UNIVERSIDADE DA CORUÑA

Facultad de Informática
Aula de Formación Informática

Manual del Curso

Introducción a la edición de documentos con LaTeX

Laura M. Castro Souto

A Coruña, 8 – 17 de Noviembre de 2004.

Presentación del curso

Objetivos

LaTeX 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 o 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
  - Glosario
- 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.lfcia.org/~laura/cursos/latex.html`

Referencias

1. Bernardo Cascales Salinas et al.

   El libro de LaTeX.

   Prentice Hall, 2004.

2. Javier Sanguino Botella.

   Iniciación a LaTeX 2ε. 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

Índice de figuras

Índice de cuadros

Parte I

Manual

Capítulo 1
Introducción

Índice general

En este primer capítulo, de carácter introductorio, intentaremos responder a las primeras preguntas de aquellos 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 LaTeX un procesador de textos más?

No, nada más lejos de la realidad. Para empezar, LaTeX 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, que nos permiten editar 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 realizamos en el texto se reflejan instantáneamente en la pantalla de nuestro ordenador, a medida que los editamos.

En LaTeX la forma de trabajar es totalmente diferente. El usuario utiliza *cualquier otro editor de textos* para crear los *ficheros de entrada*, en los que además del texto que conformará el contenido del documento, se incluyen algunas indicaciones sobre las características del propio documento. Posteriormente, LaTeX 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.

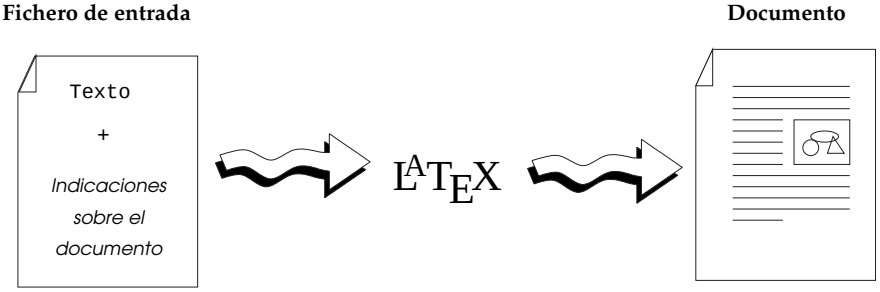


Figura 1.1: Funcionamiento de LaTeX

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

¿Cuál es la ventaja de usar LaTeX, 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 de tablas contenidos, índices, notas a pie o al margen, y un amplio etcétera.

La gran ventaja de LaTeX 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 aquel 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* LaTeX quien se ocupará del asunto. ¿A que suena bien?

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

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.

Y por cierto, 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 pretenden ser personas familiarizadas con los ordenadores que deseen obtener una pequeña visión de LaTeX.

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 de libros matemáticos o técnicos.

LaTeX proporciona, tal y como veremos a lo largo de este manual, un tratamiento 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: aprenderemos los conceptos básicos que nos permitan desenvolvernos inicialmente, para profundizar más adelante. El número de comandos e instrucciones que deberemos aprender, 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 desee.

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 tex-

tos, que siguiese lo más fielmente posible las tradicionales normas tipográficas. Así surgió TeX, nombre que hace referencia tanto al lenguaje que creó, como a su intérprete o compilador. El problema era que TeX 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 TeX una colección de comandos que simplificaban el mecanografiado, permitiendo centrarse en la estructura del texto en vez de en los comandos para dar formato. Ese pequeño conjunto de comandos se denominó LaTeX. Años más tarde, sucesivas revisiones dieron origen a LaTeX 2_ε, el último estándar, que incluía, entre otras cosas, comandos para la inclusión de gráficos y la utilización de color.

TeX 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*, LaTeX se ha convertido prácticamente en una *lingua franca* del mundo científico.

1.4. LaTeX también está ahí fuera

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 interese.

Son muchos miles los usuarios de LaTeX 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 LaTeX que creen y se intercambien "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:

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. *The Not So Short Introduction to LaTeX 2ε.*

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

5. *An introduction to LaTeX.*

   `http://www.latex-project.org/intro.html`

6. *Getting Started with TeX, LaTeX, and Friends.*

   `http://www.tug.org/begin.html`

Capítulo 2
Conceptos básicos

Índice general

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 LaTeX?

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

1. **Edición del texto *fuente***, 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 tiene la posibilidad de guardar lo que tecleemos en *texto plano*, es decir, sin formato alguno. Estos *ficheros de entrada* contendrán, además del contenido del documento propiamente dicho, una serie de indicaciones, dadas siguiendo una sintaxis determinada, que proporcionan a LaTeX 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, del mismo modo que la utilización de acentos y otros caracteres, aunque no es posible que incluya 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* LaTeX, 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`, una versión *ligera* del documento que nos permite comprobar los resultados del procesado.

3. **Visualización o impresión**. Aunque la versión *dvi* del documento que obtenemos tras la compilación del documento tiene ya la apariencia final del mismo, no empaqueta las imágenes incluidas, entre otras cosas. Es por ello

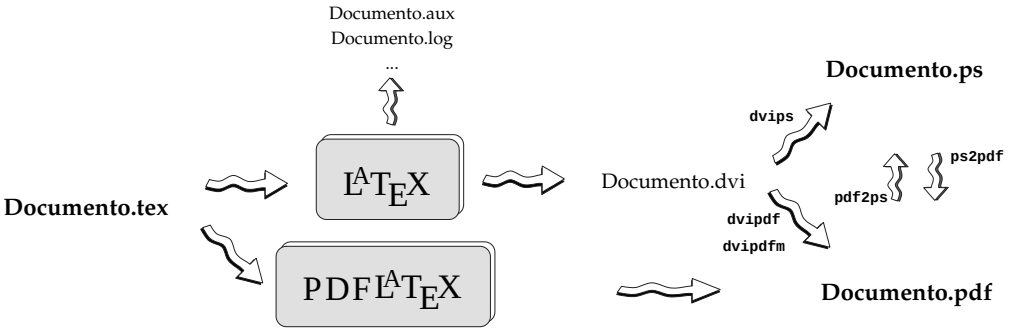


Figura 2.1: Funcionamiento detallado de LaTeX

que, usualmente, suele transformarse a otro formato, normalmente *Postscript* o *PDF*. Con cualquiera de estos formatos, ya tenemos disponible la versión definitiva de nuestro documento, perfectamente adecuada para ser no ya sólo visualizada, sino impresa o intercambiada a través de Internet.

