste último.

Contenidos

Las líneas maestras que se seguirán responden al siguiente esquema:

- Introducción

- Conceptos básicos
- Creación de documentos
- Formato de documentos
- Edición elemental de documentos
- Edición especial de documentos
 - Edición matemática
 - Objetos flotantes: tablas y figuras
- Referencias internas
 - Índices
 - Bibliografía
- Personalización

Web del curso

En la siguiente web se irán actualizando diversos contenidos, como las transparencias que se utilizarán en clase, ejercicios propuestos y soluciones:

<http://www.madsgroup.org/staff/laura/latex.html>

Referencias

1. Bernardo Cascales Salinas et al.

El libro de \LaTeX .

Prentice Hall, 2004.

2. Javier Sanguino Botella.

Iniciación a \LaTeX 2 ϵ . Un sistema para preparar documentos.

Addison-Wesley, 1997.

3. Laura M. Castro Souto, Juan José Iglesias González.

Usando \LaTeX 1.97.

<http://latex.gpul.org/html/main.html>

4. Jane Hahn.

\LaTeX for everyone. A Reference Guide and Tutorial for Typesetting Documents Using a Computer.

Prentice Hall, 1993.

5. Bernice Sacks Lipkin.

\LaTeX for Linux.

Springer, 1999.

6. Leslie Lamport.

\LaTeX A Document Preparation System. User's Guide and Reference Manual.

Addison-Wesley, 1994.

Índice general

I	Manual	1
1.	Introducción	3
1.1.	¿Qué es \LaTeX ?	3
1.1.1.	¿Es \LaTeX un procesador de textos más?	4
1.1.2.	Diferencias entre <i>edición</i> y <i>composición</i> de textos	5
1.2.	¿Para qué y para quién puede ser útil?	6
1.3.	Un poco de historia...	7
1.4.	Cómo saber más	8
2.	Conceptos básicos	11
2.1.	¿Cómo funciona \LaTeX ?	12
2.1.1.	Invocando al genio de la lámpara	13
2.1.2.	Cuántos programas distintos... ¡para verte mejor!	15
2.1.2.1.	Especial para impresión: formato Postscript	15
2.1.2.2.	Popular en Internet: formato PDF	15
2.1.2.3.	De Postscript a PDF y viceversa	16
2.1.3.	Sistemas \TeX / \LaTeX para todos los gustos	17
2.2.	Estructura de un documento	17
2.3.	Indicaciones a \LaTeX	17
2.3.1.	Comandos, órdenes, variables y entornos	17

2.3.2. Nuestro primer intento	19
2.3.3. <i>Do you speak...?</i>	20
2.3.4. Caracteres reservados	21
2.3.5. Símbolos especiales	22
2.4. Herramientas para trabajar con \LaTeX	22
2.5. Ejercicios	26
3. Creación de documentos	27
3.1. Tipos de documentos \LaTeX	27
3.1.1. Opciones de los tipos de documentos	28
3.2. Estructuración de documentos extensos	30
4. Formato de documentos	33
4.1. Portadas automáticas de \LaTeX	33
4.2. División lógica de un documento	34
4.2.1. Índice	36
4.3. Encabezados y pies de página	36
5. Edición elemental de documentos	39
5.1. Entornos y bloques	40
5.2. Fuentes	41
5.2.1. Familias	41
5.2.2. Perfiles	42
5.2.3. Grososres	43
5.2.4. Tamaños	44
5.2.5. Otros efectos	47
5.3. Listas de elementos	48
5.3.1. Listas no numeradas	48
5.3.2. Listas numeradas	49
5.3.3. Listas descriptivas	50
5.4. Alineado de texto	51

5.5. Notas al pie y al margen	52
5.6. Citas textuales	53
5.7. Texto en columnas	53
5.8. Ejercicios	55
6. Edición especial de documentos	61
6.1. Edición matemática	62
6.1.1. Entornos	62
6.1.2. Paquetes	63
6.1.3. Fórmulas a diestro y siniestro	64
6.1.3.1. Superíndices y subíndices	64
6.1.3.2. Fracciones y binomios	65
6.1.3.3. Raíces	65
6.1.3.4. Integrales, derivadas, sumatorios, límites	66
6.1.3.5. Cuantificadores y otras funciones	67
6.1.3.6. Texto dentro del entorno matemático	67
6.1.3.7. Llaves y flechas	68
6.1.3.8. Matrices y determinantes	68
6.1.3.9. Puntos suspensivos y delimitadores	69
6.1.3.10. Símbolos y espacios	70
6.2. Objetos flotantes: tablas y figuras	73
6.2.1. ¿Qué es “flotar”?	73
6.2.2. Tablas	74
6.2.2.1. Tablas flotantes	75
6.2.3. Imágenes y gráficos	77
6.2.3.1. Figuras y gráficos flotantes	79
6.3. Cartas	80
6.4. Ejercicios	83
7. Referencias internas	87
7.1. Referencias básicas	87

7.2. Bibliografía	90
7.2.1. BibT _E X	91
7.3. Índice de materias	94
7.4. Ejercicios	96
8. Personalización	97
8.1. Crear una portada propia	98
8.2. Cambiar los encabezados de página	98
8.3. Márgenes, interlineado, saltos de página y espacios	99
8.3.1. Cambiando los márgenes	99
8.3.2. Cambiando el interlineado	99
8.3.3. Saltos de página	100
8.3.4. Tratamiento del espacio	100
8.4. Segmentación de palabras	102
8.5. Evitar la numeración de elementos	102
8.6. Listas personalizadas	103
8.7. Euro	103
8.8. Colores	103
8.9. Cajas	104
II Apéndices	105
A. Errores en L^AT_EX	107
A.1. No te olvides de cerrar	107
A.2. Cada cosa en su lugar	109
A.3. Cuidado con esas tablas	110
A.4. Ojo a lo que escribimos	111
A.5. Indicar siempre las medidas	113
A.6. Lo que no se puede hacer	114
A.7. Advertencias	115

B. Presentaciones con \LaTeX	117
B.1. Entorno <code>slide</code>	117
B.2. Una herramienta sencilla: <code>prosper</code>	118
B.3. Una herramienta potente: <code>beamer</code>	119
C. Glosarios en \LaTeX	121
C.1. Paquete <code>gloss</code>	121
D. \LaTeX y el hipertexto	125
D.1. <code>latex2html</code>	125
Bibliografía	127
Glosario	131
Índice alfabético	135

Índice de figuras

1.1. Funcionamiento de \LaTeX	4
2.1. Funcionamiento detallado de \LaTeX	13
2.2. Captura de pantalla del editor Kile	24
2.3. Captura de pantalla del editor \TeX nicCenter	25
2.4. Captura de pantalla del editor $\text{i}\text{\TeX}$ Mac	25
6.1. Imagen de ejemplo	79
6.2. Ejemplo de carta en \LaTeX	82
6.3. Un pingüino deformado	85
6.4. Un pingüino tumbado	85
7.1. Un pingüino deformado	96
7.2. Un pingüino tumbado	96
B.1. Ejemplo de transparencia hecha con Prosper.	119
B.2. Ejemplo de transparencia hecha con Beamer.	120

Índice de cuadros

3.1.	Diferencias entre las distintas clases de documentos \LaTeX	31
4.1.	Comandos de estructuración de documentos \LaTeX	35
4.2.	Estilos por defecto de los documentos \LaTeX	37
5.1.	Combinaciones posibles de estilos de letra en \LaTeX	45
5.2.	Proporción de tamaños según el tamaño base del documento . . .	46
6.1.	Letras griegas y algunos otros símbolos \LaTeX	71
6.2.	Tabla de prueba	75
6.3.	Ejemplo de carta en \LaTeX (código fuente)	81
6.4.	Una tabla completa	86
8.1.	Contenido por defecto de las cabeceras en estilo myheadings . . .	99

Parte I

Manual

Capítulo 1

Introducción

Índice general

1.1. ¿Qué es \LaTeX?	3
1.1.1. ¿Es \LaTeX un procesador de textos más?	4
1.1.2. Diferencias entre <i>edición</i> y <i>composición</i> de textos . . .	5
1.2. ¿Para qué y para quién puede ser útil?	6
1.3. Un poco de historia...	7
1.4. Cómo saber más	8

EN este primer capítulo, de carácter introductorio, intentaremos responder a las preguntas iniciales de aquéllos que se encuentran por primera vez ante la herramienta \LaTeX : ¿qué es? ¿para qué puede servirme? También echaremos un vistazo rápido a su historia, cómo surgió y cómo ha ido evolucionando, y por último, citaremos algunas fuentes donde acudir en busca de más información.

1.1. ¿Qué es \LaTeX ?

\LaTeX un sistema software para la elaboración de documentos electrónicos de alta calidad, que es especialmente potente en el tratamiento de textos matemáti-

cos. Actualmente, está considerado como la herramienta más versátil y adecuada para la preparación de documentos, informes e incluso libros de carácter científico y técnico, aunque su uso es cada vez mayor en las humanidades y en disciplinas económicas y administrativas.

1.1.1. ¿Es L^AT_EX un procesador de textos más?

No, nada más lejos de la realidad. Para empezar, L^AT_EX no es un *procesador de textos* en el sentido en el que se suele emplear este término en informática. Cuando hablamos de “procesadores de texto”, estamos acostumbrados a pensar en aplicaciones de edición de documentos cuya principal característica es poseer propiedades **WYSIWYG**. Las siglas WYSIWYG son el acrónimo en inglés de la frase “*What You See Is What You Get*”, que resume el hecho de que, al trabajar con ese tipo de programas, los cambios que vamos realizando en el texto se reflejan instantáneamente en la pantalla de nuestro ordenador.

En L^AT_EX la forma de trabajar es totalmente diferente. El usuario utiliza *cualquier editor de textos* para crear los *ficheros de entrada*, en los que además del texto que conformará el contenido del documento, se incluirán indicaciones sobre las propias características de éste. Posteriormente L^AT_EX tomará ese texto, junto con las indicaciones que lo acompañan, y producirá para nosotros el documento final, tal y como refleja la figura 1.1.



Figura 1.1: Funcionamiento de L^AT_EX

1.1.2. Diferencias entre *edición* y *composición* de textos

¿Cuál es la ventaja de usar L^AT_EX, entonces, si requiere aprender la forma de hacerle indicaciones, editar con otro programa, y no vemos el resultado a medida que tecleamos? Para comprender esto es necesario darnos cuenta de la diferencia que existe entre **editar** un texto y **componerlo**¹ (tipográficamente hablando). La labor de composición de un documento abarca un gran número de tareas, en su mayoría repetitivas y mecanizables, y que en el campo de la tipografía y la imprenta se rigen por unas estrictas normas que llevan usándose desde mucho antes de que se pensase en los ordenadores como herramientas de trabajo cotidiano. Entre estas tareas encontramos, por ejemplo, la numeración de páginas, la construcción de encabezados acordes al contenido de la página actual, la numeración de capítulos, secciones o figuras, la gestión de tablas contenidos, índices, notas a pie o al margen, y un amplio etcétera.

La gran ventaja de L^AT_EX es que se ocupa de todas estas cuestiones por nosotros. En cierto modo, podríamos compararlo con un secretario personal: cuando hemos de redactar un informe para un superior, enviar una carta formal, presentar una instancia o elaborar nuestras memorias, nuestro secretario sabrá en todo momento cuántas páginas hemos escrito, corregirá todos los lugares que sean necesarios si decidimos cambiar el nombre de una sección o intercambiar los capítulos 3 y 7, se ocupará de numerar las notas que le dictemos y las figuras que le mandemos incluir, así como de buscar el lugar más adecuado para ellas (al final de la página, mejor en la página siguiente porque en esta no queda espacio...). También sabrá en qué página estaba la tabla del resumen económico del año anterior si queremos hacer referencia a ella, y no tendremos que preocuparnos por buscar la referencia a cierto libro incluido en la bibliografía porque él lo recordará por nosotros. Y si se añaden más referencias y decidimos que quedan mejor ordenadas alfabéticamente en vez de por orden de aparición, es nuestro diligente *secretario* L^AT_EX quien se ocupará del asunto. ¿A que suena bien?

¹En la bibliografía en inglés, diferencia entre *text typesetting* y *text processing*.

Indudablemente, la carga que suponen estos pequeños detalles se nos hará más patente cuando nos hayamos librado de ella. Si estamos acostumbrados a encargarnos de todo nosotros mismos, enseguida notaremos las bondades de poder concentrarnos sólo en lo importante de un documento: su contenido. \LaTeX se encargará de su formato, produciendo para nosotros un resultado con apariencia profesional. Y si ésta es nuestra primera incursión en el mundo de la creación electrónica de textos, sin duda aprenderemos a apreciar las ventajas de esta gran herramienta.

Así pues, en los siguientes capítulos aprenderemos cómo utilizar \LaTeX en nuestro propio beneficio, para producir documentos de impecable presentación dedicando el mínimo esfuerzo a las cuestiones visuales.

Como curiosidad, el nombre del que será nuestro servicial asesor de ahora en adelante, deriva de la base griega $\tau\epsilon\chi$ (raíz de palabras como *tecnología*), que significa *arte*. En inglés suele pronunciarse */leiteg/*, con un sonido final similar al escocés *loch*. Sin embargo, dado que este sonido no existe realmente en inglés, se le llama con mucha frecuencia */leitek/*. En castellano, podemos usar las formas */lateg/* o */latek/* indistintamente, pero no */latex/*.

1.2. ¿Para qué y para quién puede ser útil?

El público principal de este manual pretende ser personas familiarizadas con los ordenadores que deseen obtener una primera pequeña visión de \LaTeX , o quizás refrescar unos conocimientos poco asentados. También se espera que, una vez terminado el curso, sirva a modo de referencia breve y rápida de consulta ocasional.

\LaTeX es una herramienta más que adecuada para estudiantes, profesores, científicos, matemáticos, físicos, ingenieros, economistas y autores, en general, de informes, manuales, artículos, cartas, memorias, tesis e incluso libros matemáticos, científicos o técnicos.

\LaTeX proporciona, tal y como veremos a lo largo de este manual, un trata-

miento sencillo y robusto de todo lo relativo a formulación matemática y científica, por lo que si nuestras necesidades nos llevan a tener que escribir textos con cierta cantidad de simbología de este tipo, \LaTeX es indudablemente nuestra mejor elección. No obstante, aunque ninguno de éstos sea nuestro principal campo de actuación, si queremos obtener presentaciones elegantes sin perder mucho tiempo en la composición, también lo es.

Sin ver inmediatamente reflejado lo que se teclea, viéndonos en la tesitura de tener que aprender y adoptar una forma de trabajar nueva, los inicios con \LaTeX pueden parecer un panorama poco alentador. ¿Merece la pena usar \LaTeX ? La respuesta, por supuesto, dependerá de cada usuario. Pero el proceso de adaptación es exactamente el mismo que se supera cuando se decide cambiar de aplicación, de lenguaje de programación o de sistema operativo. Para que el trabajo dé su fruto, será necesario un poco de esfuerzo. Y para que lo aprendido no caiga en saco roto, deberemos hacer lo mismo que cuando aprendemos un nuevo idioma: no abandonarlo. \LaTeX dista mucho de ser difícil, pero sobre todo al principio requiere paciencia y práctica.

En este curso, realizaremos un acercamiento progresivo a \LaTeX : inicialmente aprenderemos los conceptos básicos que nos permitirán desenvolvemos, para profundizar más adelante. El número de comandos e instrucciones que tendremos que manejar, con el fin de realizar indicaciones a \LaTeX , será directamente proporcional al nivel de sofisticación que deseemos para nuestros documentos finales. Nuestro objetivo es poner al lector en el buen camino, ayudarle a dar sus primeros pasos, y finalmente proveerle de un mapa que le ayude a llegar tan lejos como se proponga.

1.3. Un poco de historia...

\LaTeX fue creado en 1982 por Leslie Lamport para simplificar \TeX , un lenguaje de programación creado por Donald Ervin Knuth entre los años 1977 y 1978. En aquel momento, el profesor Knuth estaba escribiendo lo que sería su famoso

libro “*The Art of Computer Programming*”. Por suerte o por desgracia, la copia de prueba que recibió de su editorial tras la maquetación no le gustó en absoluto. Terriblemente disgustado, decidió elaborar su propio sistema de edición de textos, que siguiese lo más fielmente posible las normas tipográficas tradicionales. Así surgió T_EX, nombre que hace referencia tanto al lenguaje que creó, como a su intérprete o compilador. El problema era que T_EX contenía cerca de 300 órdenes básicas, lo que hacía su manejo complejo y no siempre cómodo. Estos fueron los motivos que impulsaron a Leslie Lamport a definir sobre T_EX una colección de comandos que simplificaban su uso, permitiendo centrarse en la estructura del texto en vez de en los comandos para dar formato. Ese pequeño conjunto de comandos se denominó L^AT_EX. Años más tarde, sucesivas revisiones dieron origen a L^AT_EX 2_ε, el último estándar, que incluía, entre otras cosas, comandos para la inclusión de gráficos y la utilización de color.

T_EX ha sido considerado por expertos en tipografía y edición como la aportación más importante a esta disciplina técnico-artística desde los tiempos de Guttemberg. Al estar disponible para prácticamente cualquier entorno de usuario (distribuciones Linux, MacOS, Windows. . .) su difusión ha sido muy amplia. Así, e indudablemente gracias también a su condición de herramienta *libre*, L^AT_EX se ha convertido prácticamente en una *lingua franca* del mundo científico.

1.4. Cómo saber más

El presente documento no es más que una introducción, por lo que es inevitable que falten muchas cosas. No obstante, la experiencia dice que una vez que se proporciona la ayuda suficiente como para clarear la opacidad inicial, cada usuario puede progresar en la dirección que más le interesa.

Son muchos cientos de miles los usuarios de L^AT_EX a lo largo y ancho del mundo. No importa el idioma que usen, la versión de su sistema operativo o el entorno en el que trabajen. Los documentos L^AT_EX que puedan crear e intercambiarse “funcionarán” siempre, y siempre tendrán la misma apariencia. Cada uno

de ellos podrá abrir sin problema los ficheros de entrada con su editor favorito y \LaTeX producirá siempre a partir de ellos un documento con la apariencia que su autor obtuvo la primera vez, con todo en su sitio, sin tablas o figuras descolocadas, tal y como desearíamos.

Además de las referencias indicadas en la presentación del curso, existen múltiples recursos en la red que pueden ser consultados para resolver dudas o simplemente satisfacer la curiosidad, como por ejemplo:

1. *GPUL-Latex*.

<http://latex.gpul.org>

2. *El sitio de \LaTeX en español*.

<http://www.cervantex.org>

3. *El FAQ de CervanTeX*.

<http://corbu.aq.upm.es/~agmartin/latex/>

[FAQ-CervanTeX/FAQ-CervanTeX.html](#)

4. *Una Descripción de $\text{\LaTeX} 2_{\epsilon}$* .

Tomás Bautista et. al.

<http://www.lsi.upc.es/~eipec/pdf/ldesc2e.pdf>

5. *The Not So Short Introduction to $\text{\LaTeX} 2_{\epsilon}$* .

<http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf>

6. *An Introduction to \LaTeX* .

<http://www.latex-project.org/intro.html>

7. *Getting Started with \TeX , \LaTeX , and Friends*.

<http://www.tug.org/begin.html>

Capítulo 2

Conceptos básicos

Índice general

2.1. ¿Cómo funciona \LaTeX?	12
2.1.1. Invocando al genio de la lámpara	13
2.1.2. Cuántos programas distintos... ¡para verte mejor!	15
2.1.3. Sistemas \TeX / \LaTeX para todos los gustos	17
2.2. Estructura de un documento	17
2.3. Indicaciones a \LaTeX	17
2.3.1. Comandos, órdenes, variables y entornos	17
2.3.2. Nuestro primer intento	19
2.3.3. <i>Do you speak...?</i>	20
2.3.4. Caracteres reservados	21
2.3.5. Símbolos especiales	22
2.4. Herramientas para trabajar con \LaTeX	22
2.5. Ejercicios	26

EN este capítulo conoceremos los fundamentos básicos de \LaTeX , la forma de trabajar con él y su esquema de funcionamiento. También mencionaremos

algunas herramientas que pueden sernos útiles en el proceso.

2.1. ¿Cómo funciona L^AT_EX?

Como ya comentábamos en el capítulo 1, L^AT_EX no es simplemente un “editor” de textos, pues realiza tareas de “maquetador”. El proceso de crear documentos en L^AT_EX consta de tres pasos principales:

1. **Edición del texto *fuentes***, en lo que denominamos *fichero de entrada*. Esto, como también hemos mencionado ya, puede hacerse utilizando nuestro editor de textos favorito, ya que prácticamente cualquiera de ellos proporcionará la posibilidad de guardar lo que tecleemos como *texto plano*, es decir, texto sin formato alguno. En los *ficheros de entrada* se incluyen, además del contenido del documento propiamente dicho, una serie de indicaciones, dadas siguiendo una sintaxis determinada, que proporcionan a L^AT_EX información que usará en el proceso de maquetación o composición del documento final.
Aunque no es obligatorio, es habitual que los ficheros de entrada tengan la extensión `.tex`. Además, la longitud del nombre del fichero sólo está restringida por el sistema operativo, al igual que la utilización de tildes y otros caracteres, aunque no es posible incluir espacios.
 2. **Compilación**. Una vez listo, el *código fuente*, como se suele denominar también a los ficheros de entrada, ha de ser procesado. Esta es la tarea que lleva a cabo el *compilador* L^AT_EX, analizando las indicaciones que se incluyen con el texto y ocupándose de todos los detalles relativos a la composición del documento final. Tal y como muestra el esquema de la figura 2.1, el resultado de la compilación produce, entre diferentes ficheros auxiliares, un fichero con extensión `.dvi`, que es una versión *ligera* del documento que nos permite comprobar los resultados del procesado.
 3. **Visualización o impresión**. Aunque la versión *DVI* que obtenemos tras
-

la compilación del documento tiene ya la apariencia final del mismo, no es una versión “completa” (entre otras cosas, por ejemplo, no empaqueta las imágenes incluidas). Es por ello que suele transformarse a otro formato, normalmente *Postscript* o *PDF*. Con cualquiera de ellos tendremos ya disponible la versión definitiva, perfectamente adecuada para ser no ya sólo visualizada, sino impresa o intercambiada a través de Internet.

