Personalizar LyX: Características para el usuario avanzado

por el equipo LyX*

Versión 2.3.x

14 de enero de 2021

*Si tienes correcciones o comentarios, envíalos, por favor, a la lista de correo de Documentación de LyX, lyx-docs@lists.lyx.org. Incluye «[Customization]» en la cabecera de asunto, y envía una cc al actual mantenedor de este documento, Richard Heck <rgheck@comcast.net>.

Índice general

1. Introducción **1**

2. Archivos de configuración de LyX **3**
2.1. ¿Qué hay en LyXDir? . . . 3
2.1.1. Archivos generados automáticamente . . . 3
2.1.2. Directorios . . . 4
2.1.3. Archivos que no necesitan modificaciones . . . 5
2.1.4. Otros archivos en un par de líneas . . . 5
2.2. Directorio de configuración personal . . . 5
2.3. Ejecutar LyX con múltiples configuraciones . . . 6

3. El diálogo Preferencias **7**
3.1. Formatos . . . 7
3.2. Copiadores . . . 8
3.3. Convertidores . . . 9

4. Internacionalización de LyX **13**
4.1. Traducción de LyX . . . 13
4.1.1. Traducción de la interfaz gráfica de usuario (mensajes de texto) 13
4.1.1.1. Mensajes ambiguos . . . 14
4.1.2. Traducción de la documentación . . . 14
4.2. Cosas sobre teclados internacionales . . . 15
4.2.1. El archivo .kmap . . . 16
4.2.2. El archivo .cdef . . . 17
4.2.3. Teclas muertas . . . 18
4.2.4. Guardar la configuración de idioma . . . 18

5. Instalación de clases, formatos ... **19**
5.1. Instalación de nuevos archivos LaTeX . . . 20
5.2. Tipos de archivos de formato . . . 21
5.2.1. Módulos de formato . . . 22
5.2.1.1. Formato local . . . 23
5.2.2. Formato para archivos .sty . . . 23
5.2.3. Formato para archivos .cls . . . 25
5.2.4. Creación de plantillas . . . 25
5.2.5. Actualización de antiguos archivos de formato . . . 26
5.2.6. Cite engine files . . . 26

5.3. Estructura del archivo 'layout' 26
5.3.1. Declaración de la clase de documento y clasificación 27
5.3.2. Declaración de un módulo 29
5.3.3. The CiteEngine file declaration 29
5.3.4. Número de formato 30
5.3.5. Parámetros generales de clases de texto 30
5.3.6. Sección `ClassOptions` 34
5.3.7. Estilos de párrafo 34
5.3.8. Internacionalización de estilos de párrafo 44
5.3.9. Flotantes 45
5.3.10. Recuadros flexibles y formato del recuadro 47
5.3.11. Contadores 52
5.3.12. Descripción de la tipografía 53
5.3.13. Citation engine description 53
5.3.14. Descripción del formato de cita 55
5.4. Etiquetas para la salida XHTML 59
5.4.1. Estilos de párrafo 59
5.4.2. Recuadros XHTML 61
5.4.3. Flotantes XHTML 62
5.4.4. Formato de la Bibliografía 63
5.4.5. CSS generado por LyX 63

6. Inserción de material externo **65**
6.1. ¿Cómo funciona? 65
6.2. El archivo de configuración de plantillas externas 66
6.2.1. La cabecera de la plantilla 67
6.2.2. La sección Format 68
6.2.3. Definiciones de preámbulo 69
6.3. El mecanismo de sustitución 70
6.4. Discusión sobre seguridad 72

A. Lista de funciones soportadas por LyX en archivos de formato **75**

B. Nombres de colores disponibles para archivos de formato **77**
B.1. Color functions 77
B.2. Static colors 77
B.3. Dynamic colors 78

1. Introducción

Este manual trata de las características de LyX que pueden ser modificadas por el usuario. En él abordamos temas como atajos de teclado, opciones de vista previa en pantalla, opciones de impresora, envío de comandos mediante LyX Server, internacionalización, instalación de nuevas clases de LaTeX y de formatos de LyX, etc. Seguro que no podemos aspirar a tratar todo lo que se puede cambiar, —nuestross desarrolladores añaden características nuevas más deprisa de lo que las podemos documentar— pero explicaremos las modificaciones más comunes y y esperamos orientarte en la dirección correcta para algunas más desconocidas.