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 envuelven las etapas de *compilación* y *transformación*, y 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 y, en ocasiones, el modo de abordarlo y solucionarlo (para más detalles, véase el apéndice A, dedicado a este tema).

Como sabemos, LaTeX 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 procesado del documento:

1. En el *primer procesado* se recopila información, por ejemplo, de dónde se encuentran las figuras y el modo en que están etiquetadas, de la página en que comienza cada capítulo y su título, de 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`), e 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 procesado*, 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, la mayoría de las veces necesitaremos ejecutar el compilador LaTeX al menos un par de veces.

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

Tal y como se aprecia 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 y obtener una versión definitiva del documento que estemos creando: escoger el formato **Postscript** o el formato **PDF**.

2.1.2.1. Especial para impresión: escogiendo el 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 Documento.dvi -o Documento.ps
```

donde la opción `-o` nos permite cambiar el nombre del fichero Postscript resultante, en este caso será `Documento.ps`.

El formato Postscript presenta la ventaja fundamental de que muchas impresoras hoy en día lo entienden, 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` [6].

2.1.2.2. Popular en Internet: escogiendo el 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, ésta convierte el formato DVI directamente a PDF, mientras que la primera emplea ghostscript y `dvips`:

```
dvipdf Documento.dvi [Informe.pdf]
```

```
dvipdfm Documento.dvi [Informe.pdf]
```

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

Cuando comentábamos la herramienta `dvipdf` decíamos que hacía 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: `teTeX` (salvo las herramientas `ps2pdf`/`pdf2ps`, que suelen formar parte de un paquete denominado `psutils`).

Para otras plataformas, existen sistemas equivalentes, como `TeXshop` o `iTeXMac` para MacOSX o `MikTeX` para Windows.

2.2. Estructura de un documento

Ahora que ya sabemos cómo compilar un documento LaTeX y 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* (\) que va seguida del nombre del comando (que es sensible a mayúsculas y minúsculas) y, en caso necesario, de una lista de atributos opcionales (entre corchetes, separados por comas) u obligatorios (entre llaves). Pueden verse varios ejemplos en la página siguiente.

`\comando`	ejemplo de comando
`\Comando`	otro comando distinto
`\cmd{atributo}`	comando con atributo obligatorio
`\cmd[opcion]`	comando con atributo opcional
`\cmd[opcion,opcion2=valor]{atributo}`	comando con varios atributos opcionales y uno obligatorio

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

Por último, en LaTeX 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}
  Y este otro puede sumar ambos conjuntos de propiedades
  o que se impongan las de éste último.
 \end{entorno2}
 ...
 Aquí volvemos a las propiedades anteriores, <ordenadito!
\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 viendo. El texto del documento se teclea dentro del entorno `document`, que consituye 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 LaTeX, notaremos enseguida que no importa que nuestro texto

ocupe dos líneas en el fichero fuente: LaTeX 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 LaTeX un asesor incompetente?