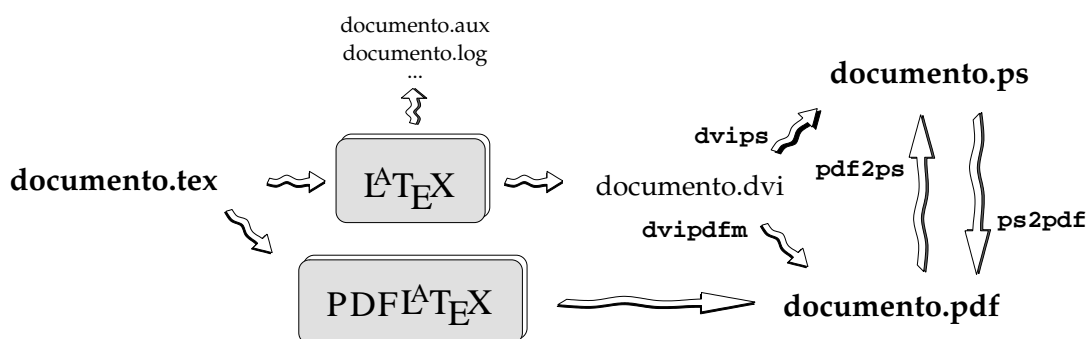


Figura 2.1: Funcionamiento detallado de \LaTeX

2.1.1. Invocando al genio de la lámpara

Hemos enumerado los tres pasos que hemos de seguir en la edición de textos con \LaTeX . Veamos ahora más concretamente las acciones que involucran las etapas de *compilación* y *transformación*. Una vez que conozcamos estas tareas, el resto del manual se centrará en la creación de documentos.

La manera de *compilar* un documento fuente \LaTeX es sencilla. Simplemente debemos invocar el comando `latex` pasándole como argumento el nombre del fichero fuente que queramos procesar:

```
latex documento.tex
```

Esto hará que el compilador \LaTeX procese el archivo `documento.tex`, generando, como ya hemos mencionado, diferentes archivos auxiliares y, si todo va

bien, también un `documento.dvi`. En caso de que se encuentre con algún tipo de error (fundamentalmente en la sintaxis o modo de utilización de las indicaciones incluidas en el propio documento fuente), el proceso se detendrá, indicándonos con diferentes mensajes lo que ocurre, e incluso, en ocasiones, el modo de abordarlo y solucionarlo (para más detalles, véase el apéndice A, dedicado a este tema).

Como sabemos, L^AT_EX se encarga de maquetar nuestro documento, llevando a cabo automáticamente todo un conjunto de tareas que involucran desde la numeración de páginas hasta el mantenimiento de referencias cruzadas y la gestión de índices de todo tipo (de materias, alfabéticos, de figuras, etc). Algunas de estas tareas requieren un doble procesamiento del documento:

1. En el *primer procesamiento* se recopila información, por ejemplo, de dónde se encuentran las figuras y el modo en que están etiquetadas, la página en que comienza cada capítulo y su título, los elementos que componen la bibliografía, etc.

La información obtenida en este primer paso se almacena en distintos ficheros auxiliares (como `documento.aux`, `documento.toc`, `documento.lof` o `documento.lot`), y la información sobre todo el proceso de compilación se guarda en el fichero `documento.log`. La presencia de estos ficheros y los datos que en ellos residen, informan y ayudan al compilador en posteriores ejecuciones.

2. En el *segundo procesamiento* se utiliza la información recopilada en el primero para dar valor a las referencias cruzadas, generar los índices completos, etc., completando de este modo la maquetación del documento.

Por este motivo, casi siempre necesitaremos llamar al compilador L^AT_EX al menos un par de veces.

2.1.2. Cuántos programas distintos tienes... ¡son para verte mejor!

Tal y como se refleja en la figura 2.1 de la página 13, son dos las opciones a la hora de transformar el documento en formato `dvi` que \LaTeX genera para obtener una versión definitiva del documento que estemos creando: escoger el formato **Postscript** o el formato **PDF**.

2.1.2.1. Especial para impresión: formato Postscript

Para transformar de formato *Device Independent* (`.dvi`) a formato *Postscript* (`.ps`) suele usarse fundamentalmente la herramienta `dvips`, que se utiliza de manera muy sencilla:

```
dvips [-o Informe.ps] documento.dvi
```

donde la opción `-o` nos permite cambiar el nombre del fichero Postscript resultante (que en este caso sería `Informe.ps` en lugar de `documento.ps`).

El formato Postscript presenta la ventaja fundamental de que muchas impresoras hoy en día son capaces de interpretarlo, sobre todo impresoras láser (como las disponibles en las AulasNet), lo que garantiza un acabado impecable sobre el papel. Para el resto de impresoras, la mayoría de los sistemas de impresión son capaces de convertir el formato Postscript al formato (*lenguaje*) nativo de la impresora, empleando utilidades como `ghostscript`/`gsview`/`ghostview` [7].

2.1.2.2. Popular en Internet: formato PDF

El formato *PDF* es un formato creado por Adobe Acrobat [1] que se ha hecho muy popular, sobre todo en Internet. El tamaño de un documento en formato PDF es considerablemente menor que su correspondiente versión en formato Postscript, y además permite algunas cosas que no están disponibles en otros formatos, como hiperenlaces dentro del propio texto.

Para transformar de formato DVI a formato PDF (`.pdf`) pueden usarse distintas herramientas, entre ellas `dvipdf` o `dvipdfm`. En general, se recomienda el uso de la segunda pues, ofreciendo la misma funcionalidad, convierte el formato DVI directamente a PDF, mientras que la primera emplea de manera combinada `ghostscript` y `dvips`. La manera de invocar cada una de ellas es:

```
dvipdf                documento.dvi [Informe.pdf]
dvipdfm [-o Informe.pdf] documento.dvi
```

El fichero de salida se llamará igual que el de entrada en los dos casos, aunque es posible indicar otro nombre alternativo (`Informe.pdf`) con carácter opcional.

Debido a la popularidad del formato PDF, ha surgido una herramienta de compilación alternativa a `latex`, denominada `pdflatex`, cuya salida es ya un fichero en formato PDF en lugar de en formato Device Independent. Las diferencias entre los compiladores `latex` y `pdflatex` son mínimas por lo que al ámbito de este curso y documento respecta, e irrelevantes en este momento. Serán comentadas más adelante y hasta entonces, consideraremos iguales ambas maneras de generar la versión PDF de nuestro documento (`latex+dvipdfm` vs. `pdflatex`).

2.1.2.3. De Postscript a PDF y viceversa

Acabamos de mencionar que la herramienta `dvipdf` hace uso de `dvips` para obtener finalmente el documento en formato PDF. Esto es posible porque se puede transformar un documento Postscript a formato PDF (y también a la inversa). Para ello están a nuestra disposición, respectivamente, las herramientas `ps2pdf` y `pdf2ps`:

```
ps2pdf documento.ps  [Informe.pdf]
pdf2ps documento.pdf [Informe.ps]
```

Ambas hacen uso de `ghostscript` y en los dos casos se puede [*opcionalmente*] indicar un nombre alternativo para el archivo generado.

2.1.3. Sistemas \TeX / \LaTeX para todos los gustos

Los programas que hemos visto hasta ahora están disponibles para cualquier distribución Linux, y se obtienen junto con la distribución de \TeX / \LaTeX más popular para este tipo de plataforma: $\text{\texttt{teTeX}}$ (salvo las herramientas $\text{\texttt{ps2pdf}}$ / $\text{\texttt{pdf2ps}}$, que suelen formar parte de un paquete denominado $\text{\texttt{psutils}}$).

Para otras plataformas existen sistemas equivalentes, como \TeX shop o $\text{\texttt{iTeXMac}}$ para MacOS X o $\text{\texttt{MikTeX}}$ para Windows.

2.2. Estructura de un documento

Ahora que ya sabemos cómo compilar un documento \LaTeX y cómo transformar la salida del compilador al formato que nos resulte más apropiado, es el momento de volver la vista a la estructura de los ficheros fuente.

Los ficheros fuente \LaTeX se dividen lógicamente en dos partes: **preámbulo** y **cuerpo**. Un fichero fuente \LaTeX siempre contendrá estas dos partes, y nunca puede prescindir de ninguna de ellas. El *preámbulo* es siempre la primera e incluye una serie de indicaciones globales sobre el documento. El *cuerpo* incluye el texto del documento, y posiblemente más indicaciones intercaladas con el mismo.

2.3. Indicaciones a \LaTeX

Prácticamente desde el inicio de este manual hemos estado mencionando que \LaTeX es susceptible de recibir (y en ocasiones espera) una serie de *indicaciones* sobre el documento a procesar. Veamos ahora qué forma tienen y cómo las reconoceremos en medio del resto del texto.

2.3.1. Comandos, órdenes, variables y entornos

Los **comandos** u **órdenes** \LaTeX comienzan siempre por una *barra inclinada a la izquierda* o *backslash* (\backslash) que va seguida del nombre del comando y, en caso

necesario, de una lista de atributos.

Los nombres de comandos son sensibles a mayúsculas y minúsculas, es decir, no es lo mismo `\orden` que `\Orden`. Por su parte, los atributos pueden ser opcionales, y aparecer entre corchetes y separados por comas, u obligatorios, y en tal caso se indican entre llaves.

A continuación se exponen varios ejemplos:

<code>\comando</code>	ejemplo de comando
<code>\Comando</code>	otro comando distinto
<code>\cmd{atributo}</code>	comando con atributo obligatorio
<code>\cmd[opción]</code>	comando con atributo opcional
<code>\cmd[opción,opción2=valor]{atributo}</code>	comando con varios atributos opcionales y uno obligatorio

En ocasiones, L^AT_EX pone a nuestra disposición **variables**, que representan valores del entorno de maquetación que podremos, bien utilizar como atributos u opciones para otros comandos, o bien modificar. Las variables siguen la misma convención que las órdenes L^AT_EX, son de la forma: `\variable`.

Por último, en L^AT_EX utilizaremos **entornos** para dar propiedades al texto, organizarlo, formatearlo y editarlo. Un entorno comienza con la indicación `\begin{nombreEntorno}` y termina con la indicación `\end{nombreEntorno}`:

```
\begin{entorno}
  El texto que se incluya dentro de este entorno
  tendrá unas características particulares
\end{entorno}
```

La mayoría de los entornos pueden incluirse unos dentro de otros (aunque hay excepciones), debiendo respetarse siempre el orden de apertura y cierre:

```
\begin{entorno1}
  El texto que se incluya dentro de este entorno
  tendrá unas características particulares
```

```
\begin{entorno2}
  Este otro texto puede sumar ambos conjuntos de propiedades
  o que se impongan las del entorno más anidado.
\end{entorno2}
Aquí volvemos a las propiedades anteriores, ordenadamente
\end{entorno1}
```

A lo largo del curso iremos aprendiendo los principales comandos y órdenes, algunas variables que nos podrán resultar de utilidad, y los entornos más habituales a la hora de trabajar con \LaTeX .

2.3.2. Nuestro primer intento

Después de tanta teoría, llega el momento de hacer la primera prueba. Nuestro primer documento \LaTeX será de lo más sencillo. Teclearemos:

```
\documentclass{article}
\begin{document}
  Este es mi primer documento \LaTeX.
\end{document}
```

La primera orden de todo documento \LaTeX debe ser la orden `\documentclass`, a la que es obligatorio indicarle el tipo de documento que queremos redactar. En este caso hemos especificado `article`, que es uno de los posibles tipos. Veremos más acerca de tipos de documentos en el próximo capítulo.

Todo lo que se incluye entre la orden `\documentclass` y el entorno `document` es lo que llamamos *preámbulo* del documento donde, como decíamos en la sección 2.2, se podrán incluir sólo comandos, que iremos descubriendo a medida que profundicemos en la materia. El texto del documento se teclea dentro del entorno `document`, que constituye el *cuerpo* del documento \LaTeX . Cualquier cosa que quede fuera de dicho entorno, después del `\end{document}` será ignorada por el compilador.

2.3.3. *Do you speak...?*

¿Demasiado trivial este primer ejemplo? Bien, ampliémoslo un poco:

```
\documentclass{article}
\begin{document}
  Esta será nuestra segunda incursión con \LaTeX{},
  tampoco nada demasiado arriesgado en realidad.
\end{document}
```

No parece un gran avance con respecto al anterior, pero si comprobamos la salida generada por L^AT_EX, notaremos enseguida que no importa que nuestro texto ocupe dos líneas en el fichero fuente: L^AT_EX se encarga de la maquetación y, por defecto, justifica nuestro texto. Nuestro secretario ya ha comenzado su labor. Sin embargo, también detectaremos varias anomalías: los caracteres acentuados no aparecen y la palabra *arriesgado* está fragmentada, aunque no por el lugar adecuado, ¿qué está pasando? ¿es L^AT_EX un asesor incompetente?

L^AT_EX es una herramienta con soporte para múltiples idiomas, pero por defecto asume que el texto se escribirá en inglés. Es por eso que los caracteres acentuados han de ser tratados de manera especial y las reglas de división de palabras son las anglosajonas. L^AT_EX no es un secretario incompetente, sigue sus reglas por defecto al pie de la letra. Poner remedio a este “desaguisado” es tan sencillo como decirle que modifique sus asunciones incluyendo en el preámbulo las siguientes órdenes:

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

La orden `\usepackage` se usa para indicar al compilador que utilice el paquete que se nombra entre llaves (argumento obligatorio). Dependiendo del paquete, pueden indicarse además opciones (como `spanish` en el caso del paquete `babel`, o `latin1` en el caso del paquete `inputenc`). Los **paquetes** son generalmente módulos que forman parte del sistema T_EX/L^AT_EX, pero que el compilador no

utiliza por defecto, y ésta es la manera de indicarle que emplee la información adicional que en ellos se incluye en la maquetación del documento actual. En este caso concreto, el paquete `babel` tiene información de maquetación relativa al idioma del documento, e indicándole la opción `spanish` conseguiremos no sólo que las reglas de división de palabras que utilice sean las propias del español, sino que las etiquetas de los capítulos o las imágenes sean *Capítulo* o *Figura* en lugar de *Chapter* o *Figure*. Por su parte, el paquete `inputenc` proporciona información a L^AT_EX sobre la codificación usada en el fichero fuente, y la opción `latin1` hará que el compilador no considere caracteres extraños las vocales acentuadas o la letra ñ, por ejemplo. Por supuesto, existen multitud de opciones para el paquete `babel`, correspondientes a infinidad de idiomas distintos, entre ellos, el `galician`.

No obstante, lo anterior no quiere decir que no se puedan emplear caracteres acentuados en idiomas que normalmente no los usan, o que L^AT_EX no se vaya a confundir nunca a la hora de segmentar una palabra y no tengamos manera de corregirle. Las tildes pueden indicarse utilizando una barra inclinada a la izquierda (\) seguida de una comilla simple y la vocal que queremos acentuar: `as\'i`. En cuanto a la segmentación silábica, volveremos a tratar este tema en el capítulo 8.

2.3.4. Caracteres reservados

Como podemos intuir a estas alturas, existen una serie de caracteres cuyo significado es especial para el compilador L^AT_EX. Uno de ellos es precisamente la barra inclinada a la izquierda (\), que hemos visto que puede indicar no sólo el comienzo de un comando, orden o variable, sino la presencia a continuación de algún tipo de secuencia especial (como en el caso de los caracteres acentuados).

Otros **caracteres reservados** son:

{ } [] # & % ~ _ ^ \$

Todos ellos se *escapan* (es decir, se “obtienen” cuando los queremos entre el texto como caracteres “normales”) de la misma manera: anteponiéndoles una \.

Las llaves y los corchetes¹ ya hemos visto que se utilizan para indicar opciones y parámetros. El tanto por ciento (%) es el símbolo de *comentario*: cualquier cosa que le siga hasta el final de la línea en que se encuentra será ignorada por el compilador. En cuanto al resto de caracteres reservados, nos los iremos encontrando a lo largo del curso y veremos para qué son utilizados y qué los convierte en caracteres especiales.

2.3.5. Símbolos especiales

Algunos símbolos no reciben el tratamiento de reservados, pero sí se comportan de manera especial. Es el caso de las comillas, los guiones y los puntos suspensivos. La forma de obtener las distintas variaciones de comillas (simples, dobles, latinas² e inglesas), así como los distintos tipos de guiones se indican en la tabla siguiente:

Comillas	<i>Simples</i>	Inglesas	‘ ’	‘ <i>hola</i> ’
	<i>Dobles</i>	Latinas	<< >>	« <i>hola</i> »
		Inglesas	‘ ‘ ’ ’	“ <i>hola</i> ”
Guiones	<i>Simples</i>		–	- <i>hola</i> -
	<i>Dobles</i>		--	– <i>hola</i> –
	<i>Triples</i>		---	— <i>hola</i> —

Por su parte, la manera correcta de obtener puntos suspensivos es con el comando `\dots`.

2.4. Herramientas para trabajar con L^AT_EX

Apenas acabamos de despegar y ya conocemos unos cuantos comandos de uso obligatorio y algunos otros que seguramente nos serán útiles. Hemos visto que tendremos que manejar diferentes herramientas a lo largo del proceso de creación de un documento. ¿Cómo organizarnos?

¹Dependiendo de la situación, puede no ser necesario escapar los corchetes.

²También llamadas francesas o españolas.

Afortunadamente, mientras el usuario no coge la soltura suficiente con \LaTeX como para decidir por sí mismo cómo le resulta más cómodo trabajar con él, existen diferentes aplicaciones que integran todas las herramientas que hemos mencionado: desde la compilación con \LaTeX o \PDFLaTeX hasta la conversión a formatos Postscript y/o PDF, incluyendo la visualización en pantalla usando visores especiales para ambos formatos, Postscript –como `gv`– o PDF –como `acroread`–.

Algunos de estos programas son:

Kile para Linux.

T_EXnicCenter para Windows.

Para MacOS X los sistemas ya mencionados T_EXshop o iT_EXMac (página 17) ya integran un editor de este tipo, con múltiples menús donde las opciones L^AT_EX más comunes están a disposición del usuario novel, que de esta manera se ve aliviado en la tarea de aprenderse el nombre de varias decenas de comandos antes de desenvolverse bien en este nuevo entorno. Además, también proporcionan accesos rápidos a las propias tareas de compilación, transformación y visualización en forma de botones en barras de herramientas totalmente configurables.

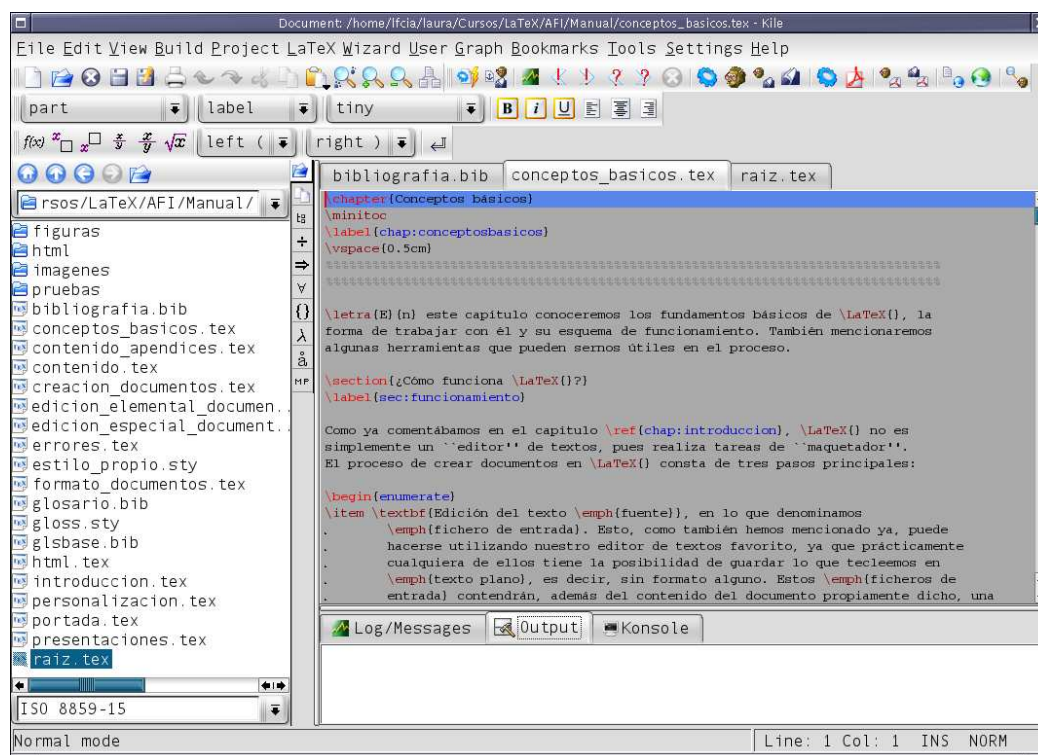


Figura 2.2: Captura de pantalla del editor Kile



2.5. Ejercicios

Ejercicio 2.1 Editar, con cualquier editor de textos (por ejemplo, `kile`), un fichero que contenga el código de ejemplo visto en la página 19. Compilarlo y observar la salida que se obtiene. Explorar las distintas posibilidades de visualización.

Ejercicio 2.2 Modificar el ejemplo anterior para que se ajuste al visto en la sección 2.3.3. Compilar y observar el resultado. ¿Qué anomalías se aprecian?