2. Archivos de configuración de LyX

Este capítulo tiene por objetivo ayudarte a encontrar tu camino a través de los archivos de configuración de LyX. Antes de seguir leyendo deberías encontrar donde están los directorios de bibliotecas y de usuario, consultando Ayuda ▷ Acerca de LyX. El directorio de bibliotecas es el sitio en el que LyX tiene sus archivos de configuración de sistema; el directorio de usuario es donde puedes colocar tus versiones modificadas. En este documento, al primero lo denominaremos `LyXDir` y al segundo UserDir.

2.1. ¿Qué hay en `LyXDir`?

`LyXDir` y sus subdirectorios contienen archivos que pueden emplearse para personalizar el comportamiento de LyX. Puedes cambiar muchos de ellos desde dentro mismo de LyX mediante el diálogo Herramientas ▷ Preferencias. La mayor parte de las adaptaciones personales que querrás hacer en LyX se puede hacer en este diálogo. Sin embargo, muchos otros aspectos internos de LyX pueden cambiarse modificando los archivos en `LyXDir`. Estos archivos pertenecen a diversas categorías, descritas en las siguientes subsecciones.

2.1.1. Archivos generados automáticamente

Los archivos que se encuentran en UserDir se generan al configurar LyX. Contienen varios valores predeterminados que se obtienen por inspección. En general, no es buena idea modificarlos, puesto que podrían ser sobrescritos en cualquier momento.

`lyxrc.defaults` contiene valores predeterminados para varioscomandos.

`packages.lst` contiene la lista de paquetes que han sido reconocidos por LyX. No es utilizada por el propio LyX, pero la información extraida, y otras cosas están disponibles en Ayuda ▷ Configuración de LaTeX.

`textclass.lst` la lista de clases de textos encontradas en los directorios `layout/`, junto con las clases de documentos LaTeX y su descripción.

`lyxmodules.lst` la lista de módulos de formato encontradas en los directorios `layout/`.

`*files.lst` lista de varios tipos de archivos relacionados con LaTeX encontrados en el sistema.

`doc/LaTeXConfig.lyx` es generado automáticamente durante la configuración a partir del archivo `LaTeXConfig.lyx.in`. Contiene información sobre la configuración de LaTeX.

2.1.2. Directorios

Estos directorios están duplicados en `LyXDir` y `UserDir`. Si determinado archivo existe en ambos sitios, se usará el de `UserDir`.

`bind/` este directorio contiene archivos con la extensión `.bind` que definen las combinaciones de teclas usadas en LyX. Si ahí existe una versión nacional de un archivo bind llamado `$LANG_xxx.bind`, ésta se usará en primer lugar.

`citeengines/` contains files with the extension `.citeengine` which define the diverse citation possibilities (natbib, biblatex etc.). See subsection 5.2.6 for details.

`clipart/` contiene archivos gráficos que pueden ser incluidos en documentos.

`doc/` contiene archivos de documentación de LyX (incluido éste que estás leyendo). El archivo `LaTeXConfig.lyx` merece atención especial, como se mencionó antes. Si existe una versión nacional del documento de ayuda en el subdirectorio `doc/xx`, éste se usará en primer lugar. Véase Capítulo 4 para detalles.

`examples/` contiene archivos de ejemplo que describen el uso de algunas características. Se puede acceder a ellos con el botón Ejemplos en el buscador de archivos.

`images/` contiene archivos de imagen que se usan en el diálogo Documento ▷ Configuración. Contiene además los iconos de las barras de herramientas y las enseñas de LyX mostradas en su inicio.

`kbd/` contiene archivos de mapas de teclado. Véase sec. 4.2 para detalles.

`layouts/` contiene las clases de textos y los archivos de los módulos descritos en Capítulo 5.

`lyx2lyx` contiene los guiones de Python `lyx2lyx` para convertir entre versiones de LyX. Pueden ejecutarse en línea de comandos si, digamos, quieres convertir ficheros por lotes.