LaTeX 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. LaTeX 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 indica 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 TeX/LaTeX, 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 LaTeX 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 LaTeX no se vaya a confundir nunca a la hora de segmentar una palabra y no tengamos manera de corregirle. Los acentos 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 LaTeX. Uno de ellos es precisamente la barra inclinada a la izquierda (\), que indica comienzo de comando, orden, variable e incluso que le sigue 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 corchetes[1] 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 ignorado 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

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

suspensivos. La forma de obtener las distintas variaciones de comillas (simples, dobles, latinas[2] e inglesas), así como los distintos tipos de guiones se indican en la tabla siguiente.

Comillas	*Simples*	Inglesas	` '	'hola'
	Dobles	Latinas	<< >>	«hola»
		Inglesas	`` ''	"hola"
Guiones	*Simples*		-	- hola
	Dobles		--	– hola
	Triples		---	— hola

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

2.4. Herramientas para trabajar con LaTeX

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?

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 y la visualización en pantalla usando visores de Postscript –como `gv`– o PDF –como `acroread`–). Algunos de estos programas son:

Kile para Unix/Linux.

TeXnicCenter para Windows.

[2] También llamadas francesas o españolas.

Para MacOSX los sistemas ya mencionados TeXshop o iTeXMac ya integran un editor de este tipo, con múltiples menús donde las opciones LaTeX más comunes están a disposición del usuario novel, liberándole de verse en la tesitura 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 tareas totalmente configurables.

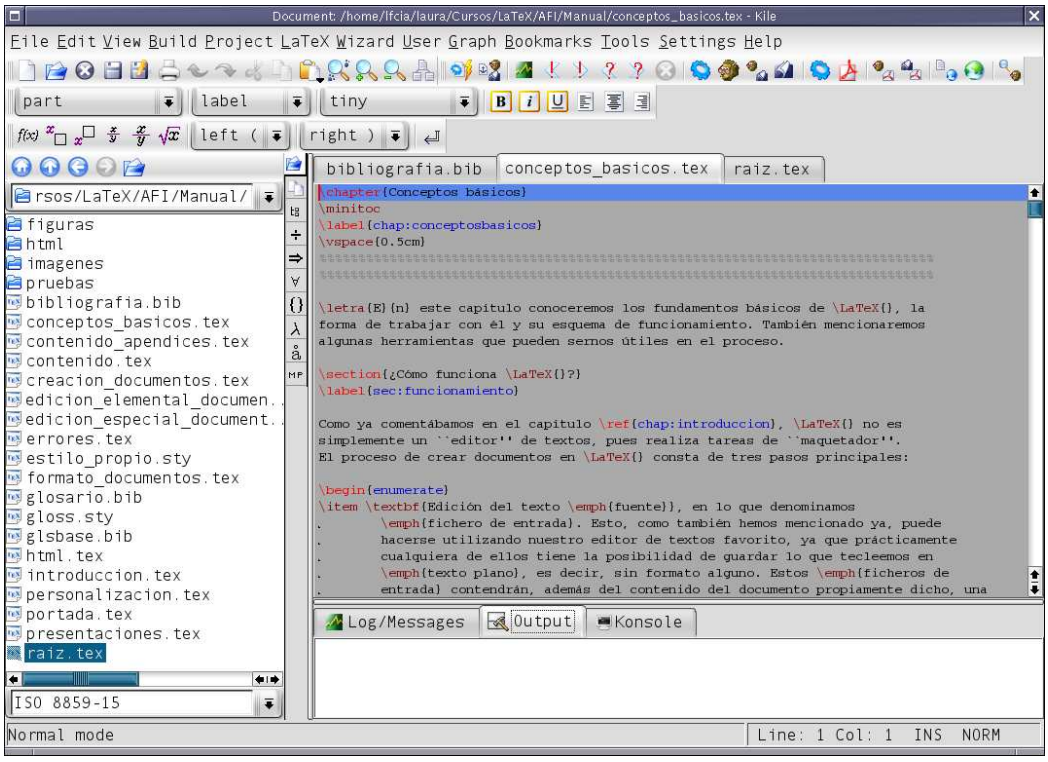


Figura 2.2: Captura de pantalla del editor Kile

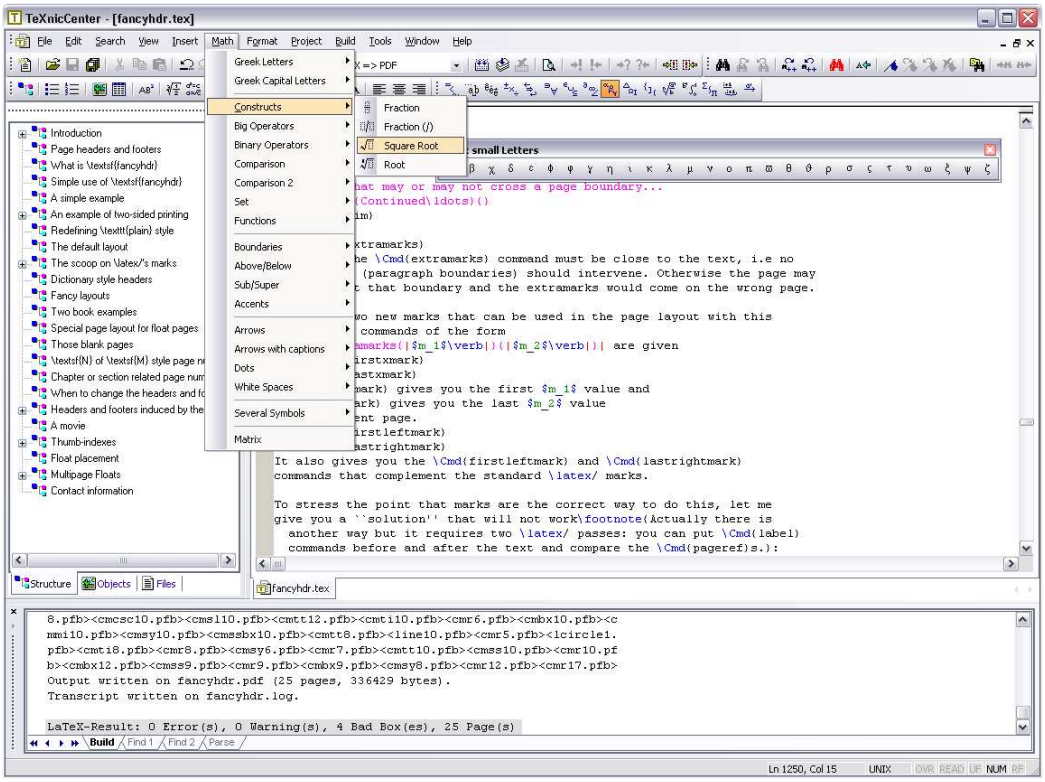


Figura 2.3: Captura de pantalla del editor TeXnicCenter

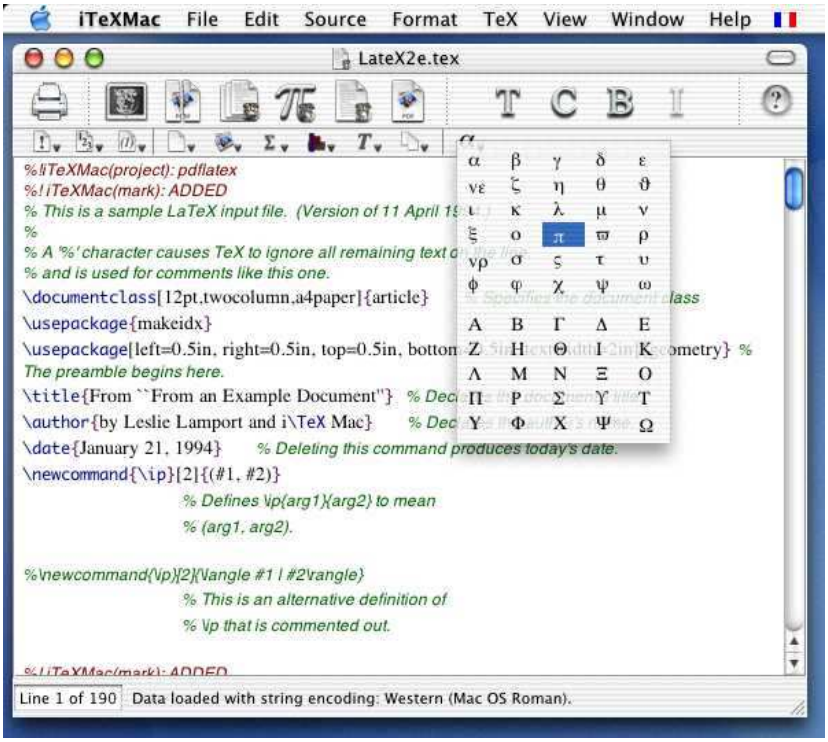


Figura 2.4: Captura de pantalla del editor iTEXMac

Capítulo 3

Creación de documentos

Índice general

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`. Junto con

estas dos clases base, disponemos de los tipos `proc` y `report`, que derivan de las dos 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 LaTeX pueden ser susceptibles de adaptar su comportamiento según nuestras indicaciones, gracias a las *opciones* que podemos suministrarles. La orden `\documentclass` presenta la siguiente serie de opciones:

Tamaño de letra – Todas las clases de documentos LaTeX 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, el tamaño de una nota 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 LaTeX sigue fielmente, no es posible especificar cualquier valor para el tamaño base de la letra del documento. LaTeX 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 LaTeX 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[1], 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 defectos 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 no muchos de nosotros escribamos un

[1] Lo haremos en el capítulo siguiente.

	`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.
[c] Espacio en blanco significa opción no disponible.

Cuadro 3.1: Diferencias entre las distintas clases de documentos LaTeX

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}
```