Ejercicio 2.3 Añadir al ejemplo modificado los paquetes `inputenc` y `babel`, con las opciones pertinentes para que su visualización sea la esperada.

Capítulo 3

Creación de documentos

Índice general

3.1. Tipos de documentos \LaTeX	27
3.1.1. Opciones de los tipos de documentos	28
3.2. Estructuración de documentos extensos	30

AHORA que conocemos cuál es la filosofía de \LaTeX y hemos dado nuestros primeros tímidos pasos, nos pondremos un poco más serios. En las próximas páginas veremos cómo iniciar la construcción de un documento en base a las características a las que responderá y cómo abordar su creación de manera genérica y lo más cómoda posible.

3.1. Tipos de documentos \LaTeX

Como veíamos en nuestro primer documento en el capítulo anterior, la primera orden de todo documento \LaTeX es el comando `\documentclass`, cuyo argumento obligatorio es una palabra que identificará el *tipo de documento* que queremos crear. En nuestro ejemplo indicábamos `article`, que es uno de los dos tipos fundamentales de documentos que \LaTeX reconoce, junto con `book`. Además de

estas dos clases base, disponemos de los tipos `proc` y `report`, que derivan de las anteriores.

Las clases `article` y `proc` están pensadas para trabajos cortos (entre 10 y 20 páginas, por ejemplo): informes, memorias, artículos o similares. Las clases `book` y `report`, por su parte, suelen utilizarse para libros o documentos de gran extensión: narraciones, relatos, amplios informes o memorias detalladas, proyectos docentes, apuntes de asignaturas, tesis,...

Al margen de estas cuatro opciones principales, existen las clases `letter` y `slides`, tipos especiales de documento que comentaremos en el capítulo 8 y el apéndice B, respectivamente.

3.1.1. Opciones de los tipos de documentos

En la sección 2.3 vimos que los comandos L^AT_EX pueden ser susceptibles de adaptar su comportamiento según nuestras indicaciones, gracias a las *opciones* que podemos suministrarles. La orden `\documentclass` acepta la siguiente serie de opciones:

Tamaño de letra — Todas las clases de documentos L^AT_EX establecen por defecto el tamaño *base* de la letra al valor 10pt. Decimos que es un tamaño *base* porque el tamaño de cosas como los títulos de las secciones, de las notas a pie de página, etc. se calcula automáticamente con relación a éste, para que se mantengan las proporciones a lo largo del documento. Si queremos indicar otro valor para el tamaño base de la letra del documento lo haremos del siguiente modo:

```
\documentclass[12pt]{article}
```

No obstante, por motivos relacionados con las normas de maquetación que L^AT_EX sigue fielmente, no es posible especificar cualquier valor para el tamaño base de la letra del documento. L^AT_EX sólo admitirá los valores 10pt (por defecto), 11pt o 12pt. Si indicamos cualquier otro, nos advertirá:

LaTeX Warning: Unused global option(s):
[13pt].

y utilizará el valor por defecto. Esto no quiere decir que no podamos tener tamaños de letra más grandes o más pequeños en nuestro documento, veremos cómo variar el tamaño de letra para casos puntuales en el capítulo *Edición elemental de documentos*.

Tamaño de papel — El formato de papel que se asume por defecto en todos los casos es `letterpaper`. Otras posibilidades son `legalpaper`, `executivepaper`, `a4paper`, `a5paper` y `b5paper` (aunque estas dos últimas no son válidas en el caso de documentos de clase `proc`).

Cuando se indica más de una opción para un comando, se separan con comas:

```
\documentclass[12pt,a4paper]{article}
```

Maquetación a una/doble cara — Podemos elegir si queremos que la maquetación del documento se haga pensando en una impresión a una (opción `oneside`) o a doble cara (opción `twoside`). Hay que tener muy presente que esto no quiere decir que el documento se vaya a imprimir a una/doble cara si no ajustamos así tanto la impresora como el programa de impresión que utilicemos en su momento, sólo quiere decir que L^AT_EX lo tendrá en cuenta a la hora de distinguir entre páginas *pares* e *impares* y colocar diferentes encabezados y ajustar apropiadamente los márgenes. La opción `oneside` es la opción por defecto en documentos `article`, `proc` y `report`, mientras que `twoside` lo es para los de clase `book`.

Cuando se activa la opción `twoside`, cobra relevancia la presencia de las opciones `openright/openany`, que especifican en qué página queremos que comiencen los capítulos en que se dividirá el documento. La opción por defecto para el tipo `book` es `openright`, lo que quiere decir que los capítulos empezarán siempre en una página impar (dejándose una página en blanco en caso necesario), salvo que se indique lo contrario.

Maquetación en columnas – Los documentos de tipo `proc` se maquetan en formato de dos columnas. Para conseguir el mismo efecto en documentos de las otras clases, en los que por defecto se asume `onecolumn`, disponemos de la opción `twocolumn`.

Maquetación de la portada – Aunque no lo hemos visto aún (lo haremos en el capítulo siguiente), \LaTeX dispone de algunos comandos referidos a la confección de portadas para el documento. Por defecto, la portada es una página a parte (`titlepage`) en las clases `book` y `report`, pero no así en `article` y `proc` (`notitlepage`).

La tabla 3.1 es un resumen de las diferencias entre los valores por defecto activos en cada tipo de documento.

Existe una última opción, `draft`, que suele utilizarse para hacer más rápido el proceso de compilación durante la construcción de un documento. Utilizando la opción `draft` (opuesta a `final`, elección por defecto en todas las clases) \LaTeX no incluirá, por ejemplo, las figuras en el documento, si no que pintará en su lugar un recuadro con el nombre de la imagen en el interior. Además, en el modo `draft` algunos fallos serán más fáciles de detectar, como por ejemplo imágenes que desbordan los márgenes del texto o palabras que \LaTeX no sabe segmentar y que invaden el margen derecho, pues se dibujarán marcas indicativas en los lugares donde se produzcan esos errores.

3.2. Estructuración de documentos extensos

Como hemos visto, \LaTeX está preparado para hacer frente a la creación de documentos muy extensos. Sin embargo, pensar en una tesis o un libro editado en un sólo fichero, hace pensar en algo tremendamente grande y poco manejable. Está claro que no vamos a abordar tarea semejante en el capítulo 3, pero sí explicaremos cómo hacerlo, pues aunque puede que no muchos de nosotros nos

	article	proc	book	report
10pt	✓ ^a	✓	✓	✓
11pt	× ^b	×	×	×
12pt	×	×	×	×
letterpaper	✓	✓	✓	✓
legalpaper	×	×	×	×
executivepaper	×	×	×	×
a4paper	×	×	×	×
a5paper	×	^c	×	×
b5paper	×		×	×
oneside	✓	✓	×	✓
twoside	×	×	✓	×
openright			✓	×
openany			×	✓
onecolumn	✓		✓	✓
twocolumn	×	✓	×	×
notitlepage	✓	✓	×	×
titlepage	×		✓	✓
final	✓	✓	✓	✓
draft	×	×	×	×

^a✓ significa opción por defecto.

^b× significa opción disponible.

^cEspacio en blanco significa opción no disponible.

Cuadro 3.1: Diferencias entre las distintas clases de documentos L^AT_EX

propongamos escribir un libro o una tesis algún día, probablemente sí redactaremos memorias o apuntes, informes o artículos y conocer la forma de estructurar no sólo lógicamente, sino físicamente un documento también puede sernos de utilidad.

Para afrontar esta tarea \LaTeX pone a nuestra disposición dos comandos fundamentales:

```
\input{fichero}  
\include{fichero}
```

Ambos realizan la misma función, reemplazar el propio comando por el contenido del archivo *fichero*. Las únicas diferencias son:

- el comando `include` no puede “anidarse”, es decir, el archivo *fichero* no podría contener a su vez más comandos `include`
 - el comando `include` asume siempre que la extensión del archivo es `.tex` (es decir, en el ejemplo, buscaría el archivo *fichero.tex*), mientras que a `input` pueden indicársele ficheros con otras extensiones (en caso de no especificar extensión alguna, también asumirá que es `.tex`)
 - para cada archivo referido mediante `include`, \LaTeX generará su propio fichero `.aux` (cosa que no sucederá con `input`), con lo cual las compilaciones serán más ágiles (puesto que la información auxiliar necesaria para \LaTeX referida a las partes del documento incluidas de este modo que no hayan sufrido modificaciones ya estará generada)
 - el comando `include` genera una nueva página al ejecutarse, y también al finalizar
-

Capítulo 4

Formato de documentos

Índice general

4.1. Portadas automáticas de \LaTeX	33
4.2. División lógica de un documento	34
4.2.1. Índice	36
4.3. Encabezados y pies de página	36

UNA tarea no poco importante a la hora de componer un documento, es decidir el formato que se le dará, la división lógica en que se estructurará y, por qué no, detalles como la portada o los encabezados y pies de página. En este capítulo nos ocuparemos de estas cuestiones.

4.1. Portadas automáticas de \LaTeX

Ya veíamos en el capítulo anterior que \LaTeX puede realizar acciones referidas a la portada de los documentos (`titlepage`, en su propia nomenclatura). Efectivamente, \LaTeX es capaz de generar automáticamente portadas sencillas y elegantes, a partir de una serie de datos que le indicaremos mediante los correspondientes comandos:

Título – Se proporciona mediante el comando

```
\title{Título del documento}
```

Autor – Se toma del comando

```
\author{Autor o autores del documento}
```

Aunque no es obligatorio incluir el nombre del autor o autores, L^AT_EX nos avisará si lo omitimos:

```
LaTeX Warning: No \author given.
```

Fecha – Además del título y el autor, en la portada que L^AT_EX genera constará además la fecha, que se corresponderá con la fecha de la última *compilación* del documento. Si queremos que la fecha tenga otro valor, o que no aparezca, debemos utilizar el comando `\date{fecha}`:

<code>\date{}</code>	fecha vacía
<code>\date{Noviembre de 2004}</code>	fecha “personalizada”
<code>\date{\today}</code>	mismo efecto que L ^A T _E X

Estos tres comandos se colocarán en el *preámbulo* del documento (recordemos, entre el `\documentclass` y el `\begin{document}`). Con esto L^AT_EX tiene donde obtener la información, pero para indicarle que efectivamente genere la portada debemos incluir el comando `\maketitle` en el *cuerpo* del documento (normalmente, justo después del citado `\begin{document}`).

4.2. División lógica de un documento

Dependiendo de la clase de documento que hayamos indicado en el comando `\documentclass`, tendremos a nuestra disposición un conjunto de comandos

destinados a dividirlo y estructurarlo lógicamente en partes, capítulos, secciones, subsecciones, etc. La lista completa de los mismos aparece en la tabla 4.1.

Todos los comandos de estructuración tienen la misma sintaxis: reciben un argumento obligatorio (el título de la división) y pueden recibir uno opcional (una versión generalmente más corta del título de la división, que aparecerá en índices, encabezados, etc). Por ejemplo:

```
\section[Introducción]{Introducción a la edición de textos}
```

La numeración de capítulos, secciones, subsecciones, etc. es correlativa y automáticamente manejada por \LaTeX . Por el contrario, las *partes* se numeran de manera independiente (también de manera transparente al usuario).

	article y proc	book y report
<i>Parte</i> ($\backslash\text{part}$)	✓	✓
<i>Capítulo</i> ($\backslash\text{chapter}$)		✓
<i>Sección</i> ($\backslash\text{section}$)	✓	✓
<i>Subsección</i> ($\backslash\text{subsection}$)	✓	✓
<i>Subsubsección</i> ($\backslash\text{subsubsection}$)	✓	✓
<i>Párrafo</i> ($\backslash\text{paragraph}$)	✓	✓
<i>Subpárrafo</i> ($\backslash\text{subparagraph}$)	✓	✓

Cuadro 4.1: Comandos de estructuración de documentos \LaTeX

Normalmente suelen usarse las divisiones en secciones y subsecciones, y en el caso de documentos un poco más extensos, en capítulos. La división en partes puede ayudar a dividir un documento realmente grande en varios bloques. La existencia de los comandos $\backslash\text{paragraph}$ y $\backslash\text{subparagraph}$ no debe confundirnos, pues su uso no es obligatorio para organizar el texto en distintos párrafos. Para ello es suficiente con la inclusión de una o más líneas en blanco entre los párrafos en el código fuente. Independientemente del número de líneas en blanco, esto es interpretado por \LaTeX como un punto y aparte¹. Para conseguir que entre

¹Es el mismo comportamiento que ante uno o más espacios en blanco entre palabras: \LaTeX siempre lo toma como uno solo.

párrafos se deje una línea en blanco en el documento final es necesario indicar la secuencia `\\` al final del párrafo fuente seguida de una o más líneas en blanco antes del párrafo que sigue.

Además de los indicados en la tabla 4.1, existe el comando `\appendix`. A partir del lugar de su inclusión en un documento, las unidades `chapter` de `book` y `report` y las unidades `section` de `article` y `proc` serán tratadas de distinta manera, en calidad de apéndices (la numeración se reinicia y cambia su estilo).

4.2.1. Índice

Una vez que estructuramos nuestro documento, es muy probable que queramos incluir un índice del mismo en algún lugar (bien al principio, bien al final). Para ello \LaTeX proporciona el comando:

```
\tableofcontents
```

En el lugar donde lo coloquemos en el documento, el compilador incluirá el índice generado a partir de la información de partes, capítulos, secciones y demás divisiones. Además, según el documento vaya sufriendo modificaciones, las sucesivas compilaciones se encargarán de actualizar dicho índice (reflejando cambios en los nombres de las divisiones, en las páginas correspondientes, etc), de manera que no tendremos que preocuparnos de nada más.

4.3. Encabezados y pies de página

Hay varios *estilos de página* predefinidos en \LaTeX : `plain`, `empty` y `headings`. Estos estilos determinan el contenido que \LaTeX incluirá en el encabezamiento y el pie de cada página, y se comportan de la siguiente manera:

`plain` determina una cabecera vacía y un pie con el número de página centrado

`empty` vacía tanto la cabecera como el pie

`headings` la cabecera contiene el número de página y el nombre de la estructura *activa* del documento (aquella en la que nos encontramos), es decir, el nombre del capítulo, la sección, etc.

Por defecto, las diferentes clases de documentos se comportan tal y como se indica en la tabla 4.2. Para alterar el comportamiento por defecto puede utilizarse el comando:

`\pagestyle{estilo}`

donde *estilo* es uno de los tres indicados anteriormente. Dicho estilo se aplica a partir del lugar donde se incluya tal orden en el código fuente. Por supuesto, existen comandos más sofisticados para personalizar las cabeceras y pies de nuestros documentos, pero los veremos en el capítulo 8.

	article	proc	book	report
<i>Estilo plain</i>	✓	✓		✓
<i>Estilo headings</i>			✓	

Cuadro 4.2: Estilos por defecto de los documentos L^AT_EX

Capítulo 5

Edición elemental de documentos

Índice general

5.1. Entornos y bloques	40
5.2. Fuentes	41
5.2.1. Familias	41
5.2.2. Perfiles	42
5.2.3. Grosos	43
5.2.4. Tamaños	44
5.2.5. Otros efectos	47
5.3. Listas de elementos	48
5.3.1. Listas no numeradas	48
5.3.2. Listas numeradas	49
5.3.3. Listas descriptivas	50
5.4. Alineado de texto	51
5.5. Notas al pie y al margen	52
5.6. Citas textuales	53
5.7. Texto en columnas	53
5.8. Ejercicios	55

DESPUÉS de revisar los conceptos básicos de la creación y el formato de documentos con L^AT_EX, pasaremos al tema de la edición.

5.1. Entornos y bloques

Antes de entrar en materia con el tema de la edición del texto, conviene que asentemos un par de conceptos: la noción de **entorno** y la noción de **bloque**, puesto que las modificaciones que le hagamos al texto afectarán siempre bien a *entornos*, bien a *bloques* de texto.

Tal y como ya apuntábamos en la sección 2.3.1, un **entorno** es una porción del documento encerrada entre dos comandos `\begin{nombreEntorno}` y `\end{nombreEntorno}`, donde *nombreEntorno* es el nombre que identifica el tipo de entorno en concreto. Según las características del mismo, el texto encerrado en él se mostrará de cierta manera, se podrán utilizar comandos especiales, etc. Ya conocemos un tipo de entorno fundamental: el entorno **document**, que encierra todo el cuerpo del documento.

Otra forma de delimitar texto es mediante **bloques**. Para eso, se utilizan llaves `{ }`¹. Dentro de cada bloque de texto se podrán emplear comandos para aplicar características al texto que permanecerán activas hasta el final del bloque.

Es muy importante recordar que los entornos y bloques se comportan en cierto modo como muñecas rusas. Es decir, se encierran unos dentro de otros y deben cerrarse en el mismo orden en que se abren, “casando” o “encajando” las cabezas (**begins**) con los cuerpos correspondientes (**ends**).

<code>\begin{...}_1</code>	<code>\begin{...}_1</code>
<code>{_2 ... }_2</code>	<code>{_2 ... {_3</code>
<code>{_3</code>	<code>}_3</code>
<code>\begin{...}_4</code>	<code>\begin{...}_4</code>
<code>{_5 ... }_5</code>	<code>{_5 ...</code>

¹Recordemos que las llaves son uno de los caracteres reservados de L^AT_EX (ver página 21).

<pre style="margin: 0;">\end{...}_4 {6 ... }_6 }_3 \end{...}_1</pre>	<pre style="margin: 0;">\end{...}_4 }_5 {6 ... }_2 }_6 \end{...}_1</pre>
✓ <i>correcto</i>	× <i>incorrecto</i>

En el ejemplo anterior, en el bloque de código de la derecha vemos el uso incorrecto de bloques y entornos, en particular el entorno 4 y los bloques 2, 5 y 6 *entrecruzan* sus “áreas de influencia” de manera inapropiada.

5.2. Fuentes

En esta sección nos ocuparemos de todas las modificaciones que podemos realizar sobre el texto que escribimos, tanto en el estilo como en el tamaño, etc.

5.2.1. Familias

En L^AT_EX existen tres familias de tipos de letra: **roman** (normal), **sanserif** (sin adornos) y **typewriter** (tipo máquina de escribir). La familia que se utiliza por defecto es la normal (*roman*).

Para cambiar la familia del tipo de letra del texto puede usarse un comando que toma como argumento el texto al que queremos aplicar la modificación, o bien una orden que actúa dentro de un bloque:

	<i>Comando + argumento</i>	<i>Bloque + orden</i>
Familia roman	<code>\textrm{<i>Texto</i>}</code>	<code>{ \rmfamily <i>Texto</i> }</code>
Familia sanserif	<code>\textsf{<i>Texto</i>}</code>	<code>{ \sffamily <i>Texto</i> }</code>
Familia typewriter	<code>\texttt{<i>Texto</i>}</code>	<code>{ \ttfamily <i>Texto</i> }</code>

En general, si se quiere aplicar la modificación a una o varias palabras, e incluso a una frase completa, se utilizará la primera opción, mientras que si se la quiere emplear para afectar a un fragmento mayor de código, como un párrafo entero, es más recomendable (por claridad) la segunda.

Una forma alternativa al uso de un bloque y la orden `\XXfamily` dentro de él, es la utilización de entornos del mismo nombre (`XXfamily`), que también resultan preferibles para afectar a párrafos completos:

	<i>Entorno</i>
Familia roman	<code>\begin{rmfamily} Texto \end{rmfamily}</code>
Familia sanserif	<code>\begin{sffamily} Texto \end{sffamily}</code>
Familia typewriter	<code>\begin{ttfamily} Texto \end{ttfamily}</code>

5.2.2. Perfiles

Disponemos de cuatro perfiles de letra en cada familia de tipos de letra \LaTeX : **recto** (normal, perfil por defecto), **itálico**, **inclinado** y **versalita**. Siguiendo el mismo esquema anterior, pueden conseguirse de la siguiente manera²:

	<i>Comando + argumento</i>	<i>Bloque + orden</i>
Perfil recto	<code>\textup{Texto}</code>	<code>{ \upshape Texto }</code>
<i>Perfil itálico</i>	<code>\textit{Texto}</code>	<code>{ \itshape Texto }</code>
<i>Perfil inclinado</i>	<code>\textsl{Texto}</code>	<code>{ \slshape Texto }</code>
PERFIL VERSALITA	<code>\textsc{Texto}</code>	<code>{ \scshape Texto }</code>

Igual que en el caso anterior, en lugar de un bloque de texto se pueden utilizar los entornos:

	<i>Entorno</i>
Perfil recto	<code>\begin{upshape} Texto \end{upshape}</code>
<i>Perfil itálico</i>	<code>\begin{itshape} Texto \end{itshape}</code>
<i>Perfil inclinado</i>	<code>\begin{slshape} Texto \end{slshape}</code>
PERFIL VERSALITA	<code>\begin{scshape} Texto \end{scshape}</code>

²Los ejemplos se aplican a la familia roman.

5.2.3. Grosos

Por último, con respecto al estilo de letra, disponemos en \LaTeX de dos grosos básicos: **medio** (normal, grosor por defecto) y **grueso** (negrita). Los comandos a aplicar son²:

²Ver el capítulo 2 de [1] para más detalles.

	<i>Comando + argumento</i>	<i>Bloque + orden</i>
Grosor normal	<code>\textmd{<i>Texto</i>}</code>	<code>{ \mdseries <i>Texto</i> }</code>
Grosor negrita	<code>\textbf{<i>Texto</i>}</code>	<code>{ \bfseries <i>Texto</i> }</code>

Y los entornos se denominan:

	<i>Entorno</i>
Grosor normal	<code>\begin{mdseries} <i>Texto</i> \end{mdseries}</code>
<i>Grosor negrita</i>	<code>\begin{bfseries} <i>Texto</i> \end{bfseries}</code>

Los comandos para aplicar distintas familias, perfiles y grosores de letra al texto pueden combinarse entre sí, sin mayores restricciones que el gusto propio del autor. No obstante, no todas las combinaciones son posibles; por ejemplo, no es posible obtener ninguna variación de la familia **typewriter** (también llamada en ocasiones *monoespaciada*) con grosor negrita, ni tampoco ninguna de la familia sin adornos en cursiva. El cuadro 5.1 muestra un resumen de todas las posibilidades.

