

Material de Apoyo

Curso de \LaTeX 2 $_{\epsilon}$

Uso y Adopción de \LaTeX 2 $_{\epsilon}$

Un enfoque práctico

Manuel E. Merino

Julio de 2023

Versión 1.5

Material de Apoyo

Licencia:

© ⓘ \$ ⓘ Creative Commons - BY-NC-SA
Attribution-NonCommercial-ShareAlike

Esta permitida la reproducción total o parcial de esta obra, siempre y cuando se brinden los créditos correspondientes. No está permitido el uso comercial de la obra ni de sus derivados. Si desea modificar, transformar y/o ampliar el contenido de esta obra, deberá distribuir sus contribuciones bajo la misma licencia.

Para ver una copia digital de la licencia, visita:

https://creativecommons.org/licenses/by-nc-sa/4.0/deed.es_ES

Colofón

Este documento ha sido elaborado utilizando: **KOMA-Script** y **L^AT_EX**

Aviso

El presente documento es material de apoyo para el curso de L^AT_EX 2_ε de 4.0 Academy, y ha sido elaborado por Manuel E. Merino, en Julio de 2023.

Índice general

1	Introducción a \LaTeX	5
1.1	¿Qué es \LaTeX ?	5
1.2	¿Por qué aprender y adoptar \LaTeX ?	6
1.3	¿Por qué no \LaTeX ?	7
1.4	Conclusiones	8
2	Instalación de Software	9
2.1	\LaTeX en línea: Overleaf	9
2.2	Distribuciones de \TeX	9
2.3	Editor \TeX Studio	12
2.4	Software adicional	13
3	Documento de \LaTeX	17
3.1	Preámbulo	17
3.2	Cuerpo del documento	18
3.3	Comandos y entornos	19
3.4	Paquetes	21
4	Escribiendo un archivo .tex	25
4.1	El archivo .tex	25
4.2	Compiladores de \LaTeX	25
4.3	¿Cómo interpreta \LaTeX al archivo .tex	26
5	Secciones de un documento	29
5.1	Página de título	29
5.2	Comandos de seccionado	30
5.3	Tabla de contenidos	32
5.4	Consideraciones adicionales con la clase book	32
5.5	Apéndices	33
6	Formato de texto	35
6.1	Familia tipográfica de texto	35
6.2	Estilos de texto	36
6.3	Comandos para dar formato al texto	36

Índice general

6.4	Cambiando las tipografías por defecto	37
6.5	Tamaño de texto	39
6.6	Formato de párrafo	40
6.7	Color de texto	45
6.8	Conclusiones	47

Capítulo 1

Introducción a \LaTeX

1.1. ¿Qué es \LaTeX ?

Hablar de \LaTeX es hablar de tres conceptos distintos:

- El sistema \LaTeX .
- El lenguaje de marcado \LaTeX (formato).
- El motor \LaTeX (*\LaTeX engine*)

1.1.1. El sistema \LaTeX

Es un sistema de composición tipográfica usado para la elaboración de documentos técnicos y científicos de alta calidad, especialmente en las áreas de ciencias, tecnología, ingeniería y matemáticas.

Es el estándar de facto para la comunicación y publicación de documentos científicos en diversas áreas.

El sistema \LaTeX gestiona la disposición (estructura lógica) y el formato (estructura visual) del documento utilizando conceptos familiares como lo son los capítulos, secciones, elementos (tablas, figuras, ecuaciones), listas, entre otros.

\LaTeX plantea una filosofía de creación de documentos en la que el contenido escrito y la apariencia visual están separados. De esta manera, los autores pueden centrarse más en el contenido que en su apariencia.

1.1.2. El lenguaje de marcado \LaTeX

\LaTeX no es un **procesador de texto** como Word, Pages o Docs. En estos programas, el usuario dispone de una página u hoja interactiva donde puede colocar directamente el

contenido de su documento, y aplicar el formato correspondiente, y este se ve reflejado de manera inmediata en la pantalla. Por eso se suele decir que son programas o editores del tipo WYSIWYG (*what you see is what you get*), “lo que ves es lo que obtienes”.

En el sistema \LaTeX en cambio, se tiene un archivo de **texto plano** o texto sin formato, el cual puede ser abierto con cualquier **editor de texto**, como Block de Notas, Notepad++, entre otros. Y es en este archivo en el cual se tiene el contenido del documento, organizado gracias a unas **etiquetas** o **marcas** que especifican la estructura lógica y/o formato visual del documento. Estas marcas y funciones componen el conjunto de macros conocido como \LaTeX , o más específicamente $\text{\LaTeX} 2_{\epsilon}$.

Este archivo de **texto plano** es procesado posteriormente por un **compilador \LaTeX** el cual producirá un documento final, generalmente en formato pdf.

Al ser el archivo de entrada un archivo de texto sin formato, puede ser editado por cualquier editor de texto, por lo que para crear nuestros documentos .tex podemos utilizar Visual Code Studio, \TeX Studio, \TeX Maker, Vim, entre otros.

1.1.3. El motor \LaTeX

Los archivos .tex son procesados por un compilador o motor \LaTeX , para producir un documento final, en formato pdf.

Existen diversos motores \LaTeX , pero tres son los más utilizados:

- $\text{pdf}\text{\LaTeX}$
- $\text{Xe}\text{\LaTeX}$
- $\text{Lua}\text{\LaTeX}$

En este curso, usaremos principalmente el motor $\text{Lua}\text{\LaTeX}$.

1.2. ¿Por qué aprender y adoptar \LaTeX ?

Como se mencionó en la sección anterior, \LaTeX es un sistema de composición tipográfica, no un procesador de texto como Word, Docs o Pages. Mientras que en los procesadores de texto clásicos siguen una filosofía **WYSIWYG**, \LaTeX adopta un enfoque distinto, recibiendo texto e instrucciones para crear un documento de salida empleando algoritmos sofisticados de alineación de texto, espaciado y colocación de elementos junto a la aplicación de formatos predefinidos personalizables (tipografía, tamaño de letra, etc.).

Todo esto se traduce en una curva de aprendizaje más pronunciada: Adoptar \LaTeX toma tiempo, pero sus beneficios pueden compensar con creces las dificultades iniciales en su aprendizaje.

Entre los principales beneficios de \LaTeX se pueden nombrar los siguientes:

- Separación de estructura visual y estructura lógica.
- Composición matemática de alta calidad (fórmulas y ecuaciones).
- Gestión de bibliografía y referencias.
- Estructura visual del documento homogénea.
- Fácil integración con sistemas de **control de versiones**, como Git.
- Repetibilidad y estabilidad a largo plazo.

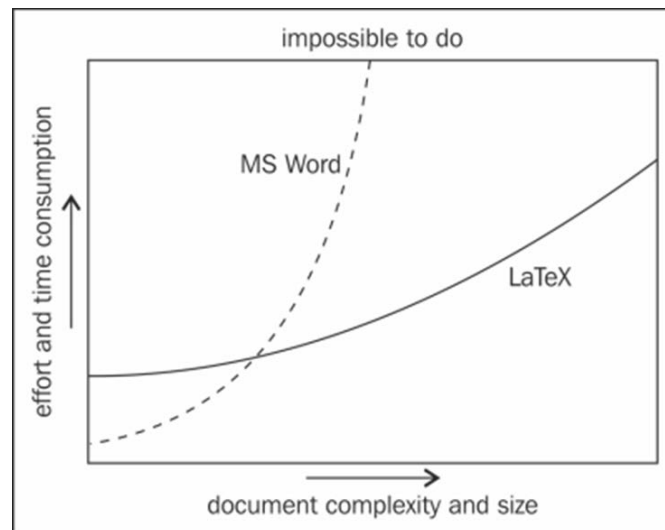


Figura 1. Curva de aprendizaje de \LaTeX

1.3. ¿Por qué no \LaTeX ?

Adoptar \LaTeX también consiste en reconocer cuando no debería ser utilizado.

- No tienes tiempo para aprender a utilizar \LaTeX .
- Participas en un trabajo colaborativo y eres el único que sabe utilizar \LaTeX .

- El documento que tienes ya ha sido escrito y diseñado utilizando otro procesador de texto.
- El documento requiere un formato o diseño muy específico y que sólo usarás una única vez.

1.4. Conclusiones

\LaTeX plantea un enfoque diferente para la creación de documentos, y lograr separar la estructura lógica de la estructura visual durante la composición de un documento resulta clave para sacarle el mayor provecho. Esto puede resultar difícil al principio, pues requiere ir en contra de ciertos hábitos heredados del uso de programas **WYSIWYG**.

Al inicio, es normal demorar más en crear un documento utilizando \LaTeX que un procesador de texto convencional, pero a largo plazo el método sistematizado de creación de documentos propuesto por \LaTeX permite ahorrar bastante tiempo y estandarizar el aspecto visual de los documentos elaborados.

Capítulo 2

Instalación de Software

Para este curso, necesitaremos una instalación de \LaTeX y un editor de texto, de preferencia con soporte de \LaTeX . Se tienen dos alternativas: una instalación local (en la computadora) o una instalación en línea (en la nube).

2.1. \LaTeX en línea: Overleaf

Overleaf es un editor en línea y gratuito. Permite acceder a una distribución \LaTeX sin necesidad de instalación: sólo requiere un ordenador y una cuenta de usuario.