y

```
\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

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, LaTeX nos avisará si lo omitimos:

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

Fecha – Además del título y el autor, en la portada que LaTeX 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}`:

`\date{}`	fecha vacía
`\date{Noviembre de 2004}`	fecha "personalizada"
`\date{\today}`	mismo efecto que LaTeX

Estos tres comandos se colocarán en el *preámbulo* del documento (recordemos, entre el `\documentclass` y el `\begin{document}`). Con esto LaTeX tiene de dónde 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`
Parte (`\part`)	✓	✓
Capítulo (`\chapter`)		✓
Sección (`\section`)	✓	✓
Subsección (`\subsection`)	✓	✓
Subsubsección (`\subsubsection`)	✓	✓
Párrafo (`\paragraph`)	✓	✓
Subpárrafo (`\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 `\paragraph` y `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 aparte[1]. Para conseguir que entre

[1] Es 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 (aquélla 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`
Estilo `plain`	✓	✓		✓
Estilo `headings`			✓	

Cuadro 4.2: Estilos por defecto de los documentos LaTeX

Capítulo 5

Edición elemental de documentos

Índice general

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

5.1. Entornos y bloques

Antes de entrar al 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 como muñecas rusas. Es decir, se encierran unos dentro de otros y deben cerrarse en el mismo orden en que se abren, "casando" las partes de arriba (`begin`s) con las partes de abajo (`end`s).

```
\begin{...}1
 {2 ... }2
 {3
   \begin{...}4
     {5 ... }5
   \end{...}4
   {6 ... }6
 }3
\end{...}1
```

✓ ***correcto***

```
\begin{...}1
 {2 ... {3
        }3
   \begin{...}4
      {5 ...
   \end{...}4
        }5 {6 ...
 }2 }6
\end{...}1
```

× ***incorrecto***

[1] Recordemos que las llaves son uno de los caracteres reservados de LaTeX (ver página 21).

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 actividad".

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 LaTeX 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:

	Comando + argumento	*Bloque + orden*
Familia roman	`\textrm{`*Texto*`}`	`{ \rmfamily` *Texto* `}`
Familia sanserif	`\textsf{`*Texto*`}`	`{ \sffamily` *Texto* `}`
`Familia typewriter`	`\texttt{`*Texto*`}`	`{ \ttfamily` *Texto* `}`

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

Una forma alternativa al uso de un bloque y la orden `\XXfamily` dentro de él, es la utilización del entorno del mismo nombre:

	Entorno
Familia roman	`\begin{rmfamily}` *Texto* `\end{rmfamily}`
Familia sanserif	`\begin{sffamily}` *Texto* `\end{sffamily}`
`Familia typewriter`	`\begin{ttfamily}` *Texto* `\end{ttfamily}`

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 manera[2]:

	Comando + argumento	*Bloque + orden*
Perfil recto	`\textup{`*Texto*`}`	`{ \upshape` *Texto* `}`
Perfil itálico	`\textit{`*Texto*`}`	`{ \itshape` *Texto* `}`
Perfil inclinado	`\textsl{`*Texto*`}`	`{ \slshape` *Texto* `}`
Perfil versalita	`\textsc{`*Texto*`}`	`{ \scshape` *Texto* `}`

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

	Entorno
Perfil recto	`\begin{upshape}` *Texto* `\end{upshape}`
Perfil itálico	`\begin{itshape}` *Texto* `\end{itshape}`
Perfil inclinado	`\begin{slshape}` *Texto* `\end{slshape}`
Perfil versalita	`\begin{scshape}` *Texto* `\end{scshape}`

5.2.3. Grosores

Por último, con respecto al estilo de letra, disponemos en LaTeX de dos grosores básicos: **medio** (normal, grosor por defecto) y **grueso** (negrita). Los comandos a aplicar son[2]:

Y los entornos se denominan:

[2]Los ejemplos se aplican a la familia `roman`.

	Comando + argumento	*Bloque + orden*
Grosor normal	`\textmd{`*Texto*`}`	`{ \mdseries` *Texto* `}`
Grosor negrita	`\textbf{`*Texto*`}`	`{ \bfseries` *Texto* `}`

	Entorno
Grosor normal	`\begin{mdseries}` *Texto* `\end{mdseries}`
Grosor negrita	`\begin{bfseries}` *Texto* `\end{bfseries}`

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

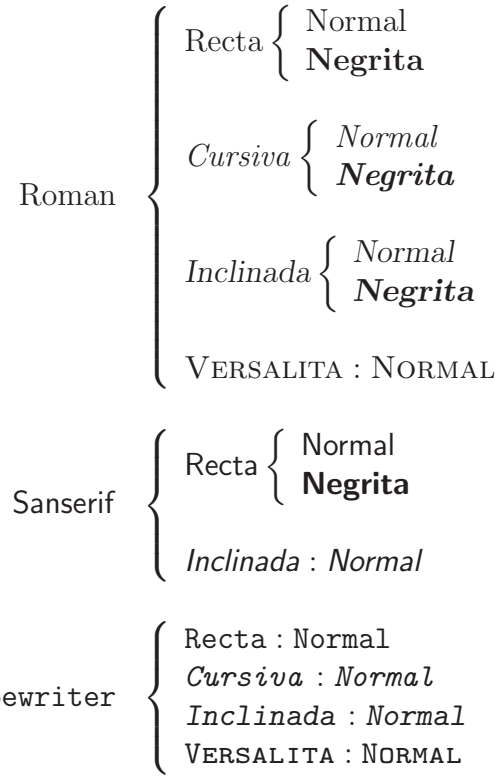
Además del estilo, es posible modificar el tamaño del texto. Para ello, LaTeX nos presenta 10 comandos, en este caso sólo es posible usarlos en bloques o como entornos:

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.

5.2.5. Otros efectos

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

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



Cuadro 5.1: Combinaciones posibles de estilos de letra en LaTeX

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

	Entorno	
`\begin{tiny}`	*Texto*	`\end{tiny}`
`\begin{scriptsize}`	*Texto*	`\end{scriptsize}`
`\begin{footnotesize}`	*Texto*	`\end{footnotesize}`
`\begin{small}`	*Texto*	`\end{small}`
`\begin{normalsize}`	*Texto*	`\end{normalsize}`
`\begin{large}`	*Texto*	`\end{large}`
`\begin{Large}`	*Texto*	`\end{Large}`
`\begin{LARGE}`	*Texto*	`\end{LARGE}`
`\begin{huge}`	*Texto*	`\end{huge}`
`\begin{Huge}`	*Texto*	`\end{Huge}`

	Opción `10pt`	*Opción* `11pt`	*Opción* `12pt`
`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

Enfatizado `\emph{`*Texto*`}`

enfatiza el *Texto* para diferenciarlo del resto del texto a su alrededor, de manera que si cambian las características de ese texto (haciéndose itálico, por ejemplo), LaTeX optaría por mostrar el *Texto* recto, para diferenciarlo.

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

Subrayado `\underline{`*Texto*`}`

Sin embargo, no es una forma de resaltar texto aconsejada, cuando se dispone de distintos estilos de letra. 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 LaTeX 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 LaTeX, 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 a 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. LaTeX se encarga de la gestión de la apariencia de los distintos niveles de profundidad:

- leche
- pan y cereales
  - trigo
  - harina

- cebada
- centeno
- maíz

- legumbres
  - lentejas
  - garbanzos