5.2.4. Tamaños

Además del estilo, es posible modificar el tamaño del texto. Para ello, L^AT_EX nos presenta 10 comandos, en este caso sólo es posible usarlos en bloques o como entornos:

	<i>Bloque + orden</i>
Diminuto	<code>{ \tiny <i>Texto</i> }</code>
El más pequeño	<code>{ \scriptsize <i>Texto</i> }</code>
Más pequeño	<code>{ \footnotesize <i>Texto</i> }</code>
Pequeño	<code>{ \small <i>Texto</i> }</code>
Normal	<code>{ \normalsize <i>Texto</i> }</code>
Grande	<code>{ \large <i>Texto</i> }</code>
Más grande	<code>{ \Large <i>Texto</i> }</code>
El más grande	<code>{ \LARGE <i>Texto</i> }</code>
Enorme	<code>{ \huge <i>Texto</i> }</code>
El más enorme	<code>{ \Huge <i>Texto</i> }</code>

Roman	{	Recta	{	Normal
				Negrita
		<i>Cursiva</i>	{	<i>Normal</i>
				<i>Negrita</i>
	{	<i>Inclinada</i>	{	<i>Normal</i>
				<i>Negrita</i>
		VERSALITA : NORMAL		
Sanserif	{	Recta	{	Normal
				Negrita
		<i>Inclinada</i>	: <i>Normal</i>	
Typewriter	{	Recta	: Normal	
		<i>Cursiva</i>	: <i>Normal</i>	
		<i>Inclinada</i>	: <i>Normal</i>	
			VERSALITA : NORMAL	

Cuadro 5.1: Combinaciones posibles de estilos de letra en \LaTeX

	<i>Entorno</i>	
<code>\begin{tiny}</code>	<i>Texto</i>	<code>\end{tiny}</code>
<code>\begin{scriptsize}</code>	<i>Texto</i>	<code>\end{scriptsize}</code>
<code>\begin{footnotesize}</code>	<i>Texto</i>	<code>\end{footnotesize}</code>
<code>\begin{small}</code>	<i>Texto</i>	<code>\end{small}</code>
<code>\begin{normalsize}</code>	<i>Texto</i>	<code>\end{normalsize}</code>
<code>\begin{large}</code>	<i>Texto</i>	<code>\end{large}</code>
<code>\begin{Large}</code>	<i>Texto</i>	<code>\end{Large}</code>
<code>\begin{LARGE}</code>	<i>Texto</i>	<code>\end{LARGE}</code>
<code>\begin{huge}</code>	<i>Texto</i>	<code>\end{huge}</code>
<code>\begin{Huge}</code>	<i>Texto</i>	<code>\end{Huge}</code>

Por supuesto, el tamaño por defecto es el que se corresponde con `normalsize`, que es el que puede seleccionarse opcionalmente en la orden `\documentclass`. Los demás tamaños varían con relación a este tamaño base según los valores que constan en la tabla 5.2.

	<i>Opción 10pt</i>	<i>Opción 11pt</i>	<i>Opción 12pt</i>
tiny	5pt	6pt	6pt
scriptsize	7pt	8pt	8pt
footnotesize	8pt	9pt	10pt
small	9pt	10pt	11pt
normalsize	10pt	11pt	12pt
large	12pt	12pt	14pt
Large	14pt	14pt	17pt
LARGE	17pt	17pt	20pt
huge	20pt	20pt	25pt
Huge	25pt	25pt	25pt

Cuadro 5.2: Proporción de tamaños según el tamaño base del documento

5.2.5. Otros efectos

Además de los comandos que hemos visto hasta ahora, existe otra manera de **enfaticar** texto, que consiste en usar el comando:

Enfatizado `\emph{Texto}`

En condiciones normales, esta orden tiene el mismo efecto que `\textit`, es decir, *italiza* el texto. Sin embargo, `\emph{}` tiene la peculiaridad de que siempre enfatiza el *Texto* para diferenciarlo del resto del texto a su alrededor, de manera que si cambian las características de ese texto circundante (haciéndose itálico, por ejemplo), L^AT_EX optaría por mostrar el *Texto enfatizado* recto, distinguiéndolo igualmente.

Existe también la posibilidad de **subrayar** texto usando el comando:

Subrayado `\underline{Texto}`

Sin embargo, el subrayado no es una forma de resaltar texto aconsejada cuando se dispone de distintos estilos de letra. De hecho, antiguamente, cuando en las composiciones sólo se disponía de un tipo de letra o se escribía a mano, el texto subrayado indicaba al impresor que debía italizarse.

Por último, existe en L^AT_EX un comando que nos permite obtener en el documento final el texto tal como lo tecleemos en el código fuente, es decir, respetando todo tipo de espacios, líneas en blanco, etc. Es un entorno donde todos los demás caracteres reservados L^AT_EX, así como todos los comandos, órdenes y variables quedan inactivos y dejan de ser indicaciones para pasar a ser simple texto. Este entorno aplica a la fuente la familia `typewriter` y se denomina `verbatim`:

`\begin{verbatim}`

En este entorno se pueden dejar	todos los espacios
que se quieran tanto entre	palabras

como

entre líneas, pues serán respetados, y escribir `\cualquiercomando[con]{o sin}` opciones, incluso aunque no exista.

Por supuesto, `\LaTeX{}` aquí no justifica nada de nada.

`\end{verbatim}`

5.3. Listas de elementos

Otro de los elementos más usados en edición de documentos, una vez examinados los comandos relativos a fuentes, son aquéllos que nos permiten estructurar las ideas que vamos exponiendo a lo largo del texto, resaltando puntos importantes o enumerando características. En esta sección veremos tres tipos distintos de entornos que nos sirven para este fin.

5.3.1. Listas no numeradas

En primer lugar, consideraremos las 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}

```

Por supuesto, este tipo de listas pueden anidarse. L^AT_EX se encarga de la gestión de la apariencia de los distintos niveles de profundidad:

	<pre>\begin{itemize}</pre>
	<pre>\item leche</pre>
■ leche	<pre>\item pan y cereales</pre>
	<pre>\begin{itemize}</pre>
■ pan y cereales	<pre>\item trigo</pre>
● trigo	<pre>\begin{itemize}</pre>
○ harina	<pre>\item harina</pre>
	<pre>\end{itemize}</pre>
● cebada	<pre>\item cebada</pre>
● centeno	<pre>\item centeno</pre>
● maíz	<pre>\item maíz</pre>
	<pre>\end{itemize}</pre>
■ legumbres	<pre>\item legumbres</pre>
● lentejas	<pre>\begin{itemize}</pre>
● garbanzos	<pre>\item lentejas</pre>
	<pre>\item garbanzos</pre>
	<pre>\end{itemize}</pre>
	<pre>\end{itemize}</pre>

5.3.2. Listas numeradas

Las listas numeradas, como su propio nombre indica, son enumeraciones de elementos:

	<code>\begin{enumerate}</code>
1. buscar un local	<code>\item buscar un local</code>
<i>a)</i> telefonear	<code>\begin{enumerate}</code>
<i>b)</i> visitar el sitio	<code>\item telefonear</code>
<i>c)</i> confirmarlo	<code>\item visitar el sitio</code>
2. enviar las invitaciones	<code>\item confirmarlo</code>
3. contratar la decoración	<code>\end{enumerate}</code>
	<code>\item enviar las invitaciones</code>
	<code>\item contratar la decoración</code>
	<code>\end{enumerate}</code>

5.3.3. Listas descriptivas

El último tipo, las listas descriptivas, es un tipo especial de lista que resalta una palabra clave, del siguiente modo:

prosa estructura o forma del lenguaje	<code>\begin{description}</code>
que...	<code>\item [prosa] estructura o forma</code>
	<code>del lenguaje que\ldots</code>
verso palabra o conjunto de palabras	<code>\item [verso] palabra o conjunto de</code>
sujetas...	<code>palabras sujetas\ldots</code>
	<code>\end{description}</code>

Se usa normalmente para descripciones de términos o similares.

Por supuesto, todos los tipos de listas pueden combinarse entre sí sin restricción alguna, anidándolas como nos apetezca (claro que siempre respetando las mismas normas que para el resto de entornos).

5.4. Alineado de texto

Ya hemos comprobado que L^AT_EX justifica siempre el texto a ambos lados por defecto. Aunque éste será normalmente el efecto deseado para cualquier tipo de texto, en caso de que no lo sea disponemos de tres entornos para alinear el texto a izquierda, derecha y centrado:

Texto alineado a la izquierda

Se realiza con mediante el entorno `flushleft`:

```
\begin{flushleft}
Este texto aparecerá alineado a la izquierda \\
y sin justificar.
\end{flushleft}
```

*Este texto aparecerá alineado a la izquierda
y sin justificar.*

Si en lugar de alinear a la izquierda un bloque de texto queremos hacer lo propio con una sola línea de texto, puede sernos igual de útil el comando `\leftline{Texto}`.

Texto alineado a la derecha

Se consigue gracias al entorno `flushright`:

```
\begin{flushright}
Este texto aparecerá alineado a la derecha \\
y sin justificar.
\end{flushright}
```

*Este texto aparecerá alineado a la derecha
y sin justificar.*

Igual que en el caso anterior, si el texto cubre toda una línea, entonces \LaTeX sí lo ajustará a los márgenes. También disponemos en este caso del comando `\rightline{Texto}`.

Texto centrado

El entorno correspondiente se denomina `center`:

```
\begin{center}
Este texto aparecerá centrado \\
y sin justificar.
\end{center}
```

*Este texto aparecerá centrado
y sin justificar.*

La orden `\centerline{Texto}` puede aplicarse a una sola línea de texto.

5.5. Notas al pie y al margen

Introducir notas a pie de página o al margen es tremendamente fácil en \LaTeX , con los comandos:

```
\footnote{Texto de la nota al pie}
\marginpar{Texto de la nota al margen}
```

Como en otros muchos casos, es \LaTeX quien se encargará, en el caso de las notas a pie, de numerarlas adecuadamente y de modificar la numeración si incluimos una nota entre otras dos anteriores, etc.

5.6. Citas textuales

Ya para terminar este tema, veremos dos entornos de propósito específico: `quote` y `quotation`. Ambos están pensados para incluir citas textuales, por lo que modifican los márgenes de la página con el fin de que el párrafo o párrafos incluidos resalten en medio del resto del texto.

Esto es un párrafo incluido dentro de un entorno `quotation`. Los márgenes se hacen más grandes para que el texto resalte en la página.

Este entorno respeta la sangría de la primera línea habitual en la tipografía española.

La diferencia entre ambos reside en que `quote` suprime la sangría de la primera línea y aumenta ligeramente el espaciado entre párrafos:

Esto es un párrafo incluido dentro de un entorno `quote`. Como se puede ver, se ha suprimido la sangría de la primera línea.

Además, también vemos que la distancia entre párrafos es algo mayor.

5.7. Texto en columnas

Utilizando la opción `twocolumn` del comando `\documentclass`, que ya vimos, podemos obtener documentos que se maquetan en formato de columnas periodísticas. Sin embargo, de manera puntual, puede interesarnos incluir entre nuestro texto algún fragmento que se muestre en varias columnas, sin que lo esté todo el documento. Para ello resulta muy útil el paquete `multicol`. Incluyendo la orden `\usepackage{multicol}` en el preámbulo del documento dispondremos del siguiente entorno:

```
\begin{multicols}{2}
Una vez más, \LaTeX{} trabaja por nosotros y distribuye
el texto automáticamente en tantas columnas como
indiquemos como argumento obligatorio del propio entorno
\texttt{multicols}. Sencillo y eficaz.
\end{multicols}
```

Una vez más, \LaTeX automáticamente en tan- obligatorio del propio en-
trabaja por nosotros y tas columnas como indi- torno `multicols`. Sencillo
distribuye nuestro texto quemos como argumento y eficaz.

5.8. Ejercicios

Ejercicio 5.1 Determinar el correspondiente texto fuente de:

Bit: Un dígito binario que puede ser un 1 o un 0 en el sistema de numeración binario.

Byte: Un grupo de 8 bits usado para representar valores desde el 0 hasta el 255, que representan un carácter alfanumérico, un carácter gráfico o un carácter de control especial.

Caché de disco: Una porción de memoria que almacena información leída del disco. Si el computador necesita esta información de nuevo, la lee de la caché de disco y no del disco duro, que es más lento.

Dirección: Una posición en memoria. Cada posición de memoria tiene una dirección.

Estado de espera: Una pausa que se genera cuando el microprocesador tiene que esperar que se lea un dato en la memoria.

Ejercicio 5.2 Introducir en el código anterior los cambios necesarios para obtener:

Bit: Un dígito binario que puede ser un 1 o un 0 en el sistema de numeración binario.

BYTE Un grupo de 8 bits usado para representar valores desde el 0 hasta el 255, que representan un carácter alfanumérico, un carácter gráfico o un carácter de control especial.

Caché de disco: Una porción de memoria que almacena información leída del disco. Si el computador necesita esta información de nuevo, la lee de la **caché de disco** y no del disco duro, que es más lento.

Dirección: Una posición en memoria. Cada posición de **memoria** tiene una dirección.

Estado de espera: Una pausa que se genera cuando el microprocesador tiene que esperar que se lea un dato en la *memoria*.

Ejercicio 5.3 En el siguiente párrafo se mezclan distintos tipos de listas. Determinar el texto fuente que le corresponde:

Los contenidos que se imparten en la parte de Variable compleja son:

1. Números complejos
 - Construcción de los números complejos
 - Forma polar. Forma exponencial
2. Funciones analíticas
 - a) Funciones de una variable compleja
 - b) Ecuaciones de Cauchy-Riemann
3. Integración compleja

Teorema de Cauchy–Goursat

Primitivas e Independencia del camino

Ejercicio 5.4 Hacer los cambios necesarios al código anterior para obtener la siguiente variación:

Los contenidos que se imparten en la parte de Variable compleja son:

1. **Números complejos**
 - Construcción de los *números complejos*
 - Forma polar. *Forma exponencial*
2. Funciones analíticas
 - a) Funciones de una *variable compleja*
 - b) Ecuaciones de Cauchy-Riemann
3. Integración compleja

Teorema de Cauchy–Goursat

Primitivas e Independencia del camino

Ejercicio 5.5 Una vez más, determinar el texto fuente para producir:

Los objetivos que nos marcaremos en este tema son:

- Concepto de derivada y aplicaciones
 - Definición mediante el límite de un cociente
 - Interpretación geométrica
 - Diferencial
 1. Concepto
 2. Interpretación geométrica
- Aproximación local de una función mediante polinomios
 - Resolución de límites
 - Ejemplos
- Estudio local de una función
 - Representación gráfica aproximada de curvas planas

Ejercicio 5.6 Cambiar los estilos de letra en el resultado del ejercicio anterior para conseguir:

Los **objetivos** que nos marcaremos en este tema son:

- **Concepto de derivada y aplicaciones**
 - *Definición mediante el límite de un cociente*
 - *Interpretación geométrica*
 - *Diferencial*
 1. Concepto
 2. Interpretación geométrica
- **Aproximación local de una función mediante polinomios**
 - *Resolución de límites*
 - *Ejemplos*
- **Estudio local de una función**
 - *Representación gráfica aproximada de curvas planas*

Ejercicio 5.7 Reproducir el siguiente texto, aplicando lo visto en la sección 5.4:

Texto alineado a la izquierda Se usa el entorno *flushleft*:

Este texto aparecerá alineado a la izquierda
y sin justificar.

```
\begin{flushleft}
  Este texto aparecerá alineado a la izquierda \\
  y sin justificar.
\end{flushleft}
```

Texto alineado a la derecha Se usa el entorno *flushright*:

Este texto aparecerá alineado a la derecha
y sin justificar.

```
\begin{flushright}
  Este texto aparecerá alineado a la derecha \\
  y sin justificar.
\end{flushright}
```

Texto centrado Se usa el entorno *center*:

Este texto aparecerá centrado
y sin justificar.

```
\begin{center}
  Este texto aparecerá centrado \\
  y sin justificar.
\end{center}
```

Ejercicio 5.8 Como último ejercicio, probaremos los entornos de citas textuales y las notas a pie de página y al margen:

Cervantes escribió:

¡qué famosa frase!

“En un lugar de la Mancha de cuyo nombre no quiero acordarme, no ha mucho que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

(...) Puesto nombre, y tan a su gusto, a su caballo, quiso ponérsele a sí mismo, y en este pensamiento duró otros ocho días, y al cabo se vino a llamar don Quijote...”

Obsérvense las diferencias con:

“En un lugar de la Mancha de cuyo nombre no quiero acordarme, no ha mucho que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor.

(...) Puesto nombre, y tan a su gusto, a su caballo, quiso ponérsele a sí mismo, y en este pensamiento duró otros ocho días, y al cabo se vino a llamar don Quijote³...”

³De *Don Quijote de La Mancha*, obra de Miguel de Cervantes Saavedra.

Edición especial de documentos

Índice general

6.1. Edición matemática	62
6.1.1. Entornos	62
6.1.2. Paquetes	63
6.1.3. Fórmulas a diestro y siniestro	64
6.2. Objetos flotantes: tablas y figuras	73
6.2.1. ¿Qué es “ <i>flotar</i> ”?	73
6.2.2. Tablas	74
6.2.3. Imágenes y gráficos	77
6.3. Cartas	80
6.4. Ejercicios	83

EN este capítulo trataremos algunos aspectos más avanzados de la edición de documentos, como pueden ser la inclusión de gráficos o tablas y una pequeña incursión en el potente y extensísimo ámbito matemático, el gran punto fuerte de \LaTeX . También habrá lugar para algunas pinceladas sobre otros temas, como la creación de cartas.

6.1. Edición matemática

La edición matemática es el terreno en el que L^AT_EX mejor demuestra todo su potencial. En las próximas secciones aprenderemos cómo utilizar el modo matemático y repasaremos brevemente los comandos más conocidos.

6.1.1. Entornos

Existen dos tipos de entornos matemáticos en L^AT_EX:

1. El entorno `math` o su equivalente, `$... $`.
2. El entorno `displaymath` o su equivalente, `$$... $$`

La diferencia entre uno y otro es que el primero se utiliza para la inclusión de formulación matemático-científica *inline*, es decir, en medio de un párrafo de texto, mientras que la segunda opción inicia un nuevo párrafo centrado.

Ejemplo de utilización del entorno `math` $a + b = c$ y `displaymath`:

$$a + b = c$$

Ejemplo de utilización del entorno `\texttt{math}`

```
\begin{math}
  a + b = c
\end{math}
y \texttt{displaymath}:
\begin{displaymath}
  a + b = c
\end{displaymath}
```

Además de estos dos entornos básicos, disponemos también de un tercer entorno, `equation`, que añade la propiedad de numeración (algo que será útil en caso de querer hacer referencia a una fórmula –véase capítulo 7, sección 7.1–).

Ejemplo de uso del entorno
`equation`:

$$a + b = c \quad (6.1)$$

Ejemplo de uso del entorno
`\texttt{equation}`:
`\begin{equation}`
`a + b = c`
`\end{equation}`

Como se puede ver, \LaTeX añade a la derecha de toda fórmula incluida en el entorno `equation` la numeración correspondiente, generada automáticamente. Este comportamiento puede alterarse ligeramente mediante un par de opciones del comando `\documentclass`:

leqno Cambia el emplazamiento por defecto de la numeración de los entornos `equation`, que aparecerá a la izquierda en lugar de a la derecha.

fleqn Hace que \LaTeX coloque las fórmulas a una distancia fija del margen izquierdo, en lugar de centradas.

6.1.2. Paquetes

Aunque muchos de los recursos y comandos más empleados del entorno matemático están incluidos en los paquetes que \LaTeX utiliza por defecto, existen tres paquetes importantes y de gran utilidad:

latexsym Ofrece al usuario un gran conjunto de símbolos matemáticos.

amsmath, **amssymb** Dos paquetes que, siguiendo el estándar de la American Mathematical Society, proporcionan diferentes comandos y símbolos.

Su inclusión en el preámbulo de cualquier documento en el que se vaya a emplear formulación matemático-científica es más que recomendable, y para la reproducción de algunos de los ejemplos que veremos en las próximas páginas, necesaria.

6.1.3. Fórmulas a diestro y siniestro

El entorno matemático de \LaTeX es muy descriptivo. La mayoría de los comandos y símbolos tienen nombres fáciles de recordar porque se corresponden con las abreviaturas de los nombres que reciben en inglés (y que, al tratarse de símbolos internacionalmente usados, son a su vez muy similares a los que se usan en español, por ejemplo).

Editar fórmulas en el entorno matemático de \LaTeX es, en principio, tan sencillo como abrir un entorno `math` o `displaymath` y comenzar a escribirla casi de la misma forma que la leeríamos. En los siguientes apartados veremos cómo se indican los recursos más habituales en esta notación, desde potencias o raíces hasta integrales, matrices o determinantes.

6.1.3.1. Superíndices y subíndices

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:

$E = mc^2$	<pre>\begin{displaymath} E = m c^2 \end{displaymath}</pre>
$a_{n+1} = a_n + 1$	<pre>\begin{displaymath} a_{n+1} = a_n + 1 \end{displaymath}</pre>

Lo único que hay que tener en cuenta es que cuando el super/subíndice está compuesto por más de un carácter (como en el caso de a_{n+1}) es necesario indicarlo entre llaves, de forma que \LaTeX sepa que es todo el contenido del bloque que lo que queremos que forme parte de dicho índice. De lo contrario, podríamos

obtener un resultado no deseado, como:

$$a_n + 1 \qquad \texttt{a_n+1}$$

6.1.3.2. Fracciones y binomios

Otra parte del lenguaje matemático que probablemente querremos utilizar en seguida son las fracciones. El comando básico para crear una fracción es:

$$\frac{1}{2} = \frac{2}{4}$$