Para crear una cuenta gratuita, se puede acceder al siguiente enlace: <http://www.overleaf.com/register>.

2.2. Distribuciones de \TeX

Para procesar un archivo de \LaTeX se requiere tener instalado un motor o compilador \LaTeX y probablemente varios paquetes adicionales. Estos requerimientos se simplifican mediante la instalación de una distribución \LaTeX , como lo es MiK \TeX , T \EX Live, entre otros.

A continuación, se mostrarán una guía paso a paso para la instalación básica de la distribución MiK \TeX , además de un editor especializado (T \EX Studio) y otros programas adicionales que serán necesarios para el curso.

2.2.1. Instalación de MiK \TeX

El primer paso para la instalación de MiK \TeX para Windows, es ingresar a la página de descargas de MiK \TeX (<https://miktex.org/download>) y descargar la versión *Installer*.

Una vez descargado el instalador ([basic-miktex-22.10-x64.exe](#)), darle doble clic para ejecutarlo. Esto abrirá una ventana donde se listan las condiciones de uso de

2 Instalación de Software

MiKTeX (Figura 2). Seleccionar la casilla *I accept the MiKTeX copying conditions*, y darle clic a **Siguiente**.



Figura 2. Instalador MiKTeX - Condiciones de Uso

En la siguiente ventana del instalador, se recomienda seleccionar la opción *Install MiKTeX only for me* (Figura 3), y darle clic a **Siguiente**.

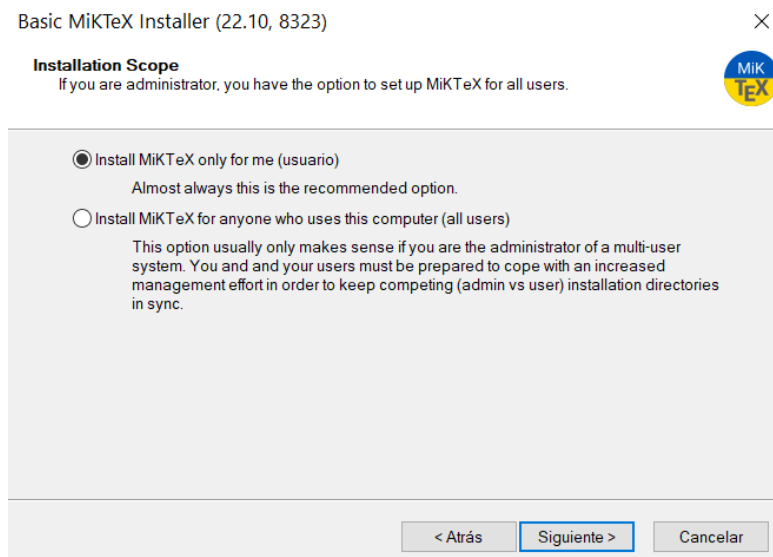


Figura 3. Instalador MiKTeX - Usuario

Posteriormente se solicitará la ruta de instalación (Figura 4). Luego de definirla, dar click a **Siguiente**.

A continuación, se sugiere seleccionar la opción de instalación de paquetes *on-the-fly*, tal y como se muestra en la figura 5, pues permitirá que MiKTeX detecte e instale de manera automática los paquetes faltantes al compilar un nuevo documento.

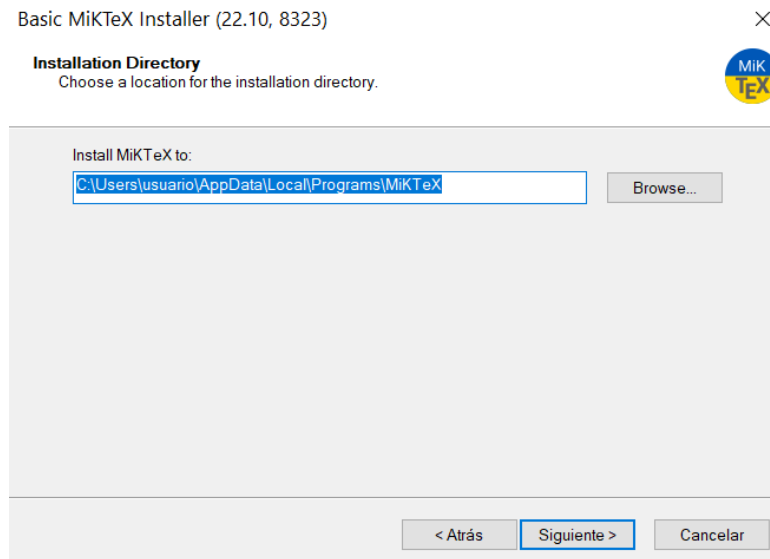


Figura 4. Instalador MiK $T_E X$ - Ruta de instalación

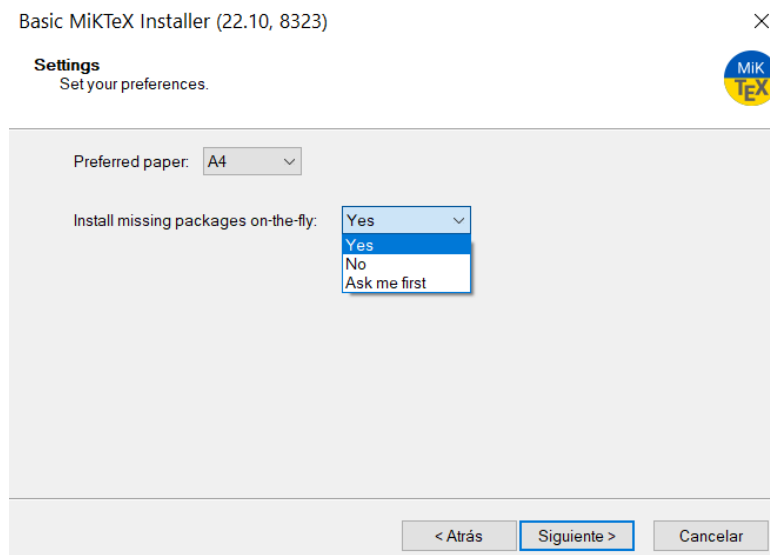


Figura 5. Instalador MiK $T_E X$ - Instalación *on-the-fly*

Luego de haber dado clic a **Siguiente** o **Next**, MiKTeX procederá a instalarse en su sistema operativo. Si no han ocurrido errores durante la instalación, aparecerá la ventana de actualización (Figura 6). Se recomienda marcar *Check for updates now* y dar clic otra vez a **Siguiente** o **Next**. Esto abrirá la consola de MiKTeX.

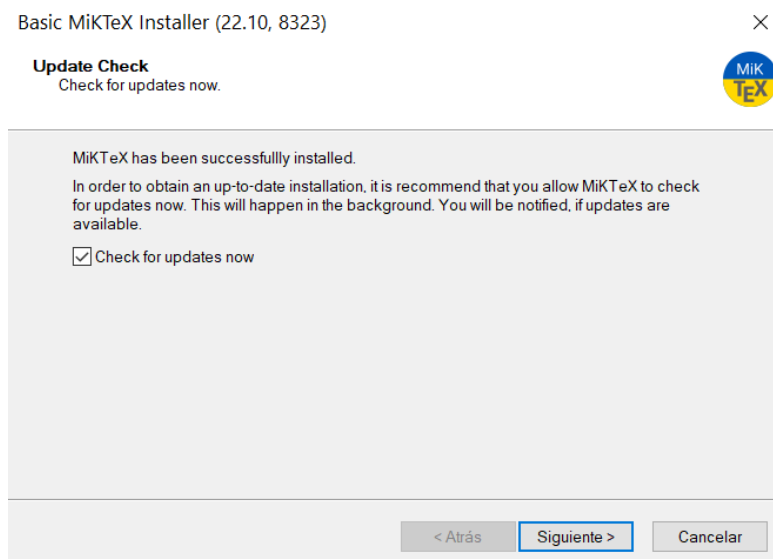


Figura 6. Instalador MiKTeX - Revisar actualizaciones

2.3. Editor T_EXStudio

Existen diversos entornos de desarrollo y editores de texto para usar L^AT_EX, y la elección es netamente personal. Sin embargo, si se desea tener un editor dedicado netamente a L^AT_EX, se sugiere utilizar T_EXMaker o T_EXStudio.

La instalación de T_EXStudio es bastante sencilla. Sólo se requiere descargar el instalador desde la página oficial (<https://www.texstudio.org/>), y ejecutar el archivo `texstudio-4.3.1-win-qt6.exe`.

En el instalador, simplemente se debe seleccionar la ruta de instalación (Figura 7), y dar clic en **Instalar**.