```
\begin{itemize}
\item leche
\item pan y cereales
  \begin{itemize}
  \item trigo
    \begin{itemize}
    \item harina
    \end{itemize}
  \item cebada
  \item centeno
  \item maíz
  \end{itemize}
\item legumbres
  \begin{itemize}
  \item lentejas
  \item garbanzos
  \end{itemize}
\end{itemize}
```

5.3.2. Listas numeradas

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

1. buscar un local
   a) llamar por teléfono
   b) visitar el sitio
   c) confirmarlo
2. enviar las invitaciones
3. contratar la decoración

```
\begin{enumerate}
\item buscar un local
  \begin{enumerate}
  \item llamar por teléfono
  \item visitar el sitio
  \item confirmarlo
  \end{enumerate}
\item enviar las invitaciones
\item contratar la decoración
\end{enumerate}
```

5.3.3. Listas descriptivas

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

prosa estructura o forma del lenguaje que...

verso palabra o conjunto de palabras sujetas...

```
\begin{description}
\item [prosa] estructura o forma
              del lenguaje que\dots
\item [verso] palabra o conjunto de
              palabras sujetas\dots
\end{description}
```

Se usan normalmente para descripciones de términos o similares.

Por supuesto, las 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 LaTeX 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, a 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 ambos entornos modifican los márgenes de la página con el fin de que el párrafo o párrafos incluidos en estos entornos 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 un fragmento que se muestre en varias columnas. 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}
El texto se distribuye automáticamente en tantas columnas
como indiquemos como argumento obligatorio del propio entorno.
\end{multicols}
```

El texto se distribuye automáticamente en tantas columnas como indiquemos como argumento obligatorio del propio entorno.

Capítulo 6
Edición especial de documentos

Índice general

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 apéndices o cartas.

6.1. Edición matemática

La edición matemática es el terreno sobre el que LaTeX mejor demuestra su gran 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 LaTeX:

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 tener una referencia a la fórmula –véase capítulo 7, sección 7.1–).

Ejemplo de uso del entorno `equation`:

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

```
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 con 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.

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 muy fáciles de recordar porque se corresponden con abreviaturas de los nombres que reciben esos símbolos en inglés (y que, al tratarse de símbolos internacionalmente usados, son nombres 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:

$$E = mc^2$$

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

$$a_{n+1} = a_n + 1$$

```
\begin{displaymath}
a_{n+1} = a_n + 1
\end{displaymath}
```

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 conveniente indicarlo entre llaves, de forma que LaTeX sepa que todo el contenido del bloque entre llaves es lo que queremos que forme parte del super/subíndice. De lo contrario, podríamos obtener un resultado no deseado, como: $a_n + 1$ (`a_n+1`).

6.1.3.2. Raíces

Las raíces se escriben en el modo matemático LaTeX 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:

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

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

6.1.3.3. 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` en el 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`: $\dfrac{1}{2} = \dfrac{2}{4}$ (`\dfrac{1}{2}=\dfrac{2}{4}`). Para conseguir el efecto contrario, es decir, tamaño *inline* en entorno *display* existe el recíproco `\tfrac`:

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

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

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

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

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

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

```
\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

LaTeX dispone de sendos comandos para proporcionar los cuantificadores universal (*para todo*, $\forall$ `\forall`) y existencial (*existe*, $\exists$ `\exists` y la negación *no existe*, $\nexists$ `\nexists`).