```
\begin{displaymath}
\frac{1}{2} = \frac{2}{4}
\end{displaymath}
```

Donde los dos argumentos obligatorios son, respectivamente, el numerador y el denominador. Si utilizamos el comando `\frac` dentro de un entorno `math`, obtendremos $\frac{1}{2} = \frac{2}{4}$. Para conseguir que el tamaño de una fórmula de este tipo en modo *inline* sea el mismo que en modo *display*, debemos utilizar el comando `\dfrac`: $\frac{1}{2} = \frac{2}{4}$ (`\dfrac{1}{2}=\dfrac{2}{4}`). Para conseguir el efecto contrario, es decir, tamaño *inline* en entorno *display* existe el recíproco `\tfrac`:

$$\frac{1}{2} = \frac{2}{4}$$

```
\begin{displaymath}
\tfrac{1}{2}=\tfrac{2}{4}
\end{displaymath}
```

En cuanto a los binomios, los comandos, totalmente análogos, son `\binom`, `\dbinom` y `\tbinom`:

$$\binom{5}{9} + \binom{11}{2}$$

```
\begin{displaymath}
\binom{5}{9} + \tbinom{11}{2}
\end{displaymath}
```

6.1.3.3. Raíces

Las raíces se escriben en el modo matemático L^AT_EX del siguiente modo:

$$\sqrt[3]{a+b}$$

```
\begin{displaymath}
\sqrt[3]{a+b}
\end{displaymath}
```

El argumento opcional es el radical de la raíz, y el obligatorio el radicando. El contenido de ambos puede ser tan grande como sea necesario, pues será ajustado automáticamente:

$$\sqrt[n+1]{\frac{a_n + b_n - 2c^2}{2}}$$

```
\begin{displaymath}
\sqrt[n+1]{\frac{a_n+b_n-2c^2}{2}}
\end{displaymath}
```

6.1.3.4. Integrales, derivadas, sumatorios, límites

El siguiente paso que daremos va en la dirección de los operadores de integración, derivación, sumatorios, productos, límites y funciones de diversa índole. Sirvan de ilustración los siguientes ejemplos:

$$\int 2x \partial x = x^2$$

```
\begin{displaymath}
\int 2x \partial x = x^2
\end{displaymath}
```

$$\sum (x+i) + \prod (x-i)$$

```
\begin{displaymath}
\sum (x+i) + \prod (x-i)
\end{displaymath}
```

$$\lim_{2x} \frac{x^2}{2x} = \infty$$

```
\begin{displaymath}
\lim \frac{x^2}{2x} = \infty
\end{displaymath}
```

Para colocar índices a este tipo de operadores se procede de la misma manera que se colocan super/subíndices a cualquier otro elemento de una fórmula:

$$\sum_{i=0}^n (x+i) + \lim_{x \rightarrow \infty} x$$

```

\begin{displaymath}
  \sum_{i=0}^n (x+i)
+ \lim_{x \rightarrow \infty} x
\end{displaymath}

```

6.1.3.5. Cuantificadores y otras funciones

L^AT_EX dispone de sendos comandos para proporcionar los cuantificadores universal (*para todo*, $\forall = \texttt{\backslashforall}$) y existencial (*existe*, $\exists = \texttt{\backslashexists}$ y la negación *no existe*, $\nexists = \texttt{\backslashnexists}$).

Asimismo, comandos bastante sencillos proporcionan funciones como el seno ($\sin = \texttt{\backslashsin}$), coseno ($\cos = \texttt{\backslashcos}$), tangente ($\tan = \texttt{\backslashtan}$), cotangente ($\cot = \texttt{\backslashcot}$), logaritmo ($\log = \texttt{\backslashlog}$), logaritmo neperiano ($\ln = \texttt{\backslashln}$), máximo ($\max = \texttt{\backslashmax}$), mínimo ($\min = \texttt{\backslashmin}$), etc.

6.1.3.6. Texto dentro del entorno matemático

Si probamos a escribir texto normal dentro del entorno matemático

```

nosllevaremosunasorpresa
\begin{displaymath}
  nos llevaremos una sorpresa
\end{displaymath}

```

Para escribir texto “normal” dentro de fórmulas matemáticas, disponemos del comando $\texttt{\backslashtext{Texto}}$:

```

\begin{displaymath}
  \text{esto ya es }
  \textbf{\textit{otra}} \text{ cosa}
\end{displaymath}

```

Además, como se puede ver, al texto incluido en un comando `\text` se le pueden aplicar toda clase de comandos de estilo (cambio de familia, de perfil, de grosor o incluso de tamaño).

6.1.3.7. Llaves y flechas

Algo que también nos puede resultar útil son los comandos para dibujar distintos tipos de flechas y comandos de agrupación:

$a \rightarrow b \Rightarrow c \Leftarrow d \leftarrow d$	<pre>\begin{displaymath} a \rightarrow b \Rightarrow c \Leftarrow d \leftarrow d \end{displaymath}</pre>
$\underbrace{a+b+c+d}_x = \overbrace{e+f+g+h}^y$	<pre>\begin{displaymath} \underbrace{a+b+c+d}_{x} = \overbrace{e+f+g+h}^y \end{displaymath}</pre>

6.1.3.8. Matrices y determinantes

Para editar matrices y/o determinantes, existe el entorno `array`, que funciona de la siguiente manera:

$\begin{pmatrix} a_1 & a_2 & \cdots & a_m \\ b_1 & b_2 & \cdots & b_m \\ \vdots & \cdots & \ddots & \vdots \\ n_1 & n_2 & \cdots & n_m \end{pmatrix}$	<pre>\begin{displaymath} \left(\begin{array}{cccc} a_1 & & a_2 & & \cdots & & a_m \\ b_1 & & b_2 & & \cdots & & b_m \\ \vdots & & \cdots & & \ddots & & \vdots \\ n_1 & & n_2 & & \cdots & & n_m \end{array} \right)</pre>
---	---

```
\end{array}                                \end{displaymath}
\right)
```

El entorno `array` es un tipo de entorno especial, que recibe argumentos como cualquier otro comando. En particular, recibe como argumento obligatorio una secuencia de caracteres, uno por cada columna que vaya a tener la matriz o determinante. Dicho carácter indica la alineación horizontal del contenido de la columna correspondiente, pudiendo ser:

c el contenido se centra (del inglés, *center*)

l el contenido se alinea a la izquierda (*left*)

r el contenido se alinea a la derecha (*right*)

Después, el contenido del entorno se estructura por filas, cuyo final se marca con la secuencia `\\`, igual que un salto de párrafo. Dentro de cada fila, el contenido de cada celda se separa mediante el carácter reservado `&`.

6.1.3.9. Puntos suspensivos y delimitadores

El ejemplo anterior nos ha servido, además, para ilustrar cómo se consiguen puntos suspensivos en todas las direcciones posibles:

- normales, igual que en el entorno no matemático: `\dots` (`...`)
- centrados verticalmente con respecto a la línea de escritura: `\cdots` (`...`)
- verticales: `\vdots` (`:`)
- diagonales: `\ddots` (`...`)

En cuanto a los delimitadores, para conseguir que tengan un tamaño ajustado al contenido simplemente usamos las secuencias `\leftdelimitador` para abrir y `\rightdelimitador` para cerrar, siendo *delimitadores* posibles:

paréntesis		(tal como hemos visto)
barra vertical	$\left \frac{1}{2} \right $	<code>\left \dfrac{1}{2} \right </code>
corchetes	$\left[\frac{2}{3} \right]$	<code>\left[\dfrac{2}{3} \right]</code>
llaves	$\left\{ \frac{3}{4} \right\}$	<code>\left\{ \dfrac{3}{4} \right\}</code>

Nótese que las llaves, que de por sí son un carácter reservado, han de escaparse cuando quieren ser utilizadas como delimitador.

Además, es posible indicar sólo uno de los dos delimitadores (sólo el izquierdo o sólo el derecho). Para ello, no obstante, no vale simplemente no poner el delimitador en cuestión, sino que hay que sustituirlo por un `\left.` o `\right.` según convenga. De este modo, el entorno `array` puede utilizarse perfectamente para crear sistemas de ecuaciones:

	<code>\begin{displaymath}</code>
	<code>\left\{</code>
$\left\{ \begin{array}{l} a + b = 4 \\ 2a + 3b = 36 \end{array} \right.$	<code>\begin{array}{rcl}</code> <code>a + b & = & 4 \\</code> <code>2a + 3b & = & 36 \\</code> <code>\end{array}</code>
	<code>\right.</code>
	<code>\end{displaymath}</code>

6.1.3.10. Símbolos y espacios

Los símbolos y operadores que ya hemos visto son sólo una pequeñísima muestra de la gran cantidad de simbología matemático-técnica que podemos utilizar en \LaTeX . La tabla 6.1 es una compilación de algunos otros símbolos útiles, como las letras griegas más utilizadas o símbolos como el del conjunto vacío.

α	<code>\alpha</code>	β	<code>\beta</code>	γ	<code>\gamma</code>	δ	<code>\delta</code>
ϵ	<code>\epsilon</code>	η	<code>\eta</code>	θ	<code>\theta</code>	κ	<code>\kappa</code>
λ	<code>\lambda</code>	μ	<code>\mu</code>	ν	<code>\nu</code>	π	<code>\pi</code>
ρ	<code>\rho</code>	σ	<code>\sigma</code>	τ	<code>\tau</code>	ϕ	<code>\phi</code>
χ	<code>\chi</code>	ψ	<code>\psi</code>	ω	<code>\omega</code>		
Γ	<code>\Gamma</code>	Δ	<code>\Delta</code>	Θ	<code>\Theta</code>	Λ	<code>\Lambda</code>
Π	<code>\Pi</code>	Σ	<code>\Sigma</code>	Φ	<code>\Phi</code>	Ψ	<code>\Psi</code>
Ω	<code>\Omega</code>						
\times	<code>\times</code>	\div	<code>\div</code>				
\oplus	<code>\oplus</code>	\ominus	<code>\ominus</code>	\otimes	<code>\otimes</code>		
∇	<code>\nabla</code>	\surd	<code>\surd</code>	\top	<code>\top</code>	\perp	<code>\perp</code>
\vee	<code>\vee</code>	\wedge	<code>\wedge</code>	\cap	<code>\cap</code>	\cup	<code>\cup</code>
\in	<code>\in</code>	\notin	<code>\notin</code>	\subset	<code>\subset</code>	\supset	<code>\supset</code>
\subseteq	<code>\subseteq</code>	\supseteq	<code>\supseteq</code>	$\not\subset$	<code>\not\subset</code>	$\not\supset$	<code>\not\supset</code>
\emptyset	<code>\emptyset</code>						
\approx	<code>\approx</code>	\approx	<code>\approx</code>	\equiv	<code>\equiv</code>	\neq	<code>\neq</code>
\geq	<code>\geq</code>	\leq	<code>\leq</code>	\geq	<code>\geq</code>	\leq	<code>\leq</code>
\gg	<code>\gg</code>	\ll	<code>\ll</code>	\ngtr	<code>\ngtr</code>	\nless	<code>\nless</code>
\ngeq	<code>\ngeq</code>	\nleq	<code>\nleq</code>				

Cuadro 6.1: Letras griegas y algunos otros símbolos \LaTeX

No obstante, para una referencia mucho más amplia de símbolos se recomienda consultar cualquier libro de la bibliografía. En particular, [25] es una compilación de todos los símbolos existentes, con referencia a los paquetes que proveen los comandos correspondientes.

Antes de dejar esta sección dedicada a la edición matemática, es obligado comentar, no sólo que la cantidad de comandos relacionados es amplísima y variadísima y que lo aquí expuesto es una pequeña muestra, sino también que, además de las vistas aquí, en ocasiones hay más de una manera de conseguir el mismo resultado, de editar la misma fórmula. Recordemos también en este punto que el propósito de este manual es sólo de iniciación, y se remite de nuevo al lector interesado a las más completas referencias presentes en la bibliografía.

6.2. Objetos flotantes: tablas y figuras

A continuación veremos cuál es la manera de incluir tablas y figuras en nuestros documentos \LaTeX .

6.2.1. ¿Qué es “*flotar*”?

Antes de entrar al detalle de los comandos y entornos relevantes a la hora de abordar la inclusión de tablas y figuras en nuestros documentos, aprenderemos el concepto de **objeto flotante** en \LaTeX , puesto que podremos dar a ambos tipos de elementos esta consideración.

Para \LaTeX , un objeto flotante es un elemento cuya posición será determinada con respecto al resto de la composición. Se trata de un bloque cuyo contenido no es lo más importante, si no que lo son sus dimensiones y la manera como se maquetan dentro de una página. Son objetos con entidad propia, que no pueden cortarse para continuarse en la página siguiente.

Para el tratamiento de este tipo de objetos \LaTeX cuenta, cómo no, con un conjunto de reglas de maquetación estrictas, pero también veremos que existe la posibilidad de realizar indicaciones o expresar preferencias que queremos que sean tenidas en cuenta a la hora de aplicar dichas reglas.

6.2.2. Tablas

La edición de tablas se lleva a cabo gracias al entorno `tabular`, cuya sintaxis es extremadamente similar a la que ya veíamos en la página 68 para las matrices (comando `array` del entorno matemático):

<p>esto sólo es una simple tabla de ejemplo</p>	<pre>\begin{tabular}{rcl} esto & sólo es & una simple \\ tabla & de & ejemplo \\ \end{tabular}</pre>
---	--

Como se puede comprobar, el esquema es el mismo: el entorno recibe como argumento obligatorio el esquema de alineación de las columnas, y dentro del entorno se teclea el contenido de las celdas (separadas por un `&`) de cada fila (separadas por `\\`). Los caracteres que indican la alineación son los mismos que ya vimos (`c`, `l` o `r`).

Para dibujar líneas alrededor de celdas, filas y columnas distinguiremos entre las líneas verticales y las horizontales. Las primeras son las más inmediatas de indicar, incluyendo símbolos `|` entre los caracteres de alineación de las columnas que deseemos. Por su parte, las líneas horizontales se consiguen con los comandos `\hline` y `\cline{rangoColumnas}`, colocados en la fila correspondiente:

<table border="1" style="border-collapse: collapse; width: 100%; text-align: center;"> <tr> <td style="padding: 5px;">esto</td> <td style="padding: 5px;">sólo es</td> <td style="padding: 5px;">una simple</td> </tr> <tr> <td style="padding: 5px;">tabla</td> <td style="padding: 5px;">de</td> <td style="padding: 5px;">ejemplo</td> </tr> </table>	esto	sólo es	una simple	tabla	de	ejemplo	<pre>\begin{tabular}{r c l } \cline{2-3} esto & sólo es & una simple \\ \hline tabla & de & ejemplo \\ \hline \hline \end{tabular}</pre>
esto	sólo es	una simple					
tabla	de	ejemplo					

Otro comando útil es `\multicolumn`, que nos permite *fundir* las celdas de una fila correspondientes a varias columnas en sólo una o, lo que es lo mismo,

hacer que una celda ocupe el lugar de varias, extendiéndose horizontalmente para abarcar varias columnas. Del mismo modo, incluyendo el paquete `multirow` en el preámbulo del documento, dispondremos también del comando `\multirow`, para hacer lo propio en lugar de en horizontal, en vertical (extendiendo una celda a varias filas de su misma columna):

esta tabla	sólo es	un simple
	ejemplo	

```

\begin{tabular}{r|c|l|}
\cline{2-3}
\multirow{2}{2cm}{esta tabla}
& sólo es & un simple \\
\cline{2-3}
& \multicolumn{2}{c}{ejemplo} \\
\hline \hline
\end{tabular}

```

6.2.2.1. Tablas flotantes

El entorno `tabular` no es de por sí un entorno flotante, sino un entorno común y corriente, como otros que hemos aprendido hasta ahora. El entorno con propiedades de objeto flotante correspondiente a las tablas es el entorno `table`.

Si añadimos a nuestros documentos tablas editadas simplemente tal y como hemos visto, corremos el riesgo de que se corten si están muy abajo en la página, no podremos colocarles un comentario de pie de tabla, no aparecerán en un índice de tablas. . . Para conseguir estos beneficios es necesario incluir el entorno `tabular` a su vez dentro de un entorno `table`, de la siguiente manera:

esta tabla	sólo es	un simple
	ejemplo	

Cuadro 6.2: Tabla de prueba

```

\begin{table}[hbt!]
\centering
\begin{tabular}{r|c|l|}
\cline{2-3}
\multirow{2}{2cm}{esta tabla} & sólo es & un simple \\
\cline{2-3}
& \multicolumn{2}{c}{ejemplo} \\
\hline \hline
\end{tabular}
\caption{Tabla de prueba}
\end{table}

```

Como vemos, el entorno `table` recibe un argumento opcional, cuya función es muy similar al argumento obligatorio de los entornos `array` o `tabular`. Se trata de un conjunto de caracteres, que indican a \LaTeX las preferencias de colocación del elemento flotante. En este caso, el número de caracteres no se corresponde con columnas, claro, si no que expresa diferentes posibilidades de ubicación entre las que escoger, ordenadas por prioridad. Las opciones son:

- h** indica como lugar preferido el mismo lugar que en el código fuente ocupa el entorno `table` (del inglés, *here*)
 - b** prefiere la tabla colocada en la parte inferior de una página con texto (*bottom*)
 - t** prefiere la tabla colocada en la parte superior de una página con texto (*top*)
 - p** indica como lugar preferido una página integrada sólo por objetos flotantes (*page*)
 - !** sugiere a \LaTeX que sea un poco más flexible en sus consideraciones para poder ocupar con mayor probabilidad un lugar preferido con mayor prioridad
-

Así, en la tabla anterior, la secuencia `[hbt!]` indica que nuestra primera preferencia es que el objeto se quede en el lugar en el que se ubica en el código fuente; de no ser posible, recomendamos que se pegue a la parte inferior de la página y en su defecto a la parte superior (de ésta o de la página siguiente); como última opción, solicitamos la colocación en una página dedicada exclusivamente a objetos flotantes (además, le indicamos a \LaTeX que sea algo permisivo en sus decisiones). En caso de no especificarse este argumento opcional, \LaTeX aplicará sus criterios, intentando desperdiciar el menor espacio posible a la par que conseguir la mejor maquetación estética.

En el ejemplo anterior hemos incluido otro par de comandos nuevos. El primero de ellos es la orden `\centering`. Como su propio nombre indica, consigue que el contenido del entorno `table` se centre con respecto a los márgenes de la página (por defecto, se alinearía a la izquierda). El otro comando novedoso es `\caption{Leyenda}`, que nos sirve para dos cosas: la primera y evidente es dotar de un comentario a pie o *Leyenda* a la tabla. El segundo, no tan evidente pero no menos útil, es la posibilidad de que la tabla en cuestión aparezca en el **índice de tablas**, pues toda tabla con leyenda aparece y una tabla sin leyenda no lo hará. El índice de tablas se consigue de manera muy similar al índice de contenidos, gracias a la orden `\listoftables`, que normalmente se colocará o bien al final del documento, o bien en el mismo lugar que el comando `\tableofcontents`.

6.2.3. Imágenes y gráficos

Para incluir imágenes y gráficos en nuestros documentos \LaTeX , lo primero que debemos hacer es declarar el paquete `graphicx`. Éste nos proporcionará todos los comandos necesarios para la tarea, que veremos en esta sección.

Los tipos de gráficos que \LaTeX admite dependen de la herramienta con que vayamos a trabajar paralelamente:

DVIPS Si compilamos nuestro documento con `latex` y utilizamos esta herramienta para transformar el resultado a formato Postscript,

los formatos gráficos que podremos utilizar son: **ps** (Postscript), **eps** (Enhanced Postscript), **pcx** (Paintbrush Bitmap Graphic), **bmp** (Bitmap). En el caso de los bitmaps, deberemos indicar las dimensiones de la figura obligatoriamente, de la manera que veremos.

DVIPDF En caso de que utilicemos **dvipdfm**, podremos incluir: **jpg** (Joint Photographic Group), **jpeg** (Joint Photographic Experts Group), **png** (Portable Network Graphic), **pdf** y también **ps** y **eps**.

PDFLATEX Si la compilación se lleva a cabo usando **pdflatex**, los formatos admitidos son: **jpg**, **jpeg**, **tif** (Tagged Image Format), **tiff** (Tagged Image File Format), **png** y **pdf**.

El comando utilizado para incluir gráficos o imágenes es el siguiente:



```
\includegraphics[width=2cm]{imagenes/ejemplo.eps}
```

Este comando tiene varias opciones, a saber:

width=*longitud* Usada en el ejemplo anterior, indica la anchura con la que se mostrará la imagen, que puede coincidir o no con su anchura real. **L^AT_EX** se encarga de hacer el escalado y de mantener las proporciones si no indicamos la opción **height**. El valor *longitud* puede ser un entero o un decimal, siempre acompañado de una unidad de medida, por ejemplo **cm**.

height=*longitud* De modo análogo a la opción **width**, se usa para indicar la altura que queremos que tenga la imagen.

scale=*valor* Indica el factor de escala que **L^AT_EX** ha de aplicar a la imagen, donde *valor* será un número decimal entre 0 y 1.

angle=valor Podemos indicar a L^AT_EX que rote la figura. Si *valor*, que ha de ser un entero entre 0 y 360, es positivo, la rotación se hará en el sentido contrario a las agujas del reloj. Si el entero es negativo, el sentido de la rotación será el inverso.

6.2.3.1. Figuras y gráficos flotantes

El equivalente al entorno `table` para tablas es el entorno `figure` para gráficos e imágenes. Especificando el comando `\includegraphics` dentro de él, haremos de nuestras imágenes objetos flotantes:

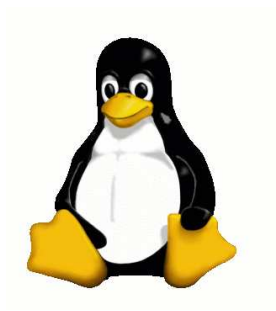


Figura 6.1: Imagen de ejemplo

```
\begin{figure}[hbt!]
\centering
\includegraphics[height=4cm]{imagenes/ejemplo.eps}
\caption{Imagen de ejemplo}
\end{figure}
```

Los comentarios hechos en el apartado 6.2.2.1 son igualmente aplicables aquí. Asimismo, la inclusión del comando `\caption` dentro del entorno `figure` consigue que en la lista generada por la orden `\listoffigures` aparezca la reseña de la imagen en cuestión.

6.3. Cartas

Para terminar este capítulo referido a la edición especial de documentos, comentaremos brevemente un tipo de documento que mencionábamos en el capítulo 3, pero cuyo tratamiento postponíamos hasta este momento.

El tipo de documento `letter` cumple una función específica, la redacción de cartas, y la declaración `\documentclass{letter}` pone a nuestra disposición no sólo el entorno `letter`, donde residirá el cuerpo de la carta, sino una serie de comandos especiales para editarla, además de realizar otros ajustes de formato.

El entorno `letter` recibe como argumento obligatorio la dirección del destinatario. El resto de comandos relevantes se expone a continuación:

`\opening{Texto}` Se utiliza para especificar el saludo de la carta (*Texto*).

`\closing{Texto}` De manera complementaria al anterior, se utiliza para indicar la despedida. Sólo tras haber empleado este comando pueden usarse:

`\ps{Texto}` Con esta orden pueden indicarse una o más posdatas.

`\cc{Texto}` Así indicamos la lista de gente que recibe copia de la misiva.

`\encl{Texto}` Así podemos listar los adjuntos que acompañan a la carta.

`\signature{Texto}` Se usa para indicar el nombre, posición, etc. de quien suscribe y firma.