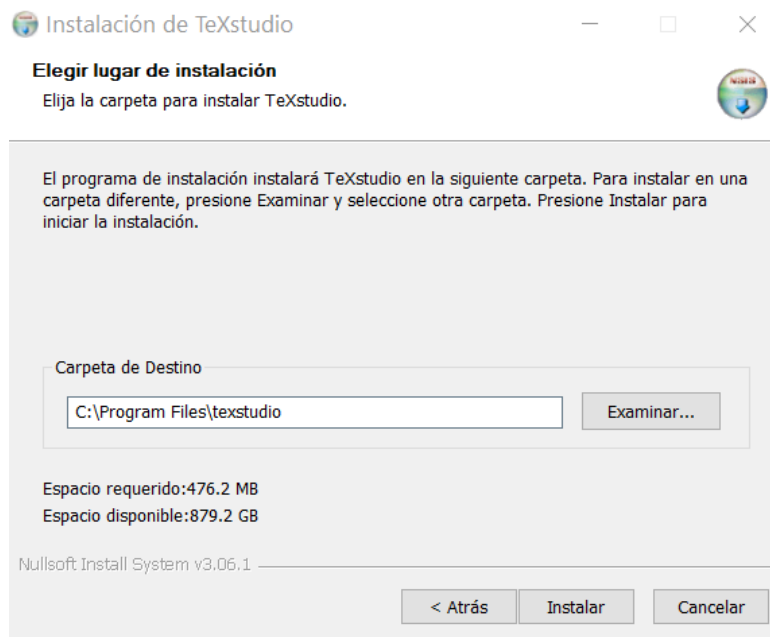


Figura 7. Instalador TeXstudio - Ruta de Instalación

2.4. Software adicional

En este curso se usarán algunos paquetes adicionales que requieren de Python y la librería Pygments para funcionar correctamente. A continuación veremos la instalación de ambos.

2.4.1. Instalación de Python

Uno de los paquetes que veremos en el curso, requiere de Python, por lo que también debe instalarse.

Se sugiere descargar la última versión de Python disponible en la [página oficial](#).

Una vez descargado el archivo `python-3.11.0-amd64.exe` (el nombre puede variar), se debe ejecutar, dándole doble clic al instalador. Esto abrirá una ventana (Figura 8), en la

En dicha ventana, se sugiere marcar la opción *Add python.exe to PATH* y dar clic a **Install Now**.

En caso preferir una instalación personalizada (*Customize installation*), no se debe desactivar la opción de instalar **pip** (Figura 9), ni olvidar de agregar Python a las variables (Figura 10). Para iniciar la instalación puede dar clic en **Install**.

2 Instalación de Software

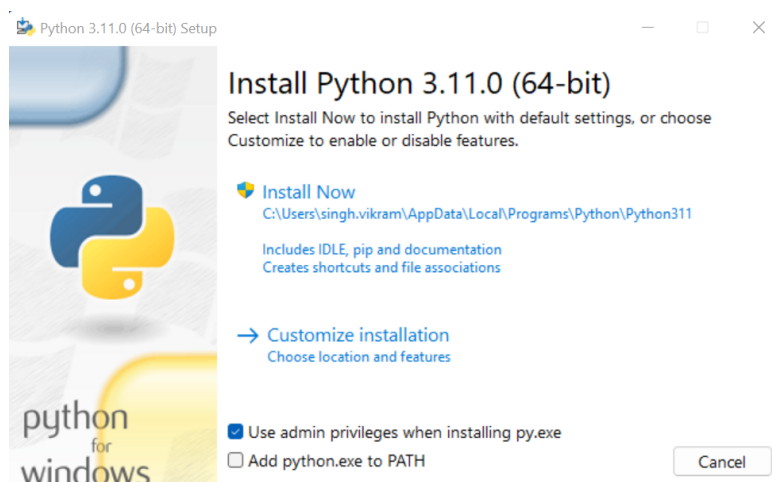


Figura 8. Instalador Python - Primera ventana

Si la instalación se ha completado de manera satisfactoria, debe aparecer la ventana de confirmación (Figura 11), donde se debe dar clic en **Close**.

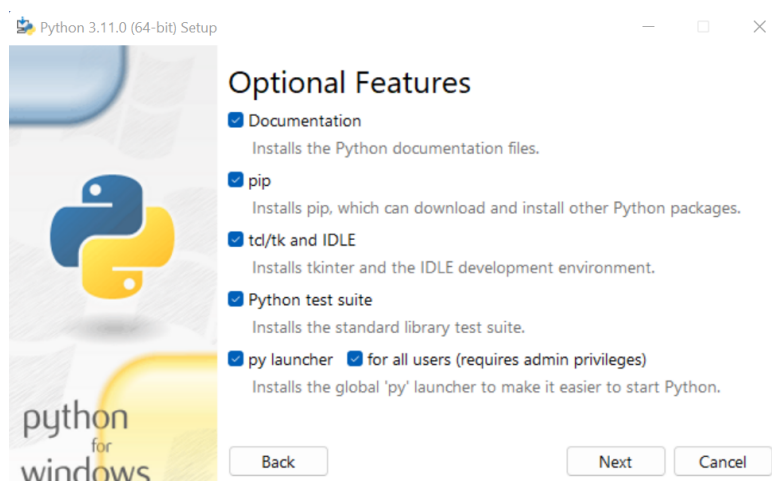


Figura 9. Instalador Python - No olvidar instalar pip

Para comprobar que Python está configurado correctamente y puede ser ejecutado desde la shell de Windows, se debe presionar las teclas **Windows** + **R**, escribir **cmd** en la ventana *Ejecutar* y presionar **Enter**.

En la ventana resultante, se debe escribir el comando **python --version** y presionar **Enter**.

La terminal debe regresar como resultado la versión de Python. En este caso, la versión instalada es Python 3.11.0.

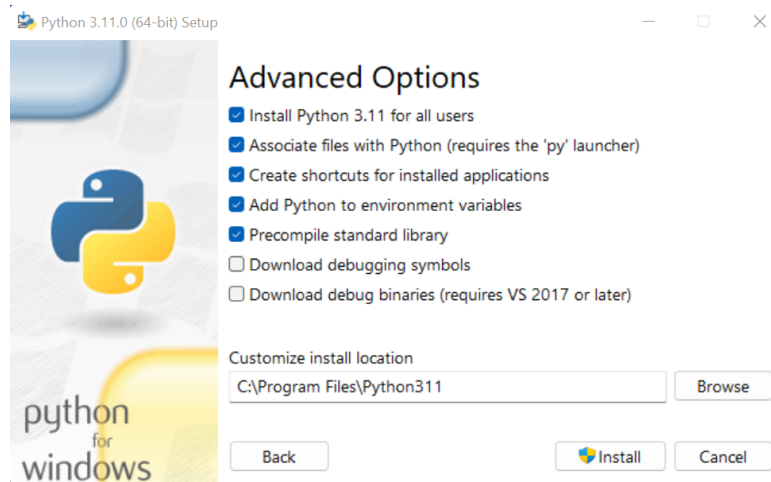


Figura 10. Instalador Python - Agregar Python a variables

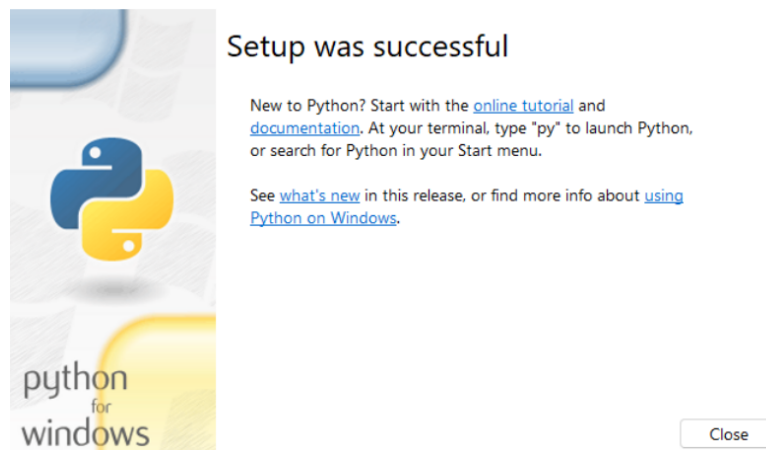


Figura 11. Instalador Python - Instalación completada

2.4.2. Instalación de Pygments

En la ventana anterior (cmd), ejecutar el comando `pip install pygments`. Es probable que `pip` solicite confirmación. En dicho caso, presionar `Y` (o escribir `yes`) y confirmar presionando `↵`.

Al finalizar la instalación de pygments, cerrar la ventana y volver a abrirla siguiendo el proceso anteriormente descrito (`Windows` + `R`, cmd, `↵`), y ejecutar el comando `pygmentize -V`.

El resultado obtenido debe ser Pygments version 2.13.0., seguido de una nota de *copyright*.

Capítulo 3

Documento de L^AT_EX

En el ejemplo 3.1, se muestra un documento básico de L^AT_EX.

Ejemplo 3.1: Documento básico de L^AT_EX

```
1 \documentclass{article}
2 % Más paquetes ...
3 \begin{document}
4   Texto de ejemplo.
5 \end{document}
```

Lo primero que se puede observar es que L^AT_EX utiliza comandos, los cuales empiezan con el símbolo \.

En segundo lugar, podemos identificar dos secciones: el **preámbulo** y el **cuerpo del documento**.

3.1. Preámbulo

El preámbulo es la primera parte de un documento en L^AT_EX y contiene la configuración global para el procesamiento del documento (e.g. tipo de documento, formato de página, formato de encabezados, paquetes a utilizar, entre otros.)

El preámbulo más sencillo consiste en una sola línea como se puede apreciar en el ejemplo 3.2.

Ejemplo 3.2: Preámbulo básico

```
1 \documentclass{article}
```

En donde **article** es la clase o tipo de documento, que por defecto puede ser:

article Usado para artículos científicos y textos simples.

letter Utilizado para crear cartas.

report Usado para redactar informes, monografías y documento de una sola cara.

book Empleado para libros, tesis y documentos a doble cara.