Asimismo, comandos bastante sencillos proporcionan funciones como el seno (sin `\sin`), coseno (cos `\cos`), tangente (tan `\tan`), cotangente (cot `\cot`), logaritmo (log `\log`), logaritmo neperiano (ln `\ln`), máximo (máx `\max`), mínimo (mín `\min`), 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 `\text{`*Texto*`}`:

esto ya es ***otra*** cosa

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

Además, 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:

$$a \rightarrow b \Rightarrow c \Leftarrow d \leftarrow d$$

```
\begin{displaymath}
a\rightarrow b\Rightarrow
c \Leftarrow d \leftarrow d
\end{displaymath}
```

$$\underbrace{a+b+c+d}_{x} = \overbrace{e+f+g+h}^{y}$$

```
\begin{displaymath}
  \underbrace{a+b+c+d}_{x}
= \overbrace{e+f+g+h}^{y}
\end{displaymath}
```

6.1.3.8. Matrices y determinantes

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

$$\left(\begin{array}{cccc} a_1 & a_2 & \cdots & a_m \\ b_1 & b_2 & \cdots & b_m \\ \vdots & \cdots & \ddots & \vdots \\ n_1 & n_2 & \cdots & n_m \end{array} \right)$$

```
\begin{displaymath}
\left(
\begin{array}{cccc}
a_1    & a_2   & \cdots & a_m    \\
b_1    & b_2   & \cdots & b_m    \\
\vdots & \dots & \ddots & \vdots \\
n_1    & n_2   & \cdots & n_m    \\
\end{array}
\right)
\end{displaymath}
```

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 (*right*)

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

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 `&`.

Este ejemplo 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` ($\dots$)
- centrados verticalmente con respecto a la línea de escritura: `\cdots` ($\cdots$)
- verticales: `\vdots` ($\vdots$)
- diagonales: `\ddots` ($\ddots$)

En cuanto a los delimitadores, para conseguir paréntesis del tamaño ajustado, simplemente usamos las secuencias `\left`*delimitador* y `\right`*delimitador*, siendo *delimitador*es posibles:

paréntesis (como hemos visto)

barra vertical $\left|\frac{1}{2}\right|$ `\left| \dfrac{1}{2} \right|`

corchetes $\left[\frac{2}{3}\right]$ `\left[ \dfrac{2}{3} \right]`

llaves $\left\{\frac{3}{4}\right\}$ `\left\{ \dfrac{3}{4} \right\}`[1]

Además, es posible indicar sólo uno de los dos delimitadores (sólo el izquierdo o sólo el derecho). En ese caso, no obstante, no vale simplemente no poner el delimitador correspondiente, sino sustituirlo por un `\left.` o `\right.` según convenga. Así, el entorno `array` puede utilizarse perfectamente para crear sistemas de ecuaciones:

[1] Nótese que hay que escapar las llaves, que de por sí son un carácter reservado.

$$\left\{ \begin{array}{rcl} a + b & = & 4 \\ 2a + 3b & = & 36 \end{array} \right.$$

```
\begin{displaymath}
\left\{ \begin{array}{rcl}
         a + b   & = & 4 \\
         2a + 3b & = & 36 \\
        \end{array}
\right.
\end{displaymath}
```

6.1.3.9. 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 siguiente es una compilación de algunos otros símbolos útiles, como las letras griegas más utilizadas o símbolos como el del conjunto vacío.

α	\alpha	β	\beta	γ	\gamma	δ	\delta
ϵ	\epsilon	η	\eta	θ	\theta	κ	\kappa
λ	\lambda	μ	\mu	ν	\nu	π	\pi
ρ	\rho	σ	\sigma	τ	\tau	ϕ	\phi
χ	\chi	ψ	\psi	ω	\omega		
Γ	\Gamma	Δ	\Delta	Θ	\Theta	Λ	\Lambda
Π	\Pi	Σ	\Sigma	Φ	\Phi	Ψ	\Psi
Ω	\Omega						
∇	\nabla	$\surd$	\surd	$\top$	\top	$\bot$	\bot
$\emptyset$	\emptyset	$\cap$	\cap	$\cup$	\cup	$\oplus$	\oplus
$\ominus$	\ominus	$\otimes$	\otimes	$\times$	\times	$\div$	\div
$\vee$	\vee	$\wedge$	\wedge	$\approx$	\approx	$\approxeq$	\approxeq
$\equiv$	\equiv	$\geq$	\geq	$\leq$	\leq	$\gneq$	\gneq
$\lneq$	\lneq	$\gg$	\gg	$\ll$	\ll	$\neq$	\neq
$\ngtr$	\ngtr	$\nless$	\nless	$\ngeq$	\ngeq	$\nleq$	\nleq
$\in$	\in	$\notin$	\notin	$\subset$	\subset	$\supset$	\supset
$\subseteq$	\subseteq	$\supseteq$	\supseteq	$\nsubseteq$	\nsubseteq	$\nsupseteq$	\nsupseteq

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, [23] 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 los más completos manuales de la bibliografía y por supuesto al sitio web [20].

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 en que se maqueta dentro de una página. Son objetos 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 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 60 para las matrices (comando `array` del entorno matemático):

esto	sólo es	una simple
tabla	de	ejemplo

```
\begin{tabular}{rcl}
esto  & sólo es & una simple \\
tabla & de      & ejemplo    \\
\end{tabular}
```

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 (`r`, `c` o `l`).