```
\documentclass{letter}
\usepackage[latin1]{inputenc}
\begin{document}
\begin{letter}{Summer School -- Institut für Informatik\\
               Technische Universität
               München\\
               Boltzmannstr. 3\\
               85748 Garching (München)}

\opening{To Whom it May Concern:}

Please have the attached document in order to take into
consideration my application for the Summer School
Marktoberdorf 2004.

\signature{Laura M. Castro}

\closing{Sincerely,}

\end{letter}
\end{document}
```

Cuadro 6.3: Ejemplo de carta en L^AT_EX (código fuente)

October 15, 2004

Summer School – Institut für Informatik
Technische Universität München
Boltzmannstr. 3
85748 Garching (München)

To Whom it May Concern:

Please have the attached document in order to take into consideration my application for the Summer School Marktoberdorf 2004.

Sincerely,

Laura M. Castro

Figura 6.2: Ejemplo de carta en L^AT_EX

6.4. Ejercicios

Ejercicio 6.1 Escribir el siguiente texto:

Se trata de poner unos ejemplos en *modo texto*. Una raíz con valor absoluto: $\sqrt{|x|}$; notación vectorial: $\vec{x} = x^i \vec{v}_i$ y notación matricial: $C = F^t g F$ o bien $[\vec{x}]_B = \mathbf{P}[\vec{x}]_{B'}$.

Ejercicio 6.2 Escribir el siguiente texto:

Se trata de un ejemplo en donde se combina el MODO PÁRRAFO con texto.

$$\frac{|x|^3}{x^2 + y} \text{ en la misma línea viene la serie: } \sum_{n=2}^{\infty} \frac{1}{\sqrt[3]{n^2 - 1}}$$

Ejercicio 6.3 Escribir:

Hay sumatorios que se pueden escribir dentro del *texto* en *modo párrafo*. Por ejemplo: $d^2 f(\vec{a})(\vec{x}, \vec{y}) = \sum_{i,j=1} f_{ij}(\vec{a}) x_i y_j$ y otras veces se escribe directamente en un *párrafo* distinto es decir:

$$\sqrt[n]{\sum_i \sum_j S^{ij} T_{ij}} \quad (6.2)$$

Ejercicio 6.4 Escribir:

Es posible combinar el **entorno científico** con las **listas**.

1. $[\vec{x}] = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}$ o $[\vec{y}] = \begin{Bmatrix} y_1 \\ \cdots \\ y_m \end{Bmatrix}$ todo esto escrito en *modo texto*
- 2.

$$T(\vec{x}, \vec{y}) = \left\{ \begin{matrix} x_1 & \cdots & x_m \end{matrix} \right\} \begin{bmatrix} a_1^1 & \cdots & a_n^1 \\ \vdots & \ddots & \vdots \\ a_1^m & \cdots & a_n^m \end{bmatrix} \begin{pmatrix} y_1 \\ \cdots \\ y_n \end{pmatrix}$$

Este punto se comienza directamente con el **modo párrafo**.

3. *Rizamos el rizo* en cuestión de matrices:

$$[A_k^{ij}] = \left\{ \begin{array}{c} \begin{bmatrix} a & b \\ c & d \end{bmatrix} \\ \begin{pmatrix} x^2 & y \\ z & \sqrt{u} \end{pmatrix} \end{array} \right\}$$

Ejercicio 6.5 Escribir lo siguiente:

Otros ejemplos son:

- La Transformada de Laplace de una función real f viene dada por:

$$\mathfrak{F}(s) = \mathcal{L}|f| = \int_0^{+\infty} e^{-sx} f(x) dx$$

- La *Delta de Dirac* en x_0 se puede definir a partir de:

$$\delta_a(x - x_0) = \begin{cases} \frac{1}{2a} & \text{si } x_0 - a < x < x_0 + a \\ 0 & \text{si } x \leq x_0 - a \text{ o bien } x \geq x_0 + a \end{cases}$$

siendo: $\delta(x - x_0) = \lim_{a \rightarrow 0} \delta_a(x - x_0)$

Ejercicio 6.6 En la misma dirección que el ejercicio anterior, tratar de desvelar el texto fuente de:

$$\int_a^b f(x) dt \approx \sum_{i=1}^n f(x_i) w_i \left\{ \begin{array}{l} \int_a^b f(x) dt \approx \\ \sum_{i=1}^n f(x_i) w_i \end{array} \right.$$

Ejercicio 6.7 En ocasiones resulta interesante combinar texto habitual con el modo científico. Tratar de hacer el siguiente esquema:

Se puede establecer la siguiente clasificación:

Fundamentos	Números Reales	<ul style="list-style-type: none"> Introducción Definición 	10 Horas
	F. Reales	<ul style="list-style-type: none"> Definiciones Ejemplos 	
	Límites	<ul style="list-style-type: none"> Definición Propiedades 	
	Continuidad		

Ejercicio 6.8 Vamos a probar algunas opciones del comando `includegraphics`:

Vamos a ver:

1. Cómo se deforma una figura
2. Cómo se rota una figura

El resultado se puede ver en las siguientes figuras:



Figura 6.3: Un pingüino deformado



Figura 6.4: Un pingüino tumbado

Ejercicio 6.9 Construir la tabla 6.4.

Ejercicio 6.10 Reproducir la carta que aparece en la página 82.

Distribución aproximada de tiempos			
	CAPÍTULO	TIEMPO	UNIDADES DIDÁCTICAS
1.	Integral definida.	8 horas.	III – Integración 35 h.
2.	Propiedades de la integral.	4 horas.	
3.	Teoremas fundamentales.	5 horas.	
4.	Cálculo de primitivas.	7 horas.	
5.	Aplicaciones de la integral definida.	6 horas.	
6.	Integrales impropias.	5 horas.	
7.	Sucesiones.	3 horas.	IV – Series 31 h.
8.	Series numéricas.	8 horas.	
9.	Series funcionales.	9 horas.	
10.	Series de potencias.	6 horas.	
11.	Series de Fourier.	5 horas.	
Total: 66 horas			

Cuadro 6.4: Una tabla completa

Capítulo 7

Referencias internas

Índice general

7.1. Referencias básicas	87
7.2. Bibliografía	90
7.2.1. BibT _E X	91
7.3. Índice de materias	94
7.4. Ejercicios	96

EN este capítulo veremos todo lo relativo a las referencias internas que podemos encontrarnos en un documento, desde simples indicaciones hasta bibliografía o índices de materias.

7.1. Referencias básicas

Las referencias son útiles para relacionar partes de un documento. Además, en L^AT_EX son una herramienta particularmente útil y cómoda, ya que simplemente tenemos que marcar los sitios o elementos susceptibles de ser referenciados y los lugares desde donde se les quiere hacer referencia. Del resto de pormenores se encarga automáticamente el compilador: si movemos los elementos referenciados o

cambiamos las referencias de lugar, bastará con recompilar para que se actualicen las reseñas afectadas.

Así pues, para utilizar referencias necesitamos dos tipos de elementos: una manera de “marcar” los puntos u objetos a los que nos vamos a referir, y una forma de hacer referencia a dichas marcas:

Etiquetas Para marcar elementos referenciables se utiliza el comando

`\label{Etiqueta}`

que establece un punto de referencia o bien etiqueta un elemento. Se puede etiquetar:

- Un elemento de una lista numerada, colocando el comando `\label` en cualquier lugar tras el `\item` pertinente (es decir, no tiene por qué ir inmediatamente después).
- Un elemento flotante, colocando el comando `\label` dentro del entorno flotante (`figure` o `table`), siempre *después* del comando `\caption`.
- Una ecuación, colocando el comando `\label` en algún lugar dentro del entorno `equation`.
- Una división de un documento (capítulo, sección, subsección, etc.); siempre que se coloque una etiqueta y no sea uno de los supuestos anteriores, la etiqueta se referirá a la división más especializada activa en ese lugar.

Referencias Hay dos tipos de referencias que pueden hacerse con respecto a un elemento o lugar etiquetado:

Referencia al objeto Gracias al comando

`\ref{Etiqueta}`

Al compilar, \LaTeX sustituirá en el documento final el comando `\ref` por el número del capítulo, sección, tabla, figura o ítem de una lista numerada que esté etiquetado con el nombre *Etiqueta*.

Referencia a la página del objeto Si en lugar de hacer referencia al objeto en sí deseamos que aparezca el número de página en el que se encuentra, utilizaremos el comando

`\pageref{Etiqueta}`

Si a medida que el documento crezca el número de página cambiase por alguna razón, \LaTeX lo solucionaría en el mismo proceso de compilación.

Es importante recordar que las etiquetas deben ser *únicas* dentro del documento. Suele resultar útil emplear nombres descriptivos que identifiquen unívocamente el elemento al que se asocia la etiqueta. No en vano dos de los errores más comunes (ver *Errores en \LaTeX* , página 107) a este respecto son la existencia de etiquetas *duplicadas* (dos o más etiquetas iguales en el documento) o la no existencia de una etiqueta (porque en los comandos `\ref` o `\pageref` se escribe mal o porque realmente nos hemos olvidado de incluir el comando `\label` correspondiente).

Se recomienda etiquetar las unidades de estructura de los documentos con nombres fácilmente recordables, sencillos (no pueden contener caracteres reservados), no muy largos, y, como decimos, únicos. Esta es una costumbre recomendable, aunque por supuesto también se pueden ir colocando cuando, donde y a medida que se necesiten.

La generación de referencias es una de las cosas que obliga a compilar un documento \LaTeX más de una vez. En la primera pasada se recopila información sobre las etiquetas y su ubicación (a medida que se encuentran), mientras que es necesaria una segunda para “sustituir” los comandos de referencia por el valor correspondiente, calculado en la primera pasada.

A modo de curiosidad, si en lugar del número de página o el que identifica al elemento deseásemos obtener el *nombre* del elemento (obviamente esto se restringe a unidades estructurales y elementos flotantes), el paquete `titleref` proporciona el comando `\titleref{Etiqueta}`, que se sustituye por el título del elemento etiquetado con el nombre *Etiqueta*.

7.2. Bibliografía

La forma más sencilla de incluir bibliografía en nuestros documentos \LaTeX consiste en utilizar el entorno `thebibliography`:

```
\begin{thebibliography}{ZZ}
\bibitem{libroLatex}
  Bernardo Cascales Salinas et al. \\\
  {\itshape El libro de \LaTeX{}}. \\\
  Prentice Hall, 2004.
\bibitem{iniciacionLatex}
  Javier Sanguino Botella. \\\
  {\itshape Iniciación a \LaTeXe{}}. \\\
  Addison-Wesley, 1997.
...
\end{thebibliography}
```

Este entorno se imprime en el lugar donde se coloca, por lo que lo habitual será que se ubique al final del documento, justo antes del `\end{document}`. Como vemos, el entorno `thebibliography` recibe un argumento obligatorio, que es una secuencia de caracteres que indica a \LaTeX la longitud máxima de las etiquetas que se utilizarán en dicha bibliografía.

Después, el entorno se organiza en *ítems*, uno por cada comando `\bibitem`, que al estilo de los `\item` de las listas, marca el comienzo de cada nuevo elemento. Este comando también recibe un argumento obligatorio, en este caso la etiqueta que identifica a la referencia bibliográfica en cuestión y que se utilizará en el resto del documento, donde sea relevante hacer referencia a dicha entrada de la bibliografía. Esto se lleva a cabo mediante el comando

```
\cite{Etiqueta}
```

que funciona exactamente igual que los comandos `\ref` o `\pageref`. En cuanto al contenido o formato de cada entrada, queda totalmente a criterio del autor.

7.2.1. BibTeX

El uso directo del entorno `thebibliography` para la inclusión de bibliografía en documentos \LaTeX deja al autor completa libertad para definir el formato de ésta, lo que supone también que sobre él recae toda la responsabilidad al respecto. Esto, totalmente contrario a la filosofía de $\text{\TeX}/\text{\LaTeX}$ de conseguir que el que escribe no se preocupe por el formato sino sólo por el contenido, es la mayoría de las veces más un engorroso inconveniente que una verdadera ventaja. Más aún si pensamos en reordenar los ítems, en mantener la consistencia y coherencia entre los mismos, etc.

Como alternativa, contamos con la herramienta BibTeX [2], creada por Oren Patashnik. BibTeX es un complemento para \LaTeX que permite manejar bases de datos bibliográficas y que se integra perfectamente con él para generar automáticamente un entorno `thebibliography` siguiendo una serie de reglas de formato establecidas, es decir, una vez más liberando al usuario precisamente de la parte mecanizable y repetitiva.

Lo primero que hay que hacer para usar BibTeX es definir una *base* con las citas bibliográficas que consideremos relevantes. La información bibliográfica se encontrará, pues, separada del resto de nuestro trabajo (lo que nos permitirá, además, reutilizarla con facilidad en diferentes documentos), y seguirá una estructura especial. Es decir, tendremos que editar un simple fichero de texto con extensión `.bib` con un formato concreto: una serie de “registros” especiales, uno por referencia bibliográfica a incluir.

Hay varios tipos de registros posibles, que se corresponden con los distintos tipos de referencias existentes. Los más habituales son:

```
@BOOK{LibroLatex,
  author    = {{Cascales Salinas, Bernardo} and others},
  publisher = {ADI},
  title     = {\LaTeX{}}, una imprenta en sus manos},
  year      = {2000}
}
```

```

@ARTICLE{armistice,
  author   = {{Cabrero, David}, {Abalde, Carlos},
             {Varela, Carlos} and {Castro, Laura}},
  title    = {ARMISTICE: An Experience Developing
             Management Software with Erlang},
  journal  = {Principles, Logics, and Implementations of
             High-Level Programming Languages (PLI'03)},
  month    = {Agosto}
  year     = {2003}
}

@MANUAL{shortLatex,
  title    = {The not so short introduction to \LaTeXe},
  author   = {Oetiker, Tobias}
}

@MISC{faq,
  title          = {FAQ de CervanTeX},
  howpublished   = {\texttt{www.tug.org/tex-archive/help/
                    es-tex-faq/FAQ-CervanTeX.pdf}}
}

```

Tras haber definido el fichero de bibliografía siguiendo este formato, su uso se indica incluyendo las siguientes líneas en algún punto del cuerpo del documento (en el lugar donde queramos que aparezca la bibliografía, que será normalmente al final):

```

\bibliography{ficheroibase}
\bibliographystyle{estilo}

```

donde *ficheroibase.bib* es el nombre del fichero de la base de datos (no hay necesidad de especificar la extensión *.bib*) y *estilo* es uno de los siguientes:

plain

Muestra las entradas de la bibliografía ordenadas alfabéticamente y numeradas. Para cada entrada, muestra la información en el orden: autor, año, título.

unsrt

Se comporta igual que **plain** pero muestra las entradas por orden de citación en el documento, en lugar de alfabéticamente.

alpha

En este caso, en vez de numerar las entradas como **plain** y **unsrt**, se les asigna una etiqueta basada en el nombre del autor y el año de publicación. El orden para la información que se muestra es: etiqueta, autor, año y título.

abbrv

Equivalente a **plain**, maneja entradas más pequeñas al abreviar los nombres de los autores, los meses y los nombres de las revistas.

Una vez hecho esto, es necesario procesar la bibliografía utilizando la herramienta **bibtex**. Este programa recibe como argumento un fichero **.aux** procedente de una primera compilación de nuestro documento **L^AT_EX**. En dicho fichero auxiliar, entre otras cosas, se identifican todas las citas bibliográficas a las que se ha hecho referencia en el documento sobre el que se trabaja. Tras el procesado con **bibtex**, pues, será necesaria una nueva compilación con **L^AT_EX** para que queden resueltas las citas:

```
latex documento.tex
bibtex documento.aux
latex documento.tex
```

Nótese que el fichero que se pasa a **bibtex** es el **.aux** generado tras la compilación de **documento** y no el fichero de la base de datos en sí (éste será localizado por la herramienta al haber sido incluido en el anterior usando el comando **\bibliography**).

Puesto que lo que hace Bib_T_EX es generar un entorno `thebibliography` automáticamente, las citas se incluyen en el texto igualmente utilizando el comando `\cite{Etiqueta}`, con la particularidad de que sólo aquéllos documentos de la base bibliográfica que sean mencionados en un `\cite` se visualizarán en la versión compilada del documento (y no todos los presentes en el fichero `.bib`, por defecto). Si queremos que se visualice algún documento no referenciado, tendremos que indicarlo con el comando `\nocite{Etiqueta}`, y si queremos que aparezcan todos los elementos presentes en la base de datos, emplearemos `\nocite*`.

7.3. Índice de materias

L_AT_EX es capaz de generar automáticamente índices de materias a medida que creamos nuestros documentos. Para ello, en primer lugar es necesario utilizar el paquete `makeidx`, e incluir en el preámbulo del documento el comando

```
\makeindex
```

Durante la edición del documento, en el momento en que queramos indexar algún término, utilizaremos la orden:

```
\index{término}
```

Hay distintos tipos de entradas posibles:

entradas simples Son las que se generan con el comando `\index`, tal y como acabamos de ver. Producirán una entrada en el índice de materias con el término indicado y la referencia a la página correspondiente al punto del documento donde se escribe el comando en el código fuente.

subentradas Producen una entrada de nivel inferior, concretando o especializando normalmente una entrada más general. Se consiguen de la siguiente manera:

```
\index{término}
\index{término!subtérmino}
\index{término!otro subtérmino}
\index{término!subtérmino!un subtérmino de segundo nivel}
```

Sólo pueden crearse dos subniveles de entradas en el índice.

referencias a otras entradas Para conseguir que una entrada nos envíe a otra (el consabido *véase...*), o que haga referencia a otra como información adicional (*véase también...*), utilizaremos:

```
\index{término|see{otro término}}
\index{término|seealso{otro término}}
```

entradas con formato Si queremos que las entradas en el índice tengan formato (es decir, un estilo de letra diferente), podemos indicarlo:

```
\index{término@\emph{término}}
\index{otro término@\texttt{otro} \textbf{término}}
```

Una vez que se compila el documento \LaTeX con los comandos `index` incluidos, se generará un fichero con extensión `.idx`, conteniendo toda la información relativa al índice. Este fichero ha de ser procesado usando la herramienta `makeindex`, que ya se incluye generalmente con las distribuciones $\text{\TeX}/\text{\LaTeX}$:

```
makeindex documento.idx
```

Esto generará un nuevo fichero, con extensión `.ind`. Este fichero tiene formato \LaTeX , y para incluirlo desde nuestro documento usaremos el comando `\printindex` en el lugar donde queramos que se muestre el índice (que será, normalmente, al final del documento). Así pues, tras obtener el fichero `.ind` será necesario compilar nuestro documento \LaTeX una vez más para obtener la versión definitiva, con el índice de materias incluido.

7.4. Ejercicios

Ejercicio 7.1 Poner en práctica los conocimientos sobre referencias internas adquiridos en este capítulo:

En la sección 6.4 (página 83), sección de ejercicios del capítulo 6, poníamos a prueba algunas opciones del comando `includegraphics`, tal y como se indica en los puntos 1 y 2 de la siguiente lista:

1. Cómo se deforma una figura
2. Cómo se rota una figura

El resultado se puede ver en las figuras 7.1 y 7.2 (página 96).



Figura 7.1: Un pingüino deformado



Figura 7.2: Un pingüino tumbado

Esta fórmula 7.1 no viene a cuento pero sirve de ejemplo:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \quad (7.1)$$

También construíamos una completa tabla, la 6.4 (página 86).

Capítulo 8

Personalización

Índice general

8.1. Crear una portada propia	98
8.2. Cambiar los encabezados de página	98
8.3. Márgenes, interlineado, saltos de página y espacios .	99
8.3.1. Cambiando los márgenes	99
8.3.2. Cambiando el interlineado	99
8.3.3. Saltos de página	100
8.3.4. Tratamiento del espacio	100
8.4. Segmentación de palabras	102
8.5. Evitar la numeración de elementos	102
8.6. Listas personalizadas	103
8.7. Euro	103
8.8. Colores	103
8.9. Cajas	104

HAY quien opina que L^AT_EX es fácil de usar en tareas simples, pero que pone las cosas difíciles si se quieren cambiar detalles concretos de la apariencia de

los documentos. En este capítulo intentaremos arrojar un poco de luz al respecto.

8.1. Crear una portada propia

Hemos visto que \LaTeX genera portadas sencillas de manera automática, a partir de un conjunto de datos. No obstante, es bastante probable que queramos confeccionar una portada a nuestro gusto, y para ello disponemos del entorno `titlepage`. Colocado al principio del documento por norma general, justo tras el `\begin{document}`, tras él se efectúa automáticamente el salto de página.

Dentro de este entorno configuraremos a nuestro gusto el contenido y la apariencia de nuestras portadas. Para esta tarea, pueden ser útiles algunos de los comandos que se verán en la sección 8.3.

8.2. Cambiar los encabezados de página

Algo que también puede querer personalizarse son las cabeceras de página. Tal y como vimos en la sección 4.3 (página 36), existen varios estilos de página predefinidos. Además de ellos, contamos con un estilo de página modificable:

`myheadings` se comporta igual que `headings`, pero incorpora los comandos

```
\markright{CabeceraDerecha}  
\markboth{CabeceraIzquierda}{CabeceraDerecha}
```

Estos comandos (que se aplicarán, respectivamente en documentos con opciones `oneside` o `twoside`) permiten especificar el contenido de las cabeceras. Por defecto sus valores son:

Para esta tarea pueden ser útiles los siguientes comandos:

- `\theEstructura` (i.e. `\thechapter`, `\thesection...`) introduce el número de la *Estructura* activa en ese lugar

		markboth		markright
		<i>izquierda</i>	<i>derecha</i>	<i>derecha</i>
oneside	article y proc book y report			<i>section</i> <i>chapter</i>
twoside	article y proc book y report	<i>section</i> <i>chapter</i>	(vacío) (vacío)	<i>subsection</i> <i>section</i>

Cuadro 8.1: Contenido por defecto de las cabeceras en estilo `myheadings`

- `\Estructuraname` (i.e. `\chaptername`) introduce el rótulo de la *Estructura* activa correspondiente (i.e. “*Capítulo*”).

8.3. Márgenes, interlineado, saltos de página y espacios

A continuación veremos cómo alterar los márgenes de nuestros documentos de manera sencilla, así como la manera de forzar saltos de página y diferentes formas de tratar con espacios en blanco.

8.3.1. Cambiando los márgenes

La forma más sencilla de cambiar los márgenes de un documento es utilizar el paquete `ansize`, que nos proporciona el comando

```
\marginsize{MargenIzquierdo}{MargenDerecho}
           {MargenSuperior}{MargenInferior}
```

Utilizando una orden como `\marginsize{2cm}{2cm}{2cm}{2cm}` en el preámbulo del documento, estableceríamos todos los márgenes del mismo a 2cm.

8.3.2. Cambiando el interlineado

En este caso, el paquete en cuestión más recomendable es el llamado `setspace`. Gracias a él podremos indicar en el preámbulo órdenes como

```
\singlespacing  
\onehalfspacing  
\doublespacing
```

8.3.3. Saltos de página

Los comandos para forzar un salto de página en un determinado lugar del documento son

```
\newpage  
\clearpage
```

La diferencia entre ambos reside en que `\clearpage`, además de cambiar de página, incorporará en ese punto, si procede, una o más páginas incluyendo los elementos flotantes que estén pendientes de ser maquetados en páginas especiales. Con este mismo comportamiento también existe el comando

```
\cleardoublepage
```

que, como su nombre indica, salta dos páginas.

8.3.4. Tratamiento del espacio

Los comandos

```
\hspace{Longitud}  
\vspace{Longitud}
```

nos sirven para introducir espacios *horizontales* y *verticales* respectivamente en nuestros documentos. Además, existen también los comandos

```
\hfill  
\dotfill  
\hrulefill  
\vfill
```

que *rellenan* con todo el espacio posible en horizontal (los tres primeros) o en vertical (el último).

El espacio puede ser rellenado con “blancos” (en el caso de `\hfill` y `\vfill`), con puntos (`\dotfill`) o con una línea horizontal (`\hrulefill`). Combinando estos comandos pueden conseguirse cosas como:

Esto.....resulta de la combinación_____de comandos de relleno.

Esto `\dotfill` resulta de la
combinación `\hrulefill` de comandos de relleno.

8.4. Segmentación de palabras

Aunque si usamos la variante correcta del paquete `babel` \LaTeX segmentará correctamente las palabras en la gran mayoría de las ocasiones, es posible que se equivoque en algún caso o que deje sin segmentar alguna palabra, invadiendo el margen izquierdo.

En esos casos, para ayudar al compilador, podemos indicarle los lugares por donde puede dividir una palabra separando sus sílabas en el código fuente mediante la secuencia `\-`, de este modo: `de\ -mos\ -tra\ -ción`. Dicha secuencia no aparecerá en la versión final, es sólo una marca para el maquetador.

8.5. Evitar la numeración de elementos

En ocasión puede que queramos introducir un capítulo que no reciba numeración y, por tanto, que no aparezca en la tabla de contenidos, o una figura con leyenda pero que no aparezca en la lista de figuras (porque no es lo suficientemente relevante o por la razón que sea). Para este tipo de situaciones, \LaTeX tiene también una solución, que pasa por marcar ese tipo de elementos que no deseamos numerar con un asterisco (*):

```
\section*{Esta sección no será numerada}
\caption*{La tabla con esta leyenda no aparecerá en el índice}
```

8.6. Listas personalizadas

Además de los tres tipos de listas que veíamos en la sección 5.3, existe un tipo de lista personalizable, denominada `list`. Este tipo de listas recibe dos argumentos obligatorios: el primero de ellos especifica el símbolo que se dibujará delante de cada ítem y el segundo de ellos puede recibir declaraciones y otro tipo de comandos, pero generalmente permanecerá vacío:

♣ lista	<code>\begin{list}{\clubsuit}{}{}</code>
♣ con símbolo	<code>\item lista</code>
♣ con símbolo	<code>\item con símbolo</code>
♣ personalizado	<code>\item personalizado</code>
	<code>\end{list}</code>

Para un buen repertorio de símbolos, puede consultarse cualquier referencia de la bibliografía, pero en especial [25].

8.7. Euro

Para disponer del comando `\euro`, que nos proporciona el símbolo €, hemos de incluir el paquete `eurosym`.

8.8. Colores

Gracias al paquete `color` podremos utilizar comandos como

texto de color	<code>{ \color{blue} texto de color }</code>
caja de color	<code>\colorbox{green}{caja de color}</code>
caja de color con borde	<code>\fcolorbox{red}{yellow}</code> <code>{caja de color con borde}</code>

con una serie de colores predefinidos (`white`, `black`, `red`, `blue`, `green`, `cyan`, `magenta`, `yellow`).

Además, usando el comando

```
\definecolor{nombreColor}{rgb|cmyk}{codificación}
```

pueden definirse nuevos colores utilizando los esquema RGB ó CMYK, donde `codificación` son 3 ó 4 números, respectivamente, entre 0 y 1.

8.9. Cajas

Para resaltar un párrafo de texto, es un buen recurso utilizar recuadros o **cajas** que lo encierren. En \LaTeX se dispone del comando

```
\fbox{Contenido}
```

Además, utilizando el paquete `fancybox` se pueden conseguir otros diseños, como:

```
\shadowbox{Contenido}
```

```
\doublebox{Contenido}
```

```
\ovalbox{Contenido}
```

```
\Ovalbox{Contenido}
```

Parte II

Apéndices

Apéndice A

Errores en L^AT_EX

Índice general

A.1. No te olvides de cerrar	107
A.2. Cada cosa en su lugar	109
A.3. Cuidado con esas tablas	110
A.4. Ojo a lo que escribimos	111
A.5. Indicar siempre las medidas	113
A.6. Lo que no se puede hacer	114
A.7. Advertencias	115

TAN importante como aprender cómo trabajar con L^AT_EX y el modo de hacerle indicaciones, es encontrar y saber interpretar los errores que podamos cometer en el proceso. Este apéndice expone algunos de los fallos más comunes.

A.1. No te olvides de cerrar

Uno de los errores más frecuentes es el desbalanceo de llaves o entornos, o en el caso de éstos últimos, el cerrarlos en un orden inadecuado. En el primero de

los casos (que ocurrirá normalmente dentro de entornos matemáticos), L^AT_EX nos advierte de la situación:

```

                                ! Extra }, or forgotten $.
\sum_{i=0}^n a_i$              1.21 $\sum_{i=0}^n a_i$
                                $

```

En el caso de los entornos, veremos el mensaje:

```

\begin{itemize}
\item 2 cucharadas de azúcar
\item 150 gr. de harina
\item 0.5 l. de leche
\end{enumerate}

! LaTeX Error: \begin{itemize} on input line 21
                    ended by \end{enumerate}.

```

O, si nos olvidamos por completo de cerrarlo:

```

\begin{itemize}
\item 2 cucharadas de azúcar
\item 150 gr. de harina
\item 0.5 l. de leche

Mezclamos la harina con...

! LaTeX Error: \begin{itemize} on input line 52
                    ended by \end{document}.

```

donde `input line` es la línea del fichero fuente en la que L^AT_EX detecta el fallo.

También puede ocurrir que lo cerremos más de una vez:

```
\begin{itemize}
\item 2 cucharadas de azúcar
\item 150 gr. de harina
\item 0.5 l. de leche
\end{itemize}
\end{itemize}
Mezclamos la harina con...

! LaTeX Error: \begin{document} ended by \end{itemize}.

See the LaTeX manual or LaTeX Companion for explanation.
Type H <return> for immediate help.
...
```

1.73 \end{itemize}

En caso de que estemos manejando varios ficheros, para identificar en cuál de ellos se halla la posible anomalía deberemos fijarnos en las líneas anteriores a la del propio fallo, retrocediendo hasta que encontremos el nombre del archivo que se estaba escaneando cuando se produjo el error:

```
[80] [81] [82] (./errores.tex
```

A.2. Cada cosa en su lugar

Otro error muy común se produce al utilizar, sin darnos cuenta, comandos especiales del entorno matemático (sobre todo símbolos) fuera del mismo, olvidándonos de indicar el cambio de modo. Esto produce el siguiente comportamiento:

Al levantar la vista,	! Missing \$ inserted.
sólo vio una enorme	<inserted text>
\Omega tallada en la	\$
fría roca...	1.33 \Omega

Más cosas que podemos por error colocar en sitios indebidos son comandos que deben ir en el preámbulo, como por ejemplo `\usepackage{paquete}`:

```
! LaTeX Error: Can be used only in preamble.
```

```
See the LaTeX manual or LaTeX Companion for explanation.
Type H <return> for immediate help.
...
```

1.58 ...el preámbulo, como por ejemplo `\usepackage`
`{paquete}`:

A.3. Cuidado con esas tablas

Las tablas son lugares particularmente proclives a la generación de fallos. Hemos de poner especial atención a los separadores (`&`), concretamente a que indicamos el número de ellos correcto según las columnas que hemos especificado:

```
\begin{tabular}{ccc}
Año 2002 & Año 2003 & Año 2004 & Año 2005 \\
\end{tabular}
```

```
! Extra alignment tab has been changed to \cr.
<recently read> \endtemplate
```

1.124 Año 2002 & Año 2003 & Año 2004 &
Año 2005 \\\

Recordemos además que `&` es un carácter reservado, por lo que su utilización en medio del texto sin escaparlo produce el error:

```
! Misplaced alignment tab character &.
1.139 ...zación en medio del texto sin escaparlo &
                                         produce
```

A.4. Ojo a lo que escribimos

Si durante la compilación L^AT_EX se encuentra con algún comando u orden que esté mal escrito o que no hayamos definido, en suma, que no pueda reconocer, nos lo advertirá de la siguiente manera:

```
hay que tener cuidado al      ! Undefined control sequence
escribir las órdenes \Latex  1.42 \Latex
```

Si se trata de un entorno:

```
\begin{descripcion}
\item [oxígeno] principal componente...
\item [nitrógeno] gas venenoso...
\item [argón] gas noble...
\end{descripcion}
```

```
! LaTeX Error: Environment descripcion undefined.
```

```
See the LaTeX manual or LaTeX Companion for explanation.
Type H <return> for immediate help.
...
```

```
1.109 \begin{descripcion}
```

Si lo que escribimos mal es el nombre de una etiqueta en alguna referencia, no obtendremos un error sino una advertencia (*warning*), del siguiente estilo:

esto pasa al incluir una	LaTeX Warning: Reference
<code>\ref{inexistente}</code>	‘inexistente’ on page 110
esto pasa al incluir una ??	undefined on input line 59.

donde el número de página se refiere a la numeración de las propias páginas del documento. Como vemos, en el documento final la referencia que no se ha podido resolver aparecerá resaltada como un par de interrogaciones. Además, por si el documento es grande y la advertencia se pierde entre el resto de los mensajes de forma que al terminar la compilación queda fuera de nuestra vista, líneas antes de finalizar \LaTeX advierte de manera general:

LaTeX Warning: There were undefined references.

lo que debería llevarnos a revisar el *log*. Algo similar ocurre con las citas bibliográficas. También es posible, por el contrario, que en lugar de no definir una etiqueta que usamos o emplear una referencia a una etiqueta que no existe, incluyamos dos etiquetas idénticas en distintas partes del documento. En ese caso, los avisos de \LaTeX serán:

```
\label{etiqueta}
...
\label{etiqueta}

LaTeX Warning: Label etiqueta multiply defined.
...
LaTeX Warning: There were multiply-defined labels.
```

Por si no recordamos por qué necesitamos compilar varias veces un documento \LaTeX (explicación en la sección 2.1.1), el propio compilador nos lo reiterará en los casos necesarios con comentarios como:

```
LaTeX Warning: Label(s) may have changed.  
Rerun to get cross-references right.
```

Por último, si en un `\input` (por ejemplo, `\input{noexiste}`) se le indica un nombre de fichero que L^AT_EX no es capaz de localizar, la compilación se detendrá con el siguiente mensaje:

```
! LaTeX Error: File 'noexiste.tex' not found.
```

```
Type X to quit or <RETURN> to proceed,  
or enter new name. (Default extension: tex)
```

```
Enter file name:
```

Sin embargo, si la inclusión se realiza con el comando `include` y el fichero no se encuentra, simplemente obtendremos un aviso:

```
No file noexiste.tex.
```

Y la compilación procederá normalmente.

A.5. Indicar siempre las medidas

Si en una figura nos olvidamos de indicar la unidad de medida en alguno de los argumentos opcionales relativos a longitudes, L^AT_EX protestará:

```
\includegraphics[width=5]{imagenes/ejemplo.eps}  
  
! Illegal unit of measure (pt inserted).  
<to be read again>  
          \relax  
1.261 ...degraphics[width=5]{imagenes/ejemplo.eps}
```

Si por el contrario, lo que está mal expresado es la propia longitud, el error será distinto:

```
\includegraphics[width=cm]{imagenes/ejemplo.eps}

! Missing number, treated as zero.
<to be read again>

cm
1.267 ...degraphics[width=cm]{imagenes/ejemplo.eps}
```

A.6. Lo que no se puede hacer

Aunque ya lo mencionábamos en el capítulo 3 (página 30), si se nos olvida que el comando `include` no puede anidarse, \LaTeX nos lo recordará:

```
! LaTeX Error: \include cannot be nested.
```

Y otra cosa que podremos intentar pero sin éxito será utilizar la secuencia `\\` para separar párrafos después de algo como un entorno:

```
...
\end{itemize}
\\
Y otra cosa que podremos intentar...
```

```
! LaTeX Error: There's no line here to end.
```

Para este tipo de situaciones debe usarse el comando `\vspace`.

A.7. Advertencias

Con bastante frecuencia, veremos avisos L^AT_EX **overfull** y **underfull** durante la compilación. El origen de este numeroso tipo de avisos está en el proceso de maquetación. Los mensajes **over** y **underfull** pueden ser relativos a la página o a una línea, y siempre significan que L^AT_EX ha tenido que ser un poco menos estricto de lo que le hubiese gustado para ajustar el contenido al espacio. En el caso de los mensajes **over** quiere decir que ha sobrepasado sus límites y en los mensajes **under**, que no ha conseguido rellenar todo el espacio sobrante como hubiera preferido.

En la práctica totalidad de las ocasiones, a pesar de estos avisos, el resultado obtenido será perfecto. No obstante, se recomienda usar la opción **draft** (véase página 30) con el fin de comprobar las ocasiones en las que la “licencia” que se ha tomado el compilador invade realmente los márgenes de manera apreciable, por ejemplo.

Apéndice B

Presentaciones con L^AT_EX

Índice general

B.1. Entorno <code>slide</code>	117
B.2. Una herramienta sencilla: <code>prosper</code>	118
B.3. Una herramienta potente: <code>beamer</code>	119

DESPUÉS de habernos acostumbrado a las bondades de L^AT_EX, es normal que nos preguntemos si, además de presentar una impecable memoria o informe hecho utilizando esta herramienta, podemos emplearla también para elaborar una presentación. La respuesta es afirmativa, y en en las siguientes páginas veremos cómo hacerlo.

B.1. Entorno `slide`

La primera aproximación que surgió en el mundo T_EX/L^AT_EX para elaborar transparencias fue la creación de un nuevo tipo de documentos: `slides`. En este tipo de documento, está disponible el entorno `slide`, cuyo contenido representa una transparencia y cuyo argumento obligatorio incluye definiciones de distinta índole y generalmente permanece vacío. De este modo, podían elaborarse docu-

mentos con múltiples entornos `slide`, rellenos a gusto del autor.

B.2. Una herramienta sencilla: prosper

Pese a cumplir con su propósito básico, la flexibilidad del tipo de documento `slides` es realmente escasa y los resultados, pobres. Es por ello que rápidamente surgen diferentes paquetes y herramientas para tratar de poner solución a esta cuestión. Entre ellas, elegimos **Prosper** como recomendación que presenta una relación dificultad/resultados mínima.

La mecánica es la misma que en el caso anterior: como tipo de documento se indica `prosper` y el argumento obligatorio del entorno `slides` es el título de la transparencia. Este entorno se repite tantas veces como transparencias queramos incorporar.