El conjunto KOMA-Script ofrece sustitutos para las clases por defecto con énfasis en la tipografía y la versatilidad.

scrartcl Usado para artículos científicos y textos simples.

scrlettr2 Utilizado para crear cartas.

scrreprt Usado para redactar informes, monografías y documento de una sola cara.

scrbook Empleado para libros, tesis y documentos a doble cara.

La principal ventaja de KOMA-Script es su alta capacidad de personalización, permitiendo modificar características del documento sin necesidad de paquetes adicionales.

Adicionalmente, el comando `\documentclass` admite parámetros adicionales, los cuales se colocan entre corchetes.

Los parámetros adicionales permiten modificar algunas opciones básicas del documento como el tamaño de letra, el tamaño de papel, orientación de página, entre otros.

El preámbulo mostrado en el ejemplo 3.3 inicia un documento tipo artículo, en papel A4 y letra de tamaño 11 puntos.

Ejemplo 3.3: Preámbulo básico con opciones

```
1 \documentclass[a4paper,11pt]{article}
```

3.2. Cuerpo del documento

La segunda parte de un documento es el **cuerpo del documento**, el cual se inicia con el comando `\begin{document}` y se cierra con `\end{document}`.

Aquí es donde se coloca el contenido y estructura del documento. Cada elemento dentro del cuerpo del documento corresponde con un elemento presente en el documento pdf final.

Todo texto fuera del cuerpo del documento es ignorado por el motor o interprete L^AT_EX durante la compilación, y por lo tanto, no aparecerá en el documento final.

En el ejemplo 3.4 se puede visualizar un documento sencillo en \LaTeX , consistente en dos párrafos. Observar que para indicar el final de un párrafo, se deja una línea en blanco (línea 4).

Ejemplo 3.4: Documento sencillo en \LaTeX

```

1 \documentclass[a4paper,12pt]{article}
2 \begin{document}
3     Este es un pequeño párrafo de ejemplo. Este documento es de tamaño A4 y
      ↪ usa un tamaño de letra de 12 puntos.
4
5     Para la creación de este documento, se ha utilizado la clase article.
6 \end{document}

```

3.3. Comandos y entornos

Un documento está constituido de diversos elementos, entre ellos los títulos de secciones, imágenes, tablas, listas, entre otros. Para poder definir estos elementos en \LaTeX , se utilizan **comandos** y **entornos**.

3.3.1. Comandos

Los comandos son instrucciones que tienen como objetivo producir un elemento o modificar la forma y apariencia de un elemento existente.

Los comandos empiezan con el símbolo `\`, seguido de uno o más letras, sin espacios entre si.

Por ejemplo, el comando `\LaTeX` produce \LaTeX .

Otros comandos, requieren adicionalmente **argumentos**, los cuales pueden ser obligatorios u opcionales. Los argumentos obligatorios van entre llaves y no pueden omitirse. Los argumentos opcionales, por otro lado, van entre corchetes y pueden ser omitidos, en cuyo caso tomarán los valores por defecto.

En el ejemplo 3.3, el comando `\documentclass` posee ambos tipos de argumentos.

Hay que considerar que los comandos que terminan en una letra, ignoran los espacios que le siguen. Esto puede observarse en el ejemplo 3.5, en donde la línea 1 produce una oración sin el espacio después de \LaTeX , mientras que las líneas 3, 5 y 7 son métodos válidos para evitar que el espacio desaparezca. Otra vez, se han dejado líneas en blanco para indicar el fin de cada párrafo, y el resultado puede apreciarse en el lado derecho.

Ejemplo 3.5: Espacio después de un comando

1	<code>\LaTeX</code> es un sistema de...	\LaTeX es un sistema de...
2		
3	<code>\LaTeX\</code> es un sistema de...	\LaTeX es un sistema de...
4		
5	<code>{\LaTeX}</code> es un sistema de...	\LaTeX es un sistema de...
6		
7	<code>\LaTeX{}</code> es un sistema de...	\LaTeX es un sistema de...

3.3.2. Entornos

Los entornos son estructuras iniciadas por el comando `\begin{entorno}` y terminadas por `\end{entorno}`. Es posible invocar comandos, o iniciar otros entornos (anidados) dentro de un entorno.

Los entornos pueden requerir argumentos obligatorios, los cuales se agregan entre llaves, mientras que los argumentos opcionales se agregan entre corchetes, con una sintaxis tal y como se muestra en el ejemplo 3.6.

Ejemplo 3.6: Sintaxis de un entorno

```

1  \begin{entorno}[opciones]{arg1}{arg2}...
2      Contenido...
3  \end{entorno}

```

Los entornos nos sirven para crear elementos, como tablas, figuras, listas, entre otros. Junto a los comandos, los entornos son los bloques constitutivos para armar un documento en \LaTeX .

En el ejemplo 3.7, se puede apreciar como utilizamos los entornos `itemize` y `enumerate` para crear listas. El resultado se puede apreciar en el lado derecho del ejemplo.

Ejemplo 3.7: Entornos de listas en L ^A T _E X	
1 <code>\begin{itemize}</code>	
2 <code>\item Gato</code>	■ Gato
3 <code>\item Conejo</code>	■ Conejo
4 <code>\item Perro</code>	■ Perro
5 <code>\end{itemize}</code>	
6	
7 <code>\begin{enumerate}</code>	
8 <code>\item Manzana</code>	1. Manzana
9 <code>\item Piña</code>	2. Piña
10 <code>\item Naranja</code>	3. Naranja
11 <code>\end{enumerate}</code>	
12	
13 <code>\begin{enumerate}</code>	
14 <code>\item Carne</code>	1. Carne
15 <code>\begin{itemize}</code>	■ Pollo
16 <code>\item Pollo</code>	■ Res
17 <code>\item Res</code>	
18 <code>\end{itemize}</code>	
19 <code>\item Verduras</code>	2. Verduras
20 <code>\begin{enumerate}</code>	a) Tomate
21 <code>\item Tomate</code>	b) Lechuga
22 <code>\item Lechuga</code>	c) Cebolla
23 <code>\item Cebolla</code>	
24 <code>\end{enumerate}</code>	
25 <code>\item Arroz</code>	3. Arroz
26 <code>\end{enumerate}</code>	

3.4. Paquetes

L^AT_EX posee cierta cantidad de comandos y entornos básicos, los cuales son provistos por el mismo L^AT_EX y la clase de documento utilizada. Con esto se cubren ciertas funcionalidades y necesidades básicas que podamos tener al momento de redactar un documento. Sin embargo, si se desea mayor control o más funcionalidades, se requerirá usar **paquetes** adicionales.

Los paquetes brindan nuevos comandos y entornos, expandiendo las capacidades de L^AT_EX. Además, algunos paquetes pueden cambiar el funcionamiento de comandos y entornos ya existentes.

Los paquetes son cargados en el **preámbulo**, utilizando para ello el comando `\usepackage[opciones]{pac`

Antes de ser cargados al documento, los paquetes deben estar instalados en la distribución L^AT_EX usada, sea Overleaf (en línea) o MiK_TE_X (instalación local). En el caso de una instalación local, si se ha seguido la guía de instalación del capítulo anterior, no es necesario preocuparse por esta parte: MiK_TE_X instalará automáticamente los paquetes

que falten cuando sean requeridos.

En el ejemplo 3.8 se tiene un documento en el que se ha cargado el paquete `babel` con las opciones `spanish` y `mexico`. El paquete `babel` ofrece soporte para escribir documentos en distintos idiomas.

Ejemplo 3.8: Cargando el paquete babel

```
1 \documentclass[a4paper,12pt]{article}
2 \usepackage[spanish,mexico]{babel}
3 \begin{document}
4   ...
5 \end{document}
```

Existen diversos paquetes, cada uno ofreciendo funcionalidades distintas. Por ejemplo el paquete `lipsum` nos permite crear texto de relleno (Lorem Ipsum) como el que aparece a continuación:

Aliquam lectus. Vivamus leo. Quisque ornare tellus ullamcorper nulla. Mauris porttitor pharetra tortor. Sed fringilla justo sed mauris. Mauris tellus. Sed non leo. Nullam elementum, magna in cursus sodales, augue est scelerisque sapien, venenatis congue nulla arcu et pede. Ut suscipit enim vel sapien. Donec congue. Maecenas urna mi, suscipit in, placerat ut, vestibulum ut, massa. Fusce ultrices nulla et nisl.

Esto podemos probarlo siguiendo el código del ejemplo 3.9.

Ejemplo 3.9: Documento con texto de relleno

```
1 \documentclass[a4paper,11pt]{article}
2 \usepackage{lipsum}
3 \begin{document}
4   \lipsum[13]
5 \end{document}
```

Del ejemplo anterior, la línea de código 2 carga el paquete `lipsum` mediante el comando `\usepackage` y la línea 4 usa el comando `\lipsum` que provee el paquete `lipsum`. Este comando acepta un argumento opcional, que en el ejemplo es el número 13. Esto indica que se imprimirá el párrafo número 13 del texto de relleno Lorem Ipsum. Prueba omitiendo el argumento opcional (usando solo `\lipsum`) o indicándole que muestre el párrafo 3 (`\lipsum[3]`), o incluso un rango de párrafos, como del párrafo 3 al 5 (`\lipsum[3-5]`).