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:

esto	sólo es	una simple
tabla	de	ejemplo

```
\begin{tabular}{r|c|l|}
\cline{2-3}
esto  & sólo es & una simple \\
\hline
tabla & de      & ejemplo    \\
\hline \hline
\end{tabular}
```

Otro comando útil es el `\multicolumn`, que nos permite *fundir* varias columnas en una o, lo que es lo mismo, hacer que una celda ocupe el lugar de varias. Recíprocamente, 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):

esta tabla	sólo es	un simple
		de ejemplo

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

6.2.2.1. Tablas flotantes

El entorno `tabular` no es de por sí un entorno flotante. El entorno flotante correspondiente a las tablas es el entorno `table`. Si incluimos las tablas tal y como hemos visto hasta ahora, 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
		de ejemplo

Cuadro 6.2: Tabla de prueba

```
\begin{table}[hbtp!]
\centering
\begin{tabular}{r|c|l|}
\cline{2-3}
\multirow{2}{2cm}{esta tabla} & sólo es & un simple \\
\cline{2-3}
& \multicolumn{2}{r}{de 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 las diferentes posibilidades 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

! sugiere a LaTeX que sea un poco más flexible en sus consideraciones para poder ocupar un lugar preferido con mayor probabilidad

Así, en la tabla anterior, la secuencia `[hbtp!]` 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 está 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.

Otro par de comandos hemos incluido en el ejemplo anterior. 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* La usada en el ejemplo anterior, indica la anchura de la imagen (que puede ser mayor o menor que la del fichero origen, LaTeX se encarga de hacer el escalado y mantener las proporciones si no indicamos la opción `height`.

height=*longitud* De modo análogo a la opción `width`, se usa para indicar la altura que queremos que tenga la imagen, que no tiene por qué coincidir con la real.

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

angle=*valor* Podemos indicar a LaTeX que rote la figura. Por defecto, si *valor*, que ha de ser un entero entre 0 y 360, 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:

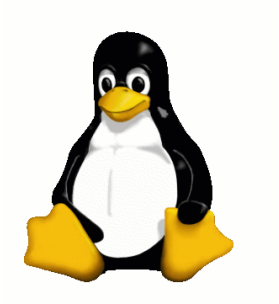


Figura 6.1: Imagen de ejemplo

```
\begin{figure}[hbtp!]
\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. Si incluimos el comando `\caption` dentro del entorno `figure` conseguiremos que en la lista generada por la orden `\listoffigures` aparezca 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 lugar.

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 recíproca 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 LaTeX (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 LaTeX

Capítulo 7
Referencias internas

Índice general

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

7.1. Referencias básicas

Las referencias son útiles para relacionar partes de un documento. Además, en LaTeX son una herramienta particularmente útil y cómoda, ya que simplemente tendremos 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 referencias afectadas.

Así pues, necesitamos dos elementos para utilizar referencias: una manera de "marcar", como decimos, los puntos a los que se va a referir, y la forma de hacer referencia a dichos puntos:

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 pequeña 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, o sección, o tabla, o 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 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 89) 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 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 y donde se necesiten.

La generación de referencias es una de las cosas que obliga a compilar el 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.

7.3. Índice de materias

LaTeX 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 préámbulo 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 página correspondiente al punto del documento donde se escribe el comando en el código fuente.

subentradas Producen una entrada de nivel inferior, concretando 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 bien 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 se ya incluye generalmente con las distribuciones TeX/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.

Capítulo 8

Personalización

Índice general

Hay quien opina que LaTeX es fácil de usar en tareas simples pero 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.

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` es igual que `headings`, pero proporciona 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:

		markboth		markright
		izquierda	*derecha*	*derecha*
oneside	`article` y `proc`			*section*
	`book` y `report`			*chapter*
twoside	`article` y `proc`	*section*	(*vacío*)	*subsection*
	`book` y `report`	*chapter*	(*vacío*)	*section*

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

Para esta tarea pueden ser útiles los siguientes comandos:

- `\the`*Estructura* (i.e. `\thechapter`, `\thesection`...) introduce el número de la *Estructura* activa en ese lugar
- `\`*Estructura*`name` (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 `anysize`, 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\-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 sin leyenda, o 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 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
♣ con símbolo
♣ personalizado

```
\begin{list}{$\clubsuit$}{}
\item lista
\item con símbolo
\item personalizado
\end{list}
```

Para un buen repertorio de símbolos, puede consultarse cualquier referencia de la bibliografía, pero en especial [23].

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

caja de color

caja de color con borde

```
{ \color{blue} texto de color }   \fcolorbox{red}{yellow}
\colorbox{green}{caja de color}          {caja de color con borde}
```

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

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

pueden definirse nuevos colores utilizando los esquema RGB ó CMYK, donde `codificacion` 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 LaTeX

Índice general

Tan importante como aprender cómo trabajar con LaTeX 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, su cerrado en distinto orden. En el primero de los casos

(que ocurrirá normalmente dentro de entornos matemáticos), LaTeX nos advierte de la situación:

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

l.73 \end{itemize}
```

En caso de que estemos manejando varios ficheros, deberemos fijarnos en unas líneas antes, hasta que encontremos el nombre de aquél 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 comandos (sobre todo símbolos) fuera del entorno matemático. Esto produce el siguiente comportamiento:

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

l.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 la cantidad de ellos:

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

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

l.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 &.
l.139 ...zación en medio del texto sin escaparlo &
                                                  produce
```

A.4. Ojo a lo que escribimos

Si durante la compilación LaTeX se encuentra con algún comando u orden que esté mal escrita 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   l.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.
 ...

l.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
\ref{inexistente}
```

```
LaTeX Warning: Reference
  ‘inexistente’ on page 86
  undefined on input line 59.
```

esto pasa al incluir una **??**

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 el error se produce de manera temprana de forma que al terminar la compilación el mensaje podría quedar fuera de nuestra vista, líneas antes de finalizar LaTeX advierte:

```
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, el aviso de LaTeX será:

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

LaTeX Warning: Label etiqueta multiply defined.


...