```
\begin{slide}{Título}
  \begin{itemize}
    \item Punto uno
    \item Punto dos
    \item Punto tres
  \end{itemize}
\end{slide}
```

Prosper dispone de una serie de diseños de página predefinidos, aplicables simplemente indicándolos como argumento opcional de la orden `\documentclass`. Las posibilidades son:

alienglow	autumn	azure	contemporain
darkblue	frames	lignesbleues	nuancegris
troispoints	gyom	rico	

Para profundizar en la creación de presentaciones con Prosper, nos remitimos a la bibliografía [15].

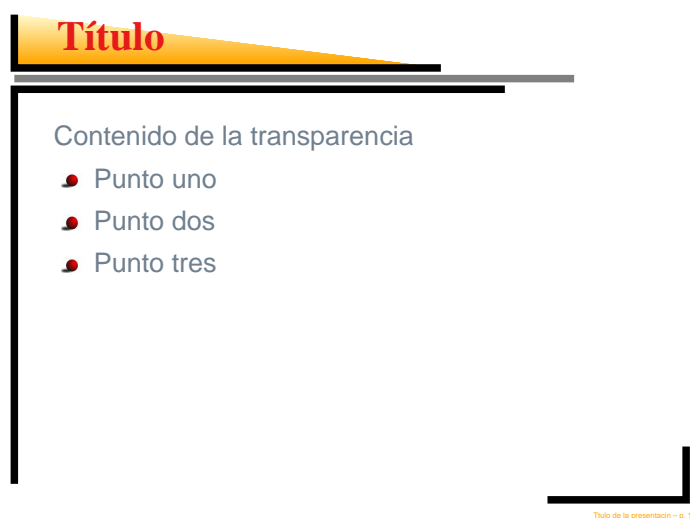


Figura B.1: Ejemplo de transparencia hecha con Prosper.

B.3. Una herramienta potente: beamer

Pese al gran avance que supuso la aparición de herramientas como Prosper, la evolución de la edición de presentaciones en \LaTeX no se detuvo ahí. Recientemente ha aparecido en escena una nueva alternativa, como parte de la tercera generación de este tipo de herramientas, que se está haciendo popular a pasos agigantados y se ha hecho un hueco por derecho propio.

El punto flaco de las herramientas coetáneas de Prosper es, sin duda, la capacidad de personalizar visualmente e incorporar dinamismo a las presentaciones. Elementos como animaciones, sonidos y vídeos son muy difíciles o prácticamente imposibles de añadir. Frente a esto, **Beamer** presenta un amplio nuevo conjunto de órdenes y comandos que se añaden a los comunes de \LaTeX .

La filosofía es esencialmente parecida a la que hemos visto hasta ahora: un nuevo tipo de documento, **beamer**, aparece en escena. El entorno `slide` de sus predecesores es sustituido por la orden `\frame{}`, de suerte que la apariencia del código fuente de una transparencia resulta similar a:

```
\frame{  
  \frametitle{Título}  
  \framesubtitle{Subtítulo}  
  
  \begin{itemize}  
    \item Punto uno  
    \item Punto dos  
    \item Punto tres  
  \end{itemize}  
}
```

El hecho de definir *frames* y no *slides* permite trabajar con efectos de muchos tipos. Además, existe la posibilidad de afinar hasta el más mínimo detalle la apariencia de las presentaciones haciendo uso de un gran número de comandos específicos de Beamer. Para más información, nos remitimos a la bibliografía [13].

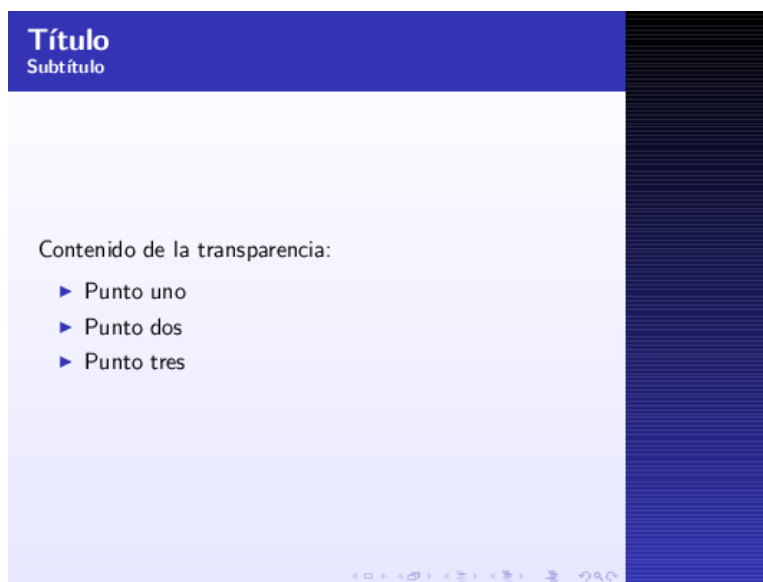


Figura B.2: Ejemplo de transparencia hecha con Beamer.

Apéndice C

Glosarios en L^AT_EX

Índice general

C.1. Paquete gloss	121
------------------------------	-----

EN el capítulo 7 veíamos cómo añadir a nuestros documentos referencias, secciones de bibliografía e incluso índices de materias. Una hibridación de estos dos últimos casos son los *glosarios*, cuya confección aprenderemos en este apéndice.

C.1. Paquete gloss

El paquete `gloss`, creado por el español José Luis Díaz, define una serie de comandos destinados a permitir la sencilla y cómoda inclusión de glosarios en documentos L^AT_EX, y se basa en la idea de usar BibT_EX para llevar a cabo esta tarea. Efectivamente, los pasos que hay que seguir para su utilización recuerdan no sólo a BibT_EX, sino también a `makeindex`.

Con la aproximación que `gloss` proporciona, el usuario edita una *base*, en este caso de términos y definiciones, en forma de fichero con una apariencia muy similar a una base bibliográfica (de hecho, también tendrá extensión `.bib`), donde todos los registros serán de la forma:

```

@GLOSSDEF{argumento,
word      = {argumento},
definition = {Valor que se proporciona a una función o comando
              a fin de concretar o modificar el resultado que
              produce.}
}

```

Igual que en el caso de las bibliografías creadas con BibTeX, los términos presentes en la base de términos que efectivamente se mostrarán en el glosario serán los que aparezcan “citados” en el documento, mediante el comando:

```
\gloss{Etiqueta}
```

Y también como en el caso de la bibliografía, si queremos que sean visibles todos los términos, independientemente de que se citen o no, añadiremos al documento:

```
\gloss[nocite]{*}
```

En el código fuente, además de añadir la orden `\usepackage{gloss}` (con la opción `[refpages]` si queremos que se añadan a cada entrada del glosario los números de página desde los que se referencia a cada una), debemos añadir en el preámbulo la orden:

```
\makegloss
```

y escribir, en el lugar en el que queramos que se imprima el glosario,

```
\printgloss{glsbase,glosario}
```

donde `glsbase` es un fichero que forma parte del paquete `gloss` y `glosario.bib` será el nombre del archivo que contiene la base de términos. Igual que sucede con las bases bibliográficas, las bases de términos son así fácilmente reutilizables, y

no es necesario mantenerlas ordenadas, pues es L^AT_EX quien realiza esta tarea al procesarlas.

Por último, para obtener el documento final invocaremos a la propia herramienta BibT_EX, con el fichero `documento.gls.aux` que se habrá generado tras una primera pasada sobre un `documento.tex` que contenga las órdenes que acabamos de mencionar para la inclusión de glosarios. Tras ello, una segunda pasada del compilador L^AT_EX nos ofrecerá la versión con el glosario, ordenado alfabéticamente, incorporado.

Apéndice D

L^AT_EX y el hipertexto

Índice general

D.1. latex2html	125
---------------------------	-----

EXPORTAR nuestros documentos L^AT_EX a HTML es una buena forma de dar a conocer nuestros contenidos al mundo a través de Internet. En este apéndice nos ocuparemos de esta cuestión.

Son muchos muchos los programas que se pueden usar a la hora de exportar un documento L^AT_EX a HTML. Aquí comentaremos uno de ellos: `latex2html`.

D.1. latex2html

El uso de `latex2html` es sencillo. Para usar esta herramienta, simplemente debemos incluir el paquete `html` (`\usepackage{html}`) en el preámbulo de nuestro documento. No es necesario compilar el documento L^AT_EX para obtener la versión HTML, puesto que `latex2html` realiza la conversión desde el código fuente. El proceso se reduce a teclear:

```
latex2html -dir dirDestino -split +1 -white documento
```

donde

- `-dir dirDestino` identifica *dirDestino* como el directorio dentro del que queremos que se genere toda la estructura HTML
 - `-split nivel` indica el nivel al que se deja de dividir las secciones en páginas HTML distintas (es decir, 0 haría que se generase un sólo documento HTML con todo el contenido, 1 generaría un documento por capítulo y así sucesivamente)
 - `-white` asegura que los fondos de las figuras sean blancos, para que posibles transparencias se muestren adecuadamente
-

Bibliografía

- [1] *Adobe Website.*
<http://www.adobe.com/products/acrobat/>.
- [2] *The BibTeX Format.*
<http://www.ecst.csuchico.edu/~jacobsd/bib/formats/bibtex.html>.
- [3] *Comprehensive T_EX Archive Network.*
<http://www.ctan.org>.
- [4] *El FAQ de CervanT_EX.*
<http://corbu.aq.upm.es/~agmartin/latex/FAQ-CervanTeX/FAQ-CervanTeX.html>.
- [5] *El sitio de L^AT_EX en español.*
<http://www.cervantex.org>.
- [6] *Getting Started with T_EX, L^AT_EX and friends.*
<http://www.tug.org/begin.html>.
- [7] *Ghostscript, Ghostview and GSview.*
<http://www.cs.wisc.edu/~ghost/>.
- [8] *Google.*
<http://www.google.es>.
- [9] *Instalar LaTeX en Windows.*
<http://www.udlap.mx/~ma108907/latex/winlatex.html>.
- [10] *An introduction to L^AT_EX.*
<http://www.latex-project.org/intro.html>.

- [11] *iTeXMac on the WEB*.
<http://itexmac.sourceforge.net/>.
 - [12] *Kile, an integrated L^AT_EX environment*.
<http://kile.sourceforge.net/>.
 - [13] *L^AT_EX Beamer*.
<http://latex-beamer.sourceforge.net/>.
 - [14] *MikT_EX Project Page*.
<http://www.miktex.org/>.
 - [15] *Prosper*.
<http://prosper.sourceforge.net/>.
 - [16] *Real Academia Española de la Lengua*.
<http://www.rae.es>.
 - [17] *Wikipedia, la enciclopedia libre*.
<http://es.wikipedia.org/>.
 - [18] Tomás Bautista et al.
Una descripción de L^AT_EX 2_ε.
<http://www.lsi.upc.es/~eipec/pdf/ldesc2e.pdf>.
 - [19] Javier Sanguino Botella.
Iniciación a L^AT_EX 2_ε Un sistema para preparar documentos.
Addison-Wesley, 1997.
 - [20] Jane Hahn.
L^AT_EX for everyone. A Reference Guide and Tutorial for typesetting documents using a computer.
Prentice Hall, 1993.
 - [21] Leslie Lamport.
A Document Preparation System L^AT_EX. User's Guide and Reference Manual.
Addison-Wesley, segunda edición, 1994.
 - [22] GPUL L^AT_EX.
El sitio de L^AT_EX del Grupo de Usuarios y Programadores de Linux.
<http://latex.gpul.org>.
-

-
- [23] Bernice Sacks Lipkin.
LaTeX for Linux. A Vade Mecum.
Springer-Verlang, 1999.
- [24] Tobias Oetiker et al.
The Not So Short Introduction to LaTeX 2_ε.
<http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf>.
- [25] Scott Pakin.
The Comprehensive LaTeX Symbol List, September 2003.
<http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf>.
- [26] Bernardo Cascales Salinas et al.
LaTeX una imprenta en sus manos.
Aula Documental de Investigación, 2000.
- [27] Bernardo Cascales Salinas et al.
El libro de LaTeX.
Prentice Hall, 2003.
- [28] ToolsCenter.org.
TeXnicCenter.
http://www.toolscenter.org/front_content.php?idcat=26.
- [29] Laura M. Castro Souto y Juan José Iglesias González.
Usando LaTeX 1.97.
Grupo de Programadores y Usuarios de Linux (GPUL).
<http://latex.gpul.org/html/main.html>.
-

Glosario

A

argumento Valor que se proporciona a una función o comando a fin de concretar o modificar el resultado que produce.

C

compilar Procesar código fuente para producir algún resultado en otro formato. El programa que realiza esta traducción recibe el nombre de *compilador*.

composición Conformar palabras, líneas y páginas, juntando las letras o caracteres y colocándolos de cierto modo y con cierto orden.

código fuente Texto escrito generalmente por una persona que se utiliza como base para generar otro código que posteriormente será interpretado o ejecutado por una computadora. El código fuente es texto simple, capaz de ser leído por cualquier editor de textos y lo que es más importante, entendible por cualquier programador.

D

DVI *DeVice Independent*. Formato de archivo informático independiente del dispositivo, empleado por T_EX como salida. A menudo, debe ser

reinterpretado por un programa secundario (postprocesador) para obtener el fichero definitivo. Lo más común es usar `dvips` para obtener un archivo Postscript.

Su nombre proviene de que el lenguaje en el que está escrito es idéntico para todos los dispositivos de lectura. El postprocesador convierte sus instrucciones al lenguaje adecuado para el dispositivo o formato de salida que se use en un determinado momento.

E

edición En informática, dar contenido a un archivo.

extensión En informática, una extensión de archivo o extensión de fichero, es una cadena de caracteres anexa al nombre de un archivo, usualmente antecedita por un punto. Su función principal es diferenciar el contenido del archivo de modo que el sistema operativo disponga el procedimiento necesario para ejecutarlo o interpretarlo.

Algunos sistemas operativos, especialmente los herederos de DOS como Windows, utilizan las extensiones de archivo para reconocer su formato, incluyendo el de archivos ejecutables. Otros sistemas operativos, como los basados en Unix, utilizan las extensiones de archivo por simple convención, no necesariamente utilizándolas para determinar su tipo.

H

HTML HTML, acrónimo en inglés de *HyperText Markup Language* (Lenguaje de Marcación de Hipertexto), es un lenguaje de marcas diseñado para estructurar textos y presentarlos en forma de hipertexto. Formato estándar para la creación de páginas web, indica al navegador cómo presentar la información gracias a una serie de indicaciones en forma de etiquetas.

P

PDF PDF (del inglés *Portable Document Format*, Formato de Documento Portable) es una forma de almacenamiento de documentos desarrollada por la empresa Adobe. Se trata de otro lenguaje de descripción de páginas, derivado de PostScript, pero más simple y liviano.

Postscript PostScript es un Lenguaje de Descripción de Página (en inglés PDL, *Page Description Language*), utilizado en muchas impresoras y como formato de transporte de archivos gráficos en talleres de impresión profesional. Está basado en el trabajo realizado por John Gaffney en Evans & Sutherland en 1976. Posteriormente, continuaron el desarrollo 'JaM' ('John and Martin', Martin Newell) en Xerox PARC, y finalmente fue implementado en su forma actual por John Warnock y otros, después de que él y Chuck Geschke fundaran Adobe Systems Incorporated (también conocido como Adobe) en 1982.

PostScript se diferenciò por utilizar un lenguaje de programación completo, en vez de una serie de secuencias de escape de bajo nivel, para describir una imagen para que sea impresa en una impresora láser o algún otro dispositivo de salida. También implementó notablemente la composición de imágenes, que consiste de un conjunto de líneas horizontales, píxeles al vuelo, descripciones por curvas de Bezier y tipografía (fuentes) de alta calidad a baja resolución (e.g. 300 puntos por pulgada). Anteriormente se creía que tipografías de mapa de bits mejoradas manualmente eran requeridas para esta tarea.

Ghostscript es una implementación abierta de un intérprete compatible con PostScript.

T

texto plano También denominados simplemente *archivos de texto*, los archivos de texto plano son aquellos que están compuestos únicamente por

texto sin formato, sólo caracteres. Carecen de información destinada a generar formatos y tipos de letra (por ejemplo, tipo de letra: Arial, Times, Courier; formato: negritas, subrayado, cursivas; tamaño, etc.).

W

WYSIWYG WYSIWYG es el acrónimo de *What You See Is What You Get* (en inglés, “lo que ves es lo que obtienes”). Se aplica a los procesadores de texto y otros editores de texto con formato que permiten escribir un documento viendo directamente el resultado final, frecuentemente el resultado impreso. Se les llama así en contraposición a otros procesadores de texto, hoy en día poco frecuentes, en los que se escribía sobre una vista codificada del formato del texto.

Ejemplos de editores tipo WYSIWYG son Microsoft Office o Writer (parte de la suite Open Office).

Ejemplo de formateador de textos que no es WYSIWYG: \LaTeX .

Índice alfabético

- €, *véase* euro
- 10pt, 28
- 11pt, 28
- 12pt, 28
- a4paper, 29
- a5paper, 29
- alineal texto, 51
 - a la derecha, 51
 - a la izquierda, 51
 - centrado, 52
- amsmath, 63
- amssymb, 63
- angle, 79
- any size, 99
- apéndices, 36
- appendix, 36
- array, 68
- article, 28
- artículo (clase de documento), 28
- author, 34
- autor o autores, 34
- b5paper, 29
- babel, 20
- bibitem, 90
- bibliografía, 90
- bibliography, 92
- bibliographystyle, 92
- BibTeX, 91
- bloque, 40
- book, 28
- borrador, 30
- cabeceras de página, 36
 - personalizadas, 98
- cajas, 104
 - de color, 103
- capítulos, 35
- caption, 77
- caracteres reservados, 21
- carta (tipo de documento), 80
- cartas
 - adjuntos, 80
 - apertura, 80

- copias, 80
 - despedida, 80
 - firma, 80
 - posdatas, 80
 - cc, 80
 - cdots, 69
 - center, 52
 - centering, 77
 - centerline, 52
 - centrar, 52
 - chapter, 35
 - citas textuales, 53
 - cite, 90
 - cleardoublepage, 100
 - clearpage, 100
 - cline, 74
 - closing, 80
 - color, 103
 - colorbox, 103
 - colores, 103
 - columnas, 30, 53
 - comandos, 17
 - comillas, 22
 - españolas, *véase también* francesas
 - francesas, 22
 - inglesas, 22
 - latinas, *véase* francesas
 - compilar, 13
 - convertir
 - .dvi en .pdf, 16
 - .dvi en .ps, 15
 - .pdf en .ps, 16
 - .ps en .pdf, 16
 - .tex en .html, 125
 - date, 34
 - ddots, 69
 - definecolor, 104
 - delimitadores, 69
 - ajustados, 69
 - description, 50
 - displaymath, 62
 - documentclass, 19
 - documento L^AT_EX
 - apéndices, 36
 - bibliografía, 90
 - BibT_EX, 91
 - citas bibliográficas, 90
 - citas textuales, 53
 - compilación, 13
 - cuerpo, 17
 - división en fragmentos, 32
 - división lógica, *véase* estructuración
 - encabezados, 36
 - personalizados, 98
 - errores, 107
 - estructuración, 34
 - etiquetas, 88
 - fecha, 34
 - fuentes, 41
 - gráficos, 77
-

-
- imágenes, 77
 - índice, 36
 - índice de materias, 94
 - interlineado, 99
 - márgenes, 99
 - notas a pie de página, 52
 - notas al margen, 52
 - pies de página, 36
 - portada, 33, 98
 - preámbulo, 17
 - referencias, 87, 88
 - tabla de contenidos, *véase* índice
 - tablas, 74
 - tipos, 28
 - opciones, 28
 - título, 34
 - transformación, 15
 - dotfill, 101
 - dots, 22, 69
 - doublespacing, 100
 - draft, 30
 - ecuaciones, 62
 - emph, 47
 - empty, 36
 - encl, 80
 - enfatizar texto, 47
 - entorno, 40
 - matemático, 62
 - entornos, 17
 - flotantes, 73
 - enumerate, 49
 - equation, 62
 - escalar
 - imágenes o gráficos, 78
 - espacios, 100
 - estilos de página, 36
 - euro, 103
 - eurosym, 103
 - executivepaper, 29
 - familia de letra
 - roman, 41
 - san serif, 41
 - typewriter, 41
 - fancybox, 104
 - fcolorbox, 103
 - fecha
 - de hoy, 34
 - de un documento, 34
 - figuras, 77
 - rotar, 79
 - figure, 79
 - final, 30
 - fleqn, 63
 - flotantes
 - figuras, 79
 - gráficos, 79
 - objetos, 73
 - tablas, 75
 - flushleft, 51
 - flushright, 51
-

footnote, 52

fórmulas matemáticas, 64

 binomios, 65

 cuantificadores, 67

 delimitadores, 69

 derivadas, 66

 determinantes, 68

 flechas, 68

 fracciones, 65

 integrales, 66

 límites, 66

 llaves, 68

 matrices, 68

 puntos suspensivos, 69

 raíces, 65

 símbolos, 70

 subíndices, 64

 sumatorios, 66

 superíndices, 64

frame, 119

fuentes, 41

girar, *véase también* rotar

gloss (orden), 122

gloss (paquete), 121

gráficos, 77

graphicx, 77

grosor de letra

grueso, 43

 medio, 43

 normal, *véase* medio

guiones, 22

 de segmentación silábica, 102

headings, 36

height, 78

hfill, 101

hline, 74

hrulefill, 101

hspace, 100

html (paquete), 125

imágenes, 77

include, 32

includegraphics, 78

index, 94

índice

 de contenidos, 36

 de figuras, 79

 de materias, 94

 de tablas, 77

 general, 36

 terminológico, 94

informe (clase de documento), 28

input, 32

inputenc, 20

interlineado, 99

item, 48

itemize, 48

label, 88

latex2html, 125

latexsym, 63

-
- left, 69
 - left., 69
 - leftline, 51
 - legalpaper, 29
 - leqno, 63
 - letter, 28, 80
 - letterpaper, 29
 - leyenda
 - de figuras, 79
 - en tablas, 77
 - libro (clase de documento), 28
 - list, 103
 - listas, 48
 - descriptivas, 50
 - no numeradas, 48
 - numeradas, 49
 - personalizadas, 103
 - listoffigures, 79
 - listoftables, 77
 - makegloss, 122
 - makeidx, 94
 - makeindex, 94, 95
 - maketitle, 34
 - márgenes, 99
 - marginpar, 52
 - marginsize, 99
 - markboth, 98
 - markright, 98
 - math, 62
 - multicol (paquete), 53
 - multicols (entorno), 53
 - multicolumn, 75
 - multirow, 75
 - myheadings, 98
 - newpage, 100
 - notas
 - a pie de página, 52
 - al margen, 52
 - notitlepage, 30
 - numeración
 - de ecuaciones, 62
 - evitar, 102
 - onecolumn, 30
 - onehalfspacing, 100
 - oneside, 29
 - openany, 29
 - opening, 80
 - openright, 29
 - órdenes, 17
 - pageref, 89
 - pagestyle, 37
 - paragraph, 35
 - párrafos, 35
 - part, 35
 - perfil de letra
 - inclinado*, 42
 - itálico*, 42
 - recto, 42
 - VERSALITA, 42
-

- plain, 36
 - preámbulo, 17
 - presentaciones L^AT_EX, 117
 - beamer, 119
 - entorno slide, 117
 - prosper, 118
 - printgloss, 122
 - printindex, 95
 - proc, 28
 - ps, 80
 - puntos suspensivos, 22, 69
 - quotation, 53
 - quote, 53
 - ref, 88
 - referencias
 - cruzadas, 87
 - internas, 87
 - report, 28
 - right, 69
 - right., 69
 - rightline, 52
 - rotar
 - imágenes o gráficos, 79
 - saltos de página, 100
 - scale, 78
 - secciones, 35
 - section, 35
 - see, 95
 - seealso, 95
 - segmentación de palabras, 102
 - setspace, 99
 - signature, 80
 - símbolos especiales, 22
 - singlespacing, 100
 - slide, 118
 - slides, 28
 - subparagraph, 35
 - subpárrafos, 35
 - subrayar, 47
 - subsecciones, 35
 - subsection, 35
 - subsubsecciones, 35
 - subsubsection, 35
 - tablas, 74
 - table, 75
 - tableofcontents, 36
 - tabular, 74
 - tamaño de letra, 44
 - base, 28
 - texto en columnas, 53
 - thebibliography, 90
 - tipo de letra, 41
 - title, 34
 - titlepage (entorno), 98
 - titlepage (opción), 30
 - titleref, 89
 - título, 34
 - transparencias, *véase* presentaciones
 - twocolumn, 30
-

`twoside`, 29

`underline`, 47

`usepackage`, 20

variables, 17

`vdots`, 69

`verbatim`, 47

`vfill`, 101

`vspace`, 100

`width`, 78

WYSIWYG, 4