3.4.1. Comprehensive T_EX Archive Network

Los paquetes se listan en la Comprehensive T_EX Archive Network, o CTAN, la cual puede ser visitada siguiendo el siguiente [enlace](#).

La CTAN no solo almacena más de 6300 paquetes, sino también su documentación, historial de actividad (actualizaciones y cambios), entre otros. Como usuarios de L^AT_EX, es importante aprender a utilizar la CTAN y saber revisar la documentación de los paquetes.

Los idiomas más utilizados en la CTAN son el inglés y el alemán, pues son los idiomas con mayor cantidad de usuarios de L^AT_EX. En caso se requiera un traductor en línea de buena calidad para poder revisar la documentación existente, se recomienda utilizar [DeepL](#).

Escribiendo un archivo .tex

En el capítulo anterior, visto durante la primera sesión, se vio que el punto de inicio para trabajar con \LaTeX es escribir uno (o más) archivos de extensión `.tex`. Este **archivo de entrada** era procesado por un **compilador o motor \LaTeX** y entregaba un archivo `.pdf` como resultado.

4.1. El archivo .tex

Es un archivo de **texto plano**, es decir, texto sin formato, que contiene una serie de instrucciones, comandos, entornos de \LaTeX .

Este archivo puede ser modificado por cualquier editor de texto, simple o especializado, pero debe ser procesado por un compilador \LaTeX para poder obtener el resultado final.

Un archivo `.tex` puede incluir o llamar a otros archivos similares. De esta manera, no es necesario tener un sólo archivo con el preámbulo y todo el contenido del documento, sino que puede ser modularizado o dividido en un archivo por capítulo.

4.2. Compiladores de \LaTeX

Toman el archivo o archivos `.tex` de entrada y los procesan para obtener el documento final. Los más conocidos son cuatro.

\LaTeX Compilador \TeX que utiliza el formato $\text{\LaTeX} 2_{\epsilon}$. Entrega un documento de salida en formato `DVI`.

pdf \LaTeX Compilador pdf \TeX que utiliza el formato $\text{\LaTeX} 2_{\epsilon}$. Entrega un documento de salida en formato `pdf`. Tiene soporte para modificar características microtipográficas.

Xe \LaTeX Compilador Xe \TeX que utiliza el formato $\text{\LaTeX} 2_{\epsilon}$. Entrega un documento de salida en formato `pdf`. Soporta UTF-8 por defecto e incluye características y

funciones para la composición de documento en diversos lenguajes, incluyendo lenguajes de escritura Derecha-Izquierda o con caracteres complejos como el arábigo. Adicionalmente, permite utilizar tipografías OpenType.

Lua \LaTeX Compilador Lua \TeX que utiliza el formato \LaTeX 2 ϵ . Entrega un documento de salida en formato pdf. Soporta UTF-8 por defecto e incluye características y funciones para la composición de documento en diversos lenguajes y con tipografías modernas, similar a Xe \LaTeX . Es el sucesor de pdf \LaTeX . Permite utilizar programación en Lua para automatizar tareas.

4.3. ¿Cómo interpreta \LaTeX al archivo .tex

En los ejemplos del capítulo anterior, vimos que podemos escribir caracteres alfabéticos de manera directa, e incluso colocar tildes y otros símbolos Unicode directamente (usando Xe \LaTeX o Lua \LaTeX).

Sin embargo, existen símbolos reservados para \LaTeX , que no podemos escribir de manera directa. A continuación veremos como interpreta \LaTeX el archivo .tex.

En la tabla 1 se listan los caracteres que pueden ser escritos directamente en \LaTeX . Es decir, aquellos que aparecen tal y como son escritos en el archivo .tex. También, se pueden escribir caracteres Unicode si se utiliza Lua \LaTeX o Xe \LaTeX .

Tabla 1: Caracteres que pueden ser escritos directamente

Tipo de carácter	Caracteres
Mayúsculas	A B C D E F G H I J K L M N Ñ O P Q R S T U V W X Y Z
Minúsculas	a b c d e f g h i j k l m n ñ o p q r s t u v w x y z
Números	0 1 2 3 4 5 6 7 8 9
Paréntesis	()
Corchetes	[]
Puntuación	, ; : ¡ ! . ¿ ?
Operadores	+ - * / =
Otros	@

En la tabla 2, se presentan los caracteres reservados que requieren el uso de comandos o métodos especiales para poder ser insertados correctamente en el texto. Además, se explica la función de cada uno de estos caracteres en \LaTeX . Por ejemplo, el símbolo \ se utiliza para marcar el inicio de un comando de \LaTeX . Si se desea colocar este símbolo en el texto, es necesario utilizar el comando `\textbackslash`.

Tabla 2: Caracteres que pueden ser escritos directamente

Carácter	Comando	¿Para qué lo usa \LaTeX ?
\$	<code>\\$</code>	Sirve para ingresar ecuaciones, activar el modo matemático.
%	<code>\%</code>	Le indica a \LaTeX que ignore el resto de la línea. Sirve para agregar comentarios.
{ }	<code>\{ \}</code>	Crea grupos e indica argumentos obligatorios de los comandos.
_	<code>_</code>	Crea subíndices en modo matemático.
^	<code>\^{}{ }</code>	Crea superíndices en modo matemático.
&	<code>\&</code>	Indica salto de columna en tablas.
#	<code>\#</code>	Sirve para identificar los argumentos de un comando personalizado, cuando este se crea.
\	<code>\textbackslash</code>	Inicio de comando o macro de \LaTeX .
~	<code>\~{}{ }</code>	Espacio no separable. Une dos palabras y evita que aparezcan en líneas distintas. Además sirve para crear la ñ a partir de la n.

A continuación veremos un ejemplo. Se desea escribir el texto siguiente.

Este es un texto de ejemplo, con el que analizaremos los comandos que debemos utilizar para componerlo, incluyendo caracteres especiales.

El 20 % de \$ 300.00, es igual a \$ 60.00.

Las llaves en \LaTeX , sirven para crear grupos y argumentos: {grupo 1} {grupo 2}.

El texto anterior contiene caracteres especiales, lo que requiere el uso de comandos específicos para poder escribirlos correctamente. Además, para mantener la estructura de párrafos, es necesario agregar líneas en blanco correspondientes en los lugares adecuados. La solución se puede ver en la sección ejemplo 4.1. Es importante destacar el uso del símbolo % para insertar comentarios y mejorar la claridad del código.

Ejemplo 4.1: Componiendo un texto con caracteres especiales

```
1 \documentclass{article}
2 \usepackage[spansh,mexico]{babel} % Carga babel con idioma español.
3 \begin{document}
4 Este es un texto de ejemplo, con el que analizaremos los comandos que
  ↳ debemos utilizar para componerlo, incluyendo caracteres especiales.
5
6 El 20\% de \$ 300.00, es igual a \$ 60.00.
7
8 Las llaves en \LaTeX{} sirven para crear grupos
9 y argumentos: {\grupo 1} {\grupo 2}.
10 \end{document}
```

Más adelante veremos como estructurar un documento en múltiples archivos .tex, para mejorar el orden, así como la creación de comandos y entornos personalizados para facilitar la redacción de documentos complejos o repetitivos.

Secciones de un documento

Generalmente la preparación de un documento sigue una estructura lógica de niveles. Por ejemplo, podemos dividir un documento en capítulos, que a su vez se dividen en secciones, y éstas pueden contener sub-secciones, y así sucesivamente.

En este capítulo aprenderemos a implementar la estructura de un documento usando comandos de seccionado, crear una página de título y elaborar una tabla de contenidos automática.

5.1. Página de título

El comando `\maketitle` genera una sección de título, la cual puede ocupar una página completa (clases `book` y `report`) o sólo una parte de esta (clase `article`).

Las clases anteriormente mencionadas generan esta sección de título a partir de información que se declara en el preámbulo. La información necesaria es:

- Título del documento
- Nombre del autor (o autores)
- Fecha de creación

En el ejemplo 5.1 se puede apreciar la implementación de una página de título para un documento de dos autores. Los comandos de \LaTeX admiten espacios y saltos de línea entre sus argumentos, por lo que podemos aprovechar esto para tener un código más ordenado. El comando `\author`, iniciado en la línea 2 del ejemplo 5.1 termina recién en la línea 10.

Múltiples autores se separan utilizando el comando `\and`, y el comando `\thanks` sirve para insertar una nota al pie de página con información adicional (puede ser el correo, afiliación, etc.) Para insertar líneas múltiples se utiliza `\\`.

Ejemplo 5.1: Página de título

```

1 \documentclass[a4paper, 11pt]{article}
2 \author{
3     Primer Autor
4     \thanks{Institución del Primer Autor}\\
5     correo@mail.com
6     \and
7     Segundo Autor
8     \thanks{Institución del Segundo Autor}\\
9     correo@mail.com
10 }
11 \title{Título de ejemplo\\Curso de \LaTeX{}}
12 \date{20 de enero}
13 \begin{document}
14     \maketitle
15 \end{document}

```

5.2. Comandos de seccionado

Las clases de documento por defecto de \LaTeX y KOMA-Script, ofrecen diversos comandos para implementar dicha estructura lógica.