`scripts/` contiene algunos archivos que demuestran las capacidades de la característica Plantilla externa. También contiene algunos guiones usados por el propio LyX.

`templates/` contiene los archivos de plantillas estándar de LyX descritas en sec. 5.2.4.

`ui/` contiene archivos con la extensión `.ui` que definen la interfaz de usuario de LyX. Esto es, archivos que definen qué ítems aparecen en qué menús y barras de herramientas.

`xtemplates/` contains files with the extension `.xtemplate` which define the templates for the insertion of external material to a LyX document; see **??**.

2.1.3. Archivos que no necesitan modificaciones

Estos archivos son utilizados internamente por LyX y generalmente no deben ser modificados sino por los desarrolladores.

`CREDITS` contiene la lista de desarrolladores de LyX. Se muestra con el menú Ayuda ▷ Acerca de LyX.

`chkconfig.ltx` es un guión de LaTeX usado durante el proceso de configuración. No lo ejecutes directamente.

`configure.py` es el guión utilizado para la reconfiguración de LyX. Genera los archivos de configuración en el directorio desde el que se ha ejecutado.

2.1.4. Otros archivos en un par de líneas

`encodings` contiene tablas que describen cómo los distintos códigos de caracteres se transcriben a Unicode.

`languages` contiene una lista de los idiomas actualmente soportados por LyX.

`latexfonts`contiene información sobre tipografías soportadas.

`layouttranslations` contiene traducciones para entornos de párrafo (véase sec. 5.3.8).

`unicodesymbols` contiene información sobre glifos unicode y su soporte por LyX vía LaTeX.

2.2. Directorio de configuración personal

Aunque uses LyX como usuario sin privilegios, puedes adaptar su configuración a tus necesidades. El directorio `UserDir` contiene todos los archivos de configuración personal. Es el directorio especificado como «Directorio del usuario» en Ayuda ▷ Acerca de LyX. Este directorio se usa como espejo de `LyXDir`, que quiere decir que cada archivo en `UserDir` es un equivalente del correspondiente archivo en `LyXDir`. Cualquiera de los archivos de configuración descritos en las secciones anteriores puede estar en el directorio global del sistema, en cuyo caso afectará a todos los usuarios, y en tu directorio local para tu propio uso.

Para clarificar las cosas vamos a poner unos ejemplos:

- Las preferencias establecidas en el diálogo Herramientas ▷ Preferencias se guardan en un archivo `preferences` en `UserDir`.
- Si reconfiguras mediante Herramientas ▷ Reconfigurar, LyX ejecuta el guión
  `configure.py`, y los archivos resultantes se escriben en tu directorio de usuario. Esto significa que cualquier archivo adicional de clase de texto que pudieras haber añadido en `UserDir/layouts` se sumará a la lista de clases en el diálogo Documento ▷ Configuración.
- Si consigues alguna documentación actualizada del sitio de LyX y no puedes instalarla porque no tienes permisos para administrar tu sistema, solo has de copiar los archivos en `UserDir/doc/XX` ¡y las entradas del menú Ayuda los abrirán!

2.3. Ejecutar LyX con múltiples configuraciones

La libertad de configuración del directorio local puede no ser suficiente si quieres tener a tu disposición más de una configuración. Por ejemplo, si quisieras usar diferentes combinaciones de teclas o configuraciones de impresora en distintos momentos. Esto lo puedes conseguir teniendo varios de estos directorios. Después especificas qué directorio usar al arrancar.

Ejecutando LyX con la opción de línea de comandos `-userdir` *<algún directorio>* hace que el programa lea la configuración de ese directorio y no del predeterminado. (Puedes determinar el directorio predeterminado ejecutando LyX sin la opción `-userdir`). Si el directorio especificado no existe, LyX propone su creación, tal como hace con el directorio predeterminado la primera vez que se ejecuta el programa. Puedes modificar las opciones de configuración en ese directorio personal adicional exactamente igual que para el directorio predeterminado. Estos directorios son completamente independientes (pero sigue leyendo). Ten presente que establecer algún valor para la variable de entorno `LYX_USERDIR_20x` tiene exactamente el mismo efecto.