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

Si recordamos por qué necesitamos compilar varias veces un documento LaTeX (la 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 LaTeX 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 medida en alguno de los argumentos opcionales relativos a longitudes, LaTeX protestará:

```
\includegraphics[width=5]{imagenes/ejemplo.eps}

! Illegal unit of measure (pt inserted).
<to be read again>
                   \relax
l.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
l.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 LaTeX `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 LaTeX 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 mensjaes `under`, que no ha conseguido rellenar todo el espacio sobrante como le hubiera gustado.

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 LaTeX

Índice general

DESPUÉS de habernos acostumbrado a las bondades de LaTeX, 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 TeX/LaTeX 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 el contenido de una transparencia y cuyo argumento obligatorio incluye definiciones de distinta índole y generalmente permanece vacío. De este modo, podían elaborarse documentos con múltiples entornos `slide` rellenados a gusto del autor.

B.2. Una herramienta sencilla: Prosper

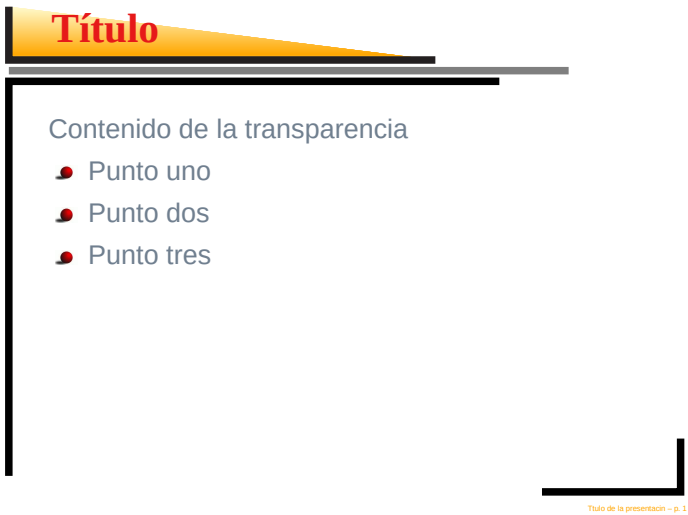
La flexibilidad del tipo de documento `slides` es realmente escasa y los resultados, pobres. Es por ello que rápidamente surgen diferentes paquetes y herramientas para tratar de poner solución a esta materia. Entre ellas, elegimos **Prosper** como recomendación de relación dificultad/resultados mínima.

La mecánica es la misma que en el caso anterior, el tipo de documento se indica `prosper` y el argumento obligatorio del entorno `slides` es el título de la transparencia.

Como ventajas, señalaremos que 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.



Apéndice C

LaTeX y el hipertexto

Índice general

EXPORTAR nuestros documentos LaTeX 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 LaTeX a HTML. Aquí comentaremos uno de ellos: `latex2html`.

C.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 LaTeX 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 midocumento
```

La opción `-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). Por último `-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] *Comprehensive TEX Archive Network.*
http://www.ctan.org.

[3] *El FAQ de CervanTEX.*
http://corbu.aq.upm.es/~agmartin/latex/FAQ-CervanTeX/FAQ-CervanTeX.html.

[4] *El sitio de LATEX en español.*
http://www.cervantex.org.

[5] *Getting Started with TEX, LATEX and friends.*
http://www.tug.org/begin.html.

[6] *Ghostscript, Ghostview and GSview.*
http://www.cs.wisc.edu/~ghost/.

[7] *Google.*
http://www.google.es.

[8] *Instalar LaTeX en Windows.*
http://www.udlap.mx/~ma108907/latex/winlatex.html.

[9] *An introduction to LATEX.*
http://www.latex-project.org/intro.html.

[10] *iTEXMac on the WEB.*
http://itexmac.sourceforge.net/.

[11] *Kile, an integrated LaTeX environment.*
`http://kile.sourceforge.net/`.

[12] *MikTeX Project Page.*
`http://www.miktex.org/`.

[13] *Prosper.*
`http://prosper.sourceforge.net/`.

[14] *Real Academia Española de la Lengua.*
`http://www.rae.es`.

[15] *Wikipedia, la enciclopedia libre.*
`http://es.wikipedia.org/`.

[16] Tomás Bautista et al.
Una descripción de LaTeX 2ε.
`http://www.lsi.upc.es/~eipec/pdf/ldesc2e.pdf`.

[17] Javier Sanguino Botella.
Iniciación a LaTeX 2ε Un sistema para preparar documentos.
Addison-Wesley, 1997.

[18] Jane Hahn.
LaTeX for everyone. A Reference Guide and Tutorial for typesetting documents using a computer.
Prentice Hall, 1993.

[19] Leslie Lamport.
A Document Preparation System LaTeX. User's Guide and Reference Manual.
Addison-Wesley, segunda edition, 1994.

[20] GPUL LaTeX.
El sitio de LaTeX del Grupo de Usuarios y Programadores de Linux.
`http://latex.gpul.org`.

[21] Bernice Sacks Lipkin.
LaTeX for Linux. A Vade Mecum.
Springer-Verlang, 1999.

[22] Tobias Oetiker et al.
The Not So Short Introduction to LaTeX 2ε.
http://www.ctan.org/tex-archive/info/lshort/english/lshort.pdf.

[23] Scott Pakin.
The Comprehensive LaTeX Symbol List, September 2003.
http://www.ctan.org/tex-archive/info/symbols/comprehensive/symbols-a4.pdf.

[24] Bernardo Cascales Salinas et al.
LaTeX una imprenta en sus manos.
Aula Documental de Investigación, 2000.

[25] Bernardo Cascales Salinas et al.
El libro de LaTeX.
Prentice Hall, 2003.

[26] ToolsCenter.org.
TeXnicCenter.
http://www.toolscenter.org/front_content.php?idcat=26.

[27] 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 programas en código fuente para producir algún resultado en otro formato. El programa que realiza esta traducción recibe el nombre de compilador.

composición Formar las palabras, líneas y páginas, juntando las letras o caracteres, juntándolos 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 TEX como salida. A menudo, debe ser

reinerpretado 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.

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 anexada al nombre de un archivo, usualmente antecedida 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.

P

PDF PDF (del inglés *Portable Document Format*, Formato de Documento Portable) es una forma de almacenamiento de documentos, desarrollado por la empresa Adobe. PDF es 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 y 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, luego de que él y Chuck Geschke fundaran Adobe Systems Incorporated (también conocido como Adobe) en 1982.
PostScript se diferenció por utilizar un lenguaje de programación completo, en vez de una serie de secuencias de escapes 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, pixeles 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 (como los editores de HTML) que permiten escribir un documento viendo directamente el resultado final, frecuentemente el resultado impreso. Se dice 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. En el caso de editores de HTML este concepto se aplica a los que permiten escribir la página sobre una vista preliminar similar a la de un procesador de textos, ocupándose en este caso el programa de generar el código fuente en HTML.

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

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

Índice alfabético