1. `\part` - Corresponde al nivel 0
2. `\chapter` - Define capítulos, correspondientes al nivel 1. La clase `article` no dispone de este comando.
3. `\section` - Corresponde al nivel 2.
4. `\subsection` - Corresponde al nivel 3.
5. `\subsubsection` - Corresponde al nivel 4.
6. `\paragraph` - Corresponde al nivel 5.
7. `\subparagraph` - Corresponde al nivel 6.

Por defecto, las clases `book` y `report` enumera los niveles 1, 2 y 3, mientras que la clase `article` enumera a los niveles 2, 3 y 4.

A continuación veremos un ejemplo de como implementar una estructura de documento utilizando los comandos anteriormente mencionados. Además, usaremos el paquete `lipsum` para generar texto de relleno.

Ejemplo 5.2: Ejemplo de secciones de una investigación

```

1 \documentclass[a4paper,11pt]{book}
2 \usepackage{lipsum}
3 \title{Redes sociales y sociedad actual}
4 \author{Juan Molina}
5 \date{\today}
6 \begin{document}
7 \maketitle
8 \chapter{Introducción}
9   \section{Antecedentes y justificación del estudio}
10  \lipsum[1-3]
11  \section{Preguntas de investigación y objetivos}
12  \lipsum[13-14]
13  \section{Metodología utilizada en el estudio}
14  \lipsum[4-5]
15
16 \chapter{Marco teórico}
17   \section{Evolución y desarrollo de las redes sociales}
18   \lipsum[13]
19   \subsection{Origen y evolución histórica de las redes sociales}
20   \lipsum[4-5]
21   \subsection{Tipos de redes sociales y características}
22   \lipsum[5-6]
23   \subsection{Funcionamiento de las redes sociales}
24   \lipsum[10-11]
25   \section{Impacto de las redes sociales en la sociedad actual}
26   \subsection{Aspectos positivos y negativos}
27   \lipsum[1-2]
28   \subsection{Efectos psicológicos y emocionales}
29   \lipsum[3-4]
30   \subsection{Influencia de las redes sociales en la cultura y el
31     ↳ comportamiento social}
32   \lipsum[2-3]
33
34 \chapter{Análisis y resultados}
35   \section{Análisis estadístico de los datos recopilados}
36   \lipsum[10-11]
37   \section{Resultados y discusión de los hallazgos}
38   \lipsum[12-13]
39   \subsection{Uso de las redes sociales en la población estudiada}
40   \lipsum[25]
41   \subsection{Relación entre el uso de las redes sociales y la salud
42     ↳ mental}
43   \lipsum[26]
44   \section{Opiniones y percepciones de los usuarios sobre el impacto de
45     ↳ las redes sociales}
46   \lipsum[16-17]
47
48 \chapter{Conclusiones y recomendaciones}
49   \lipsum[21-23]
50 \end{document}

```

5.3. Tabla de contenidos

Si las secciones de un documento son creadas utilizando los comandos mostrados anteriormente, podemos utilizar el comando `\tableofcontents` para generar una tabla de contenidos automática. En esta aparecerán, por defecto, como máximo 4 niveles de seccionado (incluyendo “partes”). Es decir, para las clases `book` y `report`, solo aparecerán hasta las subsecciones, mientras que para la clase `article` también aparecerán las subsubsecciones.

En el ejemplo 5.2 algunos títulos son demasiado extensos. \LaTeX permite especificar títulos alternativos que sólo serán utilizados para la tabla de contenidos. La sintaxis es la siguiente: `\comandoseccion[título corto]{título largo}`, en donde el comando `\comandoseccion` puede ser cualquiera de los vistos en la sección anterior. El título corto aparecerá en la tabla de contenidos mientras que el título largo en el cuerpo del documento.

Los comandos de seccionado también tienen una variante estrellada, es decir `\chapter` tiene una variante `\chapter*`. Estas variantes crean una sección que no posee numeración, y que no aparece en la tabla de contenidos.

5.4. Consideraciones adicionales con la clase book

La clase `book` posee 3 comandos que sirven para delimitar las 3 partes tradicionales de un libro:

`\frontmatter` Pensado para incluir la hoja de título, dedicatoria, prólogo, prefacio, índices, entre otras páginas preliminares.

`\mainmatter` El cuerpo principal del libro. Principalmente los capítulos.

`\backmatter` Pensado para incluir glosario, notas, índice de términos y otras páginas complementarias.

En la `frontmatter`, los páginas se numeran usando números romanos, y los comandos de seccionado no son numerados (similar a las variantes estrelladas), pero si se agregan a la tabla de contenidos.

Cuando inicia la `mainmatter`, el número de página se reinicia y usa números arábigos. Las secciones nuevamente son enumeradas y aparecen en la tabla de contenidos (funcionamiento normal).

Finalmente, en la `backmatter`, los capítulos nuevamente dejan de ser numerados (dejan de etiquetarse), pero se conserva la numeración.

5.5. Apéndices

Existe la posibilidad de agregar apéndices en \LaTeX gracias al comando `\appendix`. Este comando modifica el nombre de los capítulos (Capítulo 1, Capítulo 2, etc.) a apéndices, y cambia su numeración a alfabética (Apéndice A, Apéndice B, etc.).

\LaTeX permite redefinir el formato visual de los elementos textuales del documento mediante comandos y entornos. En este capítulo veremos como se implementan algunos cambios de formato de texto.

6.1. Familia tipográfica de texto

La familia tipográfica define las características generales de los caracteres. Existen tres familias tipográficas definidas para \LaTeX : Roman, Sans-Serif y Typewriter.

6.1.1. Familia Roman

Las tipografías Roman o Serif, tienen como característica predominante unos remates, o serifas, que son como adornos en los caracteres.

Esta oración está utilizando la familia Roman (Serif).

6.1.2. Familia Sans-Serif

Las tipografías Sans-Serif, por el contrario, carecen de estos adornos o remates.

Esta oración está utilizando la familia Sans-Serif.

6.1.3. Familia Typewriter

Las tipografías monoespaciadas o Typewriter utilizan caracteres de ancho fijo. Por ejemplo, la letra i y la letra m ocupan el mismo ancho.

Esta oración usa la familia Typewriter.

6.2. Estilos de texto

6.2.1. Serie

Establece el grosor o peso de los caracteres. Existen dos series definidas en \LaTeX : medio y **negrita**.

6.2.2. Forma

Define la forma de los caracteres. \LaTeX define 4 formas por defecto: la forma vertical, *cursiva*, *inclinada* y VERSALITAS.

6.3. Comandos para dar formato al texto

Existen dos tipos de comandos para dar formato al texto, aquellos que sólo aplican el formato a su argumento, y aquellos que funcionan como un interruptor (declaración) y aplican el formato desde su aparición hasta el final del grupo o entorno. En la tabla 3 se brindan los comandos para aplicar los formatos anteriormente mencionados.

Tabla 3: Comandos para cambiar el formato del texto

Estilo	Comando	Declaración
Roman	<code>\textrm{}</code>	<code>\rmfamily</code>
Sans	<code>\textsf{}</code>	<code>\sffamily</code>
Typewriter	<code>\texttt{}</code>	<code>\ttfamily</code>
Medio	<code>\textmd{}</code>	<code>\mdseries</code>
Negrita	<code>\textbf{}</code>	<code>\bfseries</code>
Vertical	<code>\textup{}</code>	<code>\upshape</code>
Cursiva	<code>\textit{}</code>	<code>\itshape</code>
Inclinada	<code>\textsl{}</code>	<code>\slshape</code>
Versalitas	<code>\textsc{}</code>	<code>\scshape</code>

Muchos de estos comandos pueden combinarse, las únicas limitaciones vienen a ser las siguientes:

- No pueden combinarse estilos de una misma categoría, es decir un texto no puede estar en cursiva y versalitas al mismo tiempo. Pero si puede existir un texto en negrita y cursiva: ***Ejemplo.***

- Algunas tipografías no poseen algunas combinaciones. Por ejemplo, la mayoría no posee versalitas en negrita, pero otras sí: **EJEMPLO**.
- Los comandos tipo declaración, se pueden combinar colocando uno detrás de otro, mientras que los comandos con argumento se anidan uno dentro de otro:
 - Combinando declaraciones: `{\bfseries\itshape texto}`
 - Combinando comandos `\textbf{\textit{texto}}`

6.4. Cambiando las tipografías por defecto

L^AT_EX utiliza la tipografía **Computer Modern** por defecto con sus respectivas variaciones para cada familia definida anteriormente. Si deseamos utilizar una tipografía específica, podemos indicarle a L^AT_EX que cargue paquetes de tipografías y las use en lugar de Computer Modern, o podemos usar el paquete `fontspec` para usar las tipografías que tengamos disponibles en nuestra pc (o el servidor de Overleaf).

6.4.1. Tipografías usando paquetes

Como su nombre lo indica, este método consiste en cargar paquetes específicos de tipografías para poder usarlas en nuestro documento, en reemplazo de las tipografías por defecto.

Este método es compatible con todos los compiladores o motores de L^AT_EX mencionados en capítulos anteriores (pdfL^AT_EX, XeL^AT_EX, LuaL^AT_EX, etc.).