Tener varias configuraciones también requiere más mantenimiento: si quieres añadir un nuevo formato a `NewUserDir/layouts` que esté disponible en todas tus configuraciones, debes añadirlo a cada directorio por separado. Puedes evitarlo con el siguiente truco: después de que LyX crea un directorio adicional, la mayoría de los subdirectorios (véase arriba) están vacíos. Si quieres que la nueva configuración sea una réplica de una existente, reemplaza el subdirectorio vacío con un enlace simbólico que apunte al subdirectorio deseado. Ten precaución, no obstante, con el subdirectorio `doc/`, puesto que contiene un archivo escrito por el guión de configuración (también accesible mediante Herramientas ▷ Reconfigurar) que es específico de la configuración.

3. El diálogo Preferencias

Todas las opciones del diálogo de preferencias se describen en el apéndice *El diálogo Preferencias* de la *Guía del usuario*. Ahí encontrarás más detalles sobre algunas opciones.

3.1. Formatos

El primer paso es definir tus propios formatos de archivo si no lo están ya. Para hacerlo abre el diálogo HERRAMIENTAS ▷ PREFERENCIAS. En GESTIÓN DE ARCHIVOS ▷ FORMATOS DE ARCHIVO pulsa el botón NUEVO... para definir el nuevo formato. El campo FORMATO es para el nombre que identificará el formato en la GUI (interfaz gráfica de usuario). El NOMBRE CORTO se usa para identificar el formato internamente. Además deberás introducir una extensión de archivo. Todos estos son imprescindibles. El campo opcional ATAJO DE TECLADO sirve para asociar una combinación de teclas en los menús. (Por ejemplo, teclear CTRL+D ejecuta DOCUMENTO ▷ VER ▷ DVI). VISOR

Un Formato puede tener un VISOR y un EDITOR asociados. Por ejemplo, podrías usar GHOSTVIEW para ver archivos PostScript. Puedes introducir el comando para iniciar el programa en el campo correspondiente. Al definir este comando puedes usar las cuatro variables listadas en la siguiente sección. El visor se lanza cuando ves una imagen en LyX o usas el menú DOCUMENTO ▷ VER. El editor se lanza cuando pulsas el botón EDITAR EXTERNAMENTE que se muestra al hacer clic derecho en un gráfico o en un material externo, por ejemplo.

El tipo MIME de un formato es opcional, pero si se especifica, debe ser único para todos los formatos. Se usa para detectar archivos de ese formato de los contenidos de archivo. Para algunos formatos de archivo importantes no hay tipo MIME oficialmente registrado con IANA. Por tanto LyX usa la lista extensa de tipos MIME especificada por freedesktop.org.

La opción FORMATO DE DOCUMENTO indica a LyX que ese formato es adecuado para exportar. Si esto está marcado y existe una ruta de conversión adecuada (véase sec. 3.3), el formato aparecerá en el menú ARCHIVO ▷ EXPORTAR. También aparecerá en el menú DOCUMENTO ▷ VER si se ha especificado un visor para ese formato. Los formatos puros de imagen, como `png`, no deberían usar esta opción. Los formatos que pueden corresponder tanto a gráficos vectoriales como a documentos, como `pdf`, sí deberían usarla.

La opción FORMATO DE GRÁFICO VECTORIAL indica a LyX que el formato puede contener gráficos vectoriales. Esta información se emplea para determinar el formato

objetivo de los gráficos incluidos para la exportación PDFLATEX. Los gráficos incluidos pueden requerir conversión a PDF, PNG, o JPG, puesto que PDFLATEX no puede manejar otros formatos de imagen. Si un archivo incluido no está ya en formato PDF, PNG, o JPG, se convierte a PDF si la opción de formato vectorial está marcada, y si no a PNG.

3.2. Copiadores

Puesto que todas las conversiones de un formato a otro tienen lugar en el archivo temporal de LyX, a veces es necesario modificar un archivo antes de copiarlo en el directorio temporal para que la conversión se pueda realizar.[1] Esto lo hace un Copiador: copia un archivo a (o desde) el directorio temporal y puede modificarlo en el proceso.

Las definiciones de los copiadores pueden usar ocho variables:

`$$s`	El directorio de sistema de LyX (p. e. /USR/SHARE/LYX)
`$$i`	El archivo de entrada
`$$o`	El archivo de salida
`$$b`	Nombre base (sin extensión) en el directorio temporal de LyX
`$$p`	Ruta completa al directorio temporal de LyX
`$$r`	Ruta completa al archivo original LyX que se procesa
`$$f`	Nombre del archivo LyX (sin ruta de directorio)
`$$l`	El 'nombre LaTeX'

El último debería ser el nombre de archivo tal como debería ponerse en un comando LaTeX **\include**. Solo es pertinente cuando se exportan archivos adecuados para esa inclusión.

Los copiadores pueden emplearse para hacer casi cualquier cosa con archivos de salida. Por ejemplo, supongamos que quieres copiar archivos PDF generados a un directorio especial, `/home/you/pdf/`. En ese caso deberías escribir un guión para intérprete de comandos tal como:

```
#!/bin/bash
FROMFILE=$1
TOFILE=`basename $2`
cp $FROMFILE /home/you/pdf/$TOFILE
```

[1] Por ejemplo, el archivo puede hacer referencia a otros archivos —imágenes, por ejemplo— mediante nombres de archivo relativos, y estos pueden resultar inválidos cuando el archivo se copia en el directorio temporal.

Guárdalo en tu directorio local, —`/home/you/.lyx/scripts/pdfcopier.sh`, supongamos— y hazlo ejecutable si ello es necesario en tu sistema. Después, en el diálogo HERRAMIENTAS ▷ PREFERENCIAS, en GESTIÓN DE ARCHIVOS ▷ FORMATOS DE ARCHIVO, selecciona el formato PDF(PDFLATEX) —u otro de los formatos pdf— e introduce `pdfcopier.sh $$i $$o` en el campo COPIADOR.

LyX usa los copiadores en varias de sus propias conversiones. Por ejemplo, si se encuentran los programas apropiados, LyX instalará copiadores para los formatos HTML y HTML (MS WORD). Cuando se exporta a estos formatos, el copiador se encarga de que se copien no solo el archivo HTML principal, sino también los diversos archivos asociados (estilos, imágenes, etc). Todos estos archivos se escriben en un subdirectorio del directorio en el que se encuentre el archivo original de LyX.[2]

3.3. Convertidores

Puedes definir tus propios convertidores para convertir archivos entre distintos formatos. Se hace en el diálogo HERRAMIENTAS ▷ PREFERENCIAS ▷ GESTIÓN DE ARCHIVOS ▷ CONVERTIDORES.

Para definir un nuevo convertidor de un formato a otro, selecciónalos en las listas desplegables DEL FORMATO y AL FORMATO, introduce el comando necesario para la conversión y después pulsa el botón AÑADIR. En la definición de convertidores se pueden usar distintas variables:

`$$s` El directorio de sistema de LyX

`$$i` El archivo de entrada

`$$o` El archivo de salida

`$$b` El nombre del archivo base del archivo de entrada (sin la extensión)

`$$p` La ruta al archivo de entrada

`$$r` La ruta al archivo de entrada original (esto es diferente de $$p cuando se invoca una cadena de convertidores).

`$$e` El nombre iconv para la codificación del documento.

En el campo INDICADOR ADICIONAL puedes introducir los siguientes, separados por comas:

`latex` Este convertidor ejecuta alguna forma de LaTeX. Hará estar disponibles los registros de errores LaTeX de LyX

[2] Este copiador puede adaptarse. El argumento opcional «-e» acepta una lista, separada por comas, con las extensiones que deben copiarse; si se omite se copiarán todos los archivos. El argumeto «-t» determina la extensión añadida al directorio generado. Por omisión es «LyXCONV», así, el HTML generado a partir de /RUTA/A/ARCHIVO.LYX tendrá la forma /RUTA/A/ARCHIVO.HTML.LyXCONV.