Si usamos pdfL^AT_EX, es necesario cambiar la codificación de la tipografía por defecto, utilizando el paquete `fontenc`, con la opción `T1`. En cambio, si usamos un compilador moderno como LuaL^AT_EX este paso no será necesario.

Las fuentes disponibles se listan en el [Catálogo de Fuentes de L^AT_EX](#).

Las tipografías o fuentes se agrupan de acuerdo a su familia tipográfica, su soporte de matemáticas, y otras características. Por ejemplo, si deseamos utilizar la tipografía Merriweather como familia Roman, y Merriweather Sans como Sans Serif, debemos cargar el paquete `merriweather`, como se ilustra en el ejemplo 6.1. Como se mencionó anteriormente, si usamos pdfL^AT_EX, también es necesario cargar el paquete `fontenc` con la opción `T1` (línea 3).

Ejemplo 6.1: Usando la tipografía Merriweather

```

1 \documentclass[a4paper,12pt]{article}
2 \usepackage{merriweather}
3 %\usepackage[T1]{fontenc} % Usar sólo con pdflatex
4 \usepackage{lipsum}
5 \begin{document}
6   \section{Roman}
7   \lipsum[5-6]
8   \section{SansSerif}
9   \sffamily
10  \lipsum[5-6]
11 \end{document}

```

6.4.2. Tipografías con fontspec

Este método nos permite utilizar tipografías TrueType u OpenType que tengamos instaladas en nuestro ordenador.

Para ello se requiere cargar el paquete `fontspec` y hacer uso de los siguientes comandos para definir las tipografías por defecto.

`\setmainfont{fuente}`: Reemplaza la familia Roman / Serif por `fuente`.

`\setsansfont{fuente}`: Reemplaza la familia SansSerif por `fuente`.

`\setmonofont{fuente}`: Reemplaza la familia Typewriter por `fuente`.

En el ejemplo 6.2 podemos ver como utilizamos este método para cargar las tipografías Times New Roman, Arial e Inconsolata para nuestro documento. Adicionalmente, la tipografía Arial se ha escalado en un 91 % para reducir su tamaño natural.

Ejemplo 6.2: Usando fontspec

```

1 \documentclass[a4paper,12pt]{article}
2 \usepackage{fontspec}
3 \setmainfont{Times New Roman}
4 \setsansfont[Scale=0.91]{Arial}
5 \setmonofont{Inconsolata}
6 \usepackage{lipsum}
7 \begin{document}
8   \section{Times New Roman}
9   \lipsum[5-6]
10  \section{Arial}
11  \sffamily
12  \lipsum[5-6]
13  \section{Inconsolata}

```

```

14 \ttfamily
15 \lipsum[13]
16 \end{document}

```

6.5. Tamaño de texto

6.5.1. Tamaño base o normal

El tamaño base del texto, se define como opción de la clase de documento, usando el comando `\documentclass`.

Las clases por defecto de \LaTeX permiten seleccionar entre 10pt, 11pt o 12pt. Las clases de KOMA-Script permiten utilizar los tamaños 8pt, 9pt, 10pt, 12pt, 14pt, 17pt o 20pt.

Este es el tamaño normal o de referencia para el texto, a partir del cual se calculan los tamaños relativos.

6.5.2. Tamaños relativos de texto y tamaños absolutos

En \LaTeX existen comandos que cambian el tamaño del texto de manera relativa al tamaño de referencia del documento. Por ejemplo, en un documento con un tamaño base de 11pt, el comando `\Large` producirá texto con un tamaño menor al mismo comando usado en un documento de 12pt.

Esto es útil pues generalmente se suele utilizar el tamaño del texto para dar énfasis o diferenciar entre distintos niveles de estructura. En el ejemplo 6.3 se pueden apreciar los comandos para cambiar el tamaño del texto. Estos comandos son tipo declaración, por lo que si se requiere limitar su alcance, se recomienda crear grupos usando llaves.

Sin embargo, también se puede definir el tamaño (y separación) del texto usando el comando `\fontsize{size}{bskip}`, en donde `size` se refiere al tamaño del texto deseado y `bskip` a la separación entre las líneas de un párrafo.

Para que el comando anterior tenga efecto, es necesario seleccionar o cargar de nuevo la fuente. Por esta razón, se recomienda añadir el comando `\selectfont` al final. Para ilustrar esto, puede observarse una muestra en el ejemplo 6.4.

Ejemplo 6.3: Tamaños relativos

1	<code>\tiny</code>	<code>tiny</code>	<code>tiny</code>
2			<code>scriptsize</code>
3	<code>\scriptsize</code>	<code>scriptsize</code>	<code>footnotesize</code>
4			<code>small</code>
5	<code>\footnotesize</code>	<code>footnotesize</code>	<code>normalsize</code>
6			<code>large</code>
7	<code>\small</code>	<code>small</code>	<code>Large</code>
8			<code>LARGE</code>
9	<code>\normalsize</code>	<code>normalsize</code>	<code>huge</code>
10			<code>Huge</code>
11	<code>\large</code>	<code>large</code>	
12			
13	<code>\Large</code>	<code>Large</code>	
14			
15	<code>\LARGE</code>	<code>LARGE</code>	
16			
17	<code>\huge</code>	<code>huge</code>	
18			
19	<code>\Huge</code>	<code>Huge</code>	
20			

Ejemplo 6.4: Tamaños absolutos

1	<code>{\fontsize{11}{18}\selectfont</code>	
	↪ Texto tamaño 11 y espacio 18.	
	↪ Este párrafo debe terminarse	
	↪ para que el efecto sea	
	↪ visible. Aquí el párrafo se	
	↪ terminó dejando una línea	
	↪ libre al final.	
2		
3	<code>}</code>	
4		
5	<code>\rule{\linewidth}{0.5pt}</code>	
6		
7	<code>{\fontsize{11}{18}\selectfont</code>	En
	↪ este otro párrafo, no se ha	
	↪ terminado de manera correcta	
	↪ dejando una línea vacía, por	
	↪ lo que la separación será	
	↪ incorrecta.}	

Texto tamaño 11 y espacio 18. Este párrafo debe terminarse para que el efecto sea visible. Aquí el párrafo se terminó dejando una línea libre al final.

En este otro párrafo, no se ha terminado de manera correcta dejando una línea vacía, por lo que la separación será incorrecta.

6.6. Formato de párrafo

En \LaTeX , es posible personalizar el formato de los párrafos para que cumplan con ciertas características tipográficas deseables. Por ejemplo, se pueden ajustar el espaciado entre líneas, el tamaño y tipo de fuente (visto en la sección anterior), el sangrado de las

primeras líneas y la separación entre párrafos. Para cambiar estas características, se utilizan diferentes comandos y entornos.

En los próximos párrafos, exploraremos algunas de estas opciones con más detalle.

6.6.1. Alineación de párrafo

La alineación horizontal del párrafo puede ser de 4 tipos, alineado a la izquierda, a la derecha, al centro, o justificado (por defecto).

Mediante el paquete `ragged2e` se tiene acceso a comandos tipo declaración y entornos que son mejores que los incluidos por defecto en \LaTeX .

Alineación a la izquierda

El primer tipo de alineación es la alineación a la izquierda, como se puede apreciar en el siguiente párrafo.

La música sonaba en el bar, inundando el ambiente con su ritmo y sus melodías. Las luces de neón parpadeaban en la noche, dibujando figuras caprichosas en las paredes. La gente bailaba al son de la música, moviendo sus cuerpos al compás de la noche. El aire estaba cargado de energía y emoción, y todos parecían estar disfrutando al máximo la experiencia. Era una noche llena de vida y de alegría, una noche para recordar para siempre.

Para justificar el texto a la izquierda, se pueden utilizar tanto el entorno `FlushLeft` como el comando `\RaggedRight`. Sin embargo, hay que tener en cuenta que el entorno `FlushLeft` agrega espacio vertical antes y después del párrafo, lo que puede afectar la apariencia general del texto.

Por otro lado, el comando `\RaggedRight` funciona como una declaración y no agrega espacio adicional al texto. Su uso es similar al de otros comandos tipo declaración mencionados previamente, como `\bfseries` o `\itshape`, y su efecto se mantiene hasta que se le da otro comando que lo anule o se cierra el grupo donde se encuentra.

Alineación a la derecha

El siguiente párrafo tiene una alineación a la derecha.

La música sonaba en el bar, inundando el ambiente con su ritmo y sus melodías. Las luces de neón parpadeaban en la noche, dibujando figuras en las paredes. La gente bailaba al son de la música, moviendo sus cuerpos al compás de la noche. El aire estaba cargado de energía y emoción, y todos parecían estar disfrutando al máximo la experiencia.

De manera similar a la alineación a la izquierda, también existen opciones para alinear el texto a la derecha en L^AT_EX. Para ello, se pueden utilizar el entorno `FlushRight` o el comando `\RaggedLeft`. Es importante tener en cuenta que, al igual que en el caso anterior, el uso del entorno agrega espacio vertical antes y después del párrafo, mientras que el comando funciona como una declaración y no agrega espacio adicional al texto.

Centrado

El siguiente párrafo está centrado usando el entorno `Center`.

La música sonaba en el bar, inundando el ambiente con su ritmo y sus melodías. Las luces de neón parpadeaban en la noche, dibujando figuras caprichosas en las paredes. La gente bailaba al son de la música, moviendo sus cuerpos al compás de la noche.

El comando correspondiente para centrar el texto es `\Centering`.

Justificado

Finalmente, el texto justificado se puede implementar mediante el entorno `justify` y/o el comando `\justifying`.

La música sonaba en el bar, inundando el ambiente con su ritmo y sus melodías. Las luces de neón parpadeaban en la noche, dibujando figuras caprichosas en las paredes. La gente bailaba al son de la música, moviendo sus cuerpos al compás de la noche. El aire estaba cargado de energía y emoción, y todos parecían estar disfrutando al máximo la experiencia. Era una noche llena de vida y de alegría, una noche para recordar para siempre.

6.6.2. Interlineado

Para el interlineado, se suele emplear el paquete `setspace`, el cual permite utilizar los comandos `\singlespacing`, `\onehalfspacing` y `\doublespacing`, para lograr un

espaciado simple, de 1.5 y doble, respectivamente. Para valores distintos, también está disponible el comando `\setstretch{valor}`. Se sugiere el uso de estos comandos en el preámbulo, para que afecten de manera total al documento.

Adicionalmente, si en el cuerpo del documento se desean secciones de distinto espaciado al definido con los comandos anteriores, `setspace` define los entornos `singlespace`, `onehalfspace` y `doublespace`. De igual forma, para un entorno con un espaciado distinto, se puede usar el entorno `spacing`, el cual requiere como argumento el valor numérico del espaciado. El ejemplo 6.5 ilustra esto.

Ejemplo 6.5: Ejemplo de espaciados

```

1 \begin{spacing}{1}
2   Espaciado 1: \lipsum[75]
3 \end{spacing}
4 \begin{spacing}{1.2}
5   Espaciado 1.2: \lipsum[66]
6 \end{spacing}
7 \begin{spacing}{1.4}
8   Espaciado 1.4: \lipsum[75]
9 \end{spacing}

```

Espaciado 1: Pellentesque interdum sapien sed nulla. Proin tincidunt. Aliquam volutpat est vel massa. Sed dolor lacus, imperdiet non, ornare non, commodo eu, neque. Integer pretium semper justo. Proin risus. Nullam id quam. Nam neque. Duis vitae wisi ullamcorper diam congue ultricies. Quisque ligula. Mauris vehicula.

Espaciado 1.2: Nunc sed pede. Praesent vitae lectus. Praesent neque justo, vehicula eget, interdum id, facilisis et, nibh. Phasellus at purus et libero lacinia dictum. Fusce aliquet. Nulla eu ante placerat leo semper dictum. Mauris metus. Curabitur lobortis. Curabitur sollicitudin hendrerit nunc. Donec ultrices lacus id ipsum.

Espaciado 1.4: Pellentesque interdum sapien sed nulla. Proin tincidunt. Aliquam volutpat est vel massa. Sed dolor lacus, imperdiet non, ornare non, commodo eu, neque. Integer pretium semper justo. Proin risus. Nullam id quam. Nam neque. Duis vitae wisi ullamcorper diam congue ultricies. Quisque ligula. Mauris vehicula.

6.6.3. Espaciado entre párrafos y sangría en primera línea

Existen dos métodos para poder diferenciar cuando un párrafo termina y empieza uno nuevo. El primero consiste en dejar espacio vertical entre párrafos, mientras que el segundo método consiste en agregar un espacio (sangría) a la primera línea de cada párrafo. Por defecto, \LaTeX utiliza el primer método.

Para implementar cualquiera de los dos métodos se requiere cargar el paquete `parskip`. Este paquete cuenta con diversas opciones, y a continuación veremos algunas de ellas.

- `skip` Especifica la separación vertical entre párrafos (`\parskip`).
- `indent` Especifica la sangría de primera línea (`\parindent`).
- `tocskip` Especifica la separación en la tabla de contenidos.

Suponiendo que se desea aplicar una sangría de 1.25 cm a la primera línea de cada párrafo, y además, una separación de 6 pt entre párrafos, se puede implementar tal y como lo ilustra el ejemplo 6.6. Se ha usado el paquete `lipsum` para generar texto de relleno.

Ejemplo 6.6: Usando el paquete parskip

```

1 \documentclass{article}
2 \usepackage[indent=1.25cm,skip=6pt]{parskip}
3 \usepackage{lipsum}
4 \begin{document}
5   \section{Una sección}
6   \lipsum[1-3]
7   \section{Otra sección}
8   \lipsum[4-6]
9 \end{document}
```

El ejemplo anterior podemos mejorarlo si le damos cierta tolerancia a la distancia vertical entre párrafos, como se ilustra en el ejemplo 6.7. En este caso, la distancia vertical máxima entre párrafos será de 8 pt y la mínima de 5 pt, lo que brinda a \LaTeX un mejor margen de ajuste para componer las páginas del documento.

Ejemplo 6.7: Usando el paquete parskip 2

```

1 \documentclass{article}
2 \usepackage[indent=1.25cm,skip=6pt plus 2pt minus 1pt]{parskip}
3 \usepackage{lipsum}
4 \begin{document}
5   \section{Una sección}
6   \lipsum[1-3]
```




















```

7 \section{Otra sección}
8 \lipsum[4-6]
9 \end{document}

```

6.7. Color de texto

Para poder editar el color del texto, se requiere usar el paquete `xcolor`, el cual define los siguientes colores por defecto:

 red,
  green,
  blue,
  cyan,
  magenta,
  yellow,
  black,
  gray,
  white,
  darkgray,
  lightgray,
  brown,
  lime,
  olive,
  orange,
  pink,
  purple,
  violet, y
  teal.

Y permite cargar colores adicionales si se utilizan los siguientes argumentos opcionales:

- `dvipsnames`: 68 colores en formato CMYK.
- `svgnames`: 151 colores en formato rgb.
- `x11names`: 317 colores en formato rgb.

Los cuales pueden consultarse en la [pagina 39](#) de su documentación.

El paquete `xcolor` permite utilizar los siguientes comandos:

- `\textcolor{color}{texto}`: Colorea el `texto` del color `color`.
- `\color{color}`: Sirve como declaración de color de texto, vigente en el grupo actual o hasta que se vuelva a declarar el color de texto.
- `\pagecolor{color}`: Declara el color de la página actual y las páginas siguientes, o hasta que vuelva a ser declarado.
- `\nopagecolor`: Revierte la declaración de color de página a su valor por defecto (transparente).
- `\colorbox{color}{texto}`: Crea una caja de texto `texto` con color de fondo `color`.

- `\fcolorbox{marco}{fondo}{texto}`: Crea una caja de texto `texto` con color de fondo `fondo` y marco de color `marco`.
- `\mathcolor{color}{math}`: Similar al comando `\textcolor` pero para entornos del modo matemático.
- `\definecolor{nombre}{modelo}{spec}`: Define el color `nombre`. El `modelo` puede ser rgb, cmyk, hsb, HTML, entre otros.

Para ilustrar su uso, en el ejemplo 6.8 se muestran algunas oraciones con efectos de color de texto.

Ejemplo 6.8: Algunos ejemplos con colores

```

1 \begin{Center}
2 \textcolor{blue}{Texto en color azul}
3
4 {\color{red} Este grupo está en color rojo}
5
6 \colorbox{pink}{Una caja de color rosado}
7
8 \fcolorbox{blue}{blue!10}{Esta caja de texto tiene borde azul y fondo azul
  ↳ al 10\%}
9 \end{Center}

```

Texto en color azul
 Este grupo está en color rojo
 Una caja de color rosado

Esta caja de texto tiene borde azul y fondo azul al 10 %

Finalmente, en el ejemplo 6.9 se ilustra el uso de los comandos de definición de color. En particular, en dicho ejemplo se implementa una paleta de colores accesible para personas con daltonismo, basada en sugerencias de la IBM Design Library, recogidas por David Nichols en su artículo [Coloring for Colorblindness](#).

Ejemplo 6.9: Definiendo colores

```

1 \documentclass[a4paper, 11pt]{article}
2 \usepackage{xcolor}
3 % Definiendo colores
4 \definecolor{IBM_azul}{HTML}{648FFF}
5 \definecolor{IBM_lila}{HTML}{785EF0}
6 \definecolor{IBM_rosa}{HTML}{DC267F}
7 \definecolor{IBM_naranja}{HTML}{FE6100}
8 \definecolor{IBM_dorado}{HTML}{FFB000}
9 \begin{document}
10 \bfseries

```

```
11 \textcolor{IBM_azul}{Texto en color IBM Azul}
12
13 \textcolor{IBM_lila}{Texto en color IBM Lila}
14
15 \textcolor{IBM_rosa}{Texto en color IBM Rosa}
16
17 \textcolor{IBM_naranja}{Texto en color IBM Naranja}
18
19 \textcolor{IBM_dorado}{Texto en color IBM Dorado}
20 \end{document}
```

6.8. Conclusiones

En este capítulo se han presentado diversos paquetes, comandos y entornos para modificar el formato de texto. Usar estos comandos de manera directa en el documento puede resultar tedioso y va en contra del principio de separación de estructura lógica y estilo visual propuesto por \LaTeX . Por esto, en el siguiente capítulo, veremos como implementar comandos y entornos personalizados que resulten en un documento **semántico**. Esto permitirá que el estilo visual del documento sea fácilmente modificado, simplemente cambiando la definición de dichos comandos y entornos.