`needaux` Necesita el archivo LaTeX .AUX para la conversión

`nice` Necesita un «bonito» archivo del trastero, que en la práctica quiere decir un archivo LaTeX como el que queremos exportar, sin `input@path`.

`xml` La salida es XML

Las tres siguientes no son realmente indicadores porque aceptan un argumento en la forma KEY = VALUE:

`parselog` Si se pone, el error estándar del convertidor se redirecciona a un archivo `infile.out`, y el guión dado como argumento se ejecutará como: `script <infile.out >infile.log`. El argumento puede contener `$$s`.

`resultdir` El nombre del directorio en el que el convertidor descargará los archivos generados. LyX no creará ese directorio y no copia nada en él, aunque copiará este directorio al destino. El argumento puede contener `$$b`, que será reemplazado por el nombre base de los archivos de entrada y de salida, respectivamente, cuando se copia el directorio.
Ten en cuenta que resultdir y usetempdir no tienen sentido juntos. El último será ignorado si se da el primero.

`resultfile` Determina el nombre del archivo de salida y puede contener `$$b`. Solo es sensible con `resultdir` y además es opcional; si no se da, por omisión es 'index'.

Ninguno de estos tres últimos se usan actualmente en ninguno de los convertidores que son instalados por LyX.

No tienes que definir convertidores entre todos los formatos que quieras convertir. Por ejemplo, observarás que no hay convertidor 'LyX a PostScript', sin embargo LyX exporta a PostScript. Lo hace creando primero un archivo LaTeX (no es necesario un convertidor para esto), que luego es convertido a DVI mediante el convertidor 'LaTeX a DVI', y por último convierte el archivo DVI resultante a PostScript. LyX encuentra automáticamente estas 'cadenas' de convertidores y siempre escogerá la cadena más corta posible. Sin embargo, aún puedes definir múltiples métodos de conversión entre formatos. Por ejemplo, la configuración estándar de LyX provee cinco caminos para convertir LaTeX a PDF:

1. directamente, usando PDFLATEX
2. mediante (DVI y) PostScript, usando PS2PDF
3. mediante DVI, usando DVIPDFM
4. directamente, usando XeTeX
5. directamente, usando LuaTeX

Para construir estas cadenas alternativas tienes que definir múltiples 'formatos de archivo' objetivo, como se describe en la sección sec. 3.1. Por ejemplo, en la configuración estándar, se definen los formatos llamados PDF (para PS2PDF), PDF2 (para PDFLATEX), PDF3 (para DVIPDFM), PDF4 (para XeTeX), y PDF5 (para LuaTeX), todos ellos compartiendo la extensión .PDF, y que corresponden a los métodos de conversión antes mencionados.

4. Internacionalización de LyX

LyX soporta el uso de una interfaz traducida. La última vez que lo comprobamos, LyX suministraba textos en treinta idiomas. El idioma elegido se denomina *locale*. (Para lecciones adicionales sobre configuración de idiomas, véase también la documentación de «locale» del sistema operativo. En Linux, la página de manual de `locale(5)` es un buen punto de partida).

Hay que advertir que estas traducciones funcionarán pero tienen algunos fallos. En particular, todos los diálogos se han diseñado con el inglés en mente, lo que significa que algún texto traducido podría ser demasiado largo para ajustarse al espacio asignado. Esto solo es un problema de presentación en pantalla y no causará ningún perjuicio. Además, encontrarás que algunas traducciones no definen atajos de teclado para todo. A veces, simplemente no hay letras libres suficientes para ello. Otras veces, el traductor todavía no lo ha completado. Nuestro equipo de traducción, al que podrías unirte si quieres,[1] intentará, por supuesto, corregir estas deficiencias en futuras versiones de LyX.

4.1. Traducción de LyX

4.1.1. Traducción de la interfaz gráfica de usuario (mensajes de texto)

LyX utiliza la biblioteca GNU `gettext` para manejar la internacionalización de la interfaz. Para que LyX hable tu idioma preferido en todos los menús y diálogos es necesario un archivo `po` para ese idioma. Si está disponible, tendrás que generar a partir de él un archivo `mo` e instalarlo. El procedimiento para hacer todo esto se explica en la documentación de GNU `gettext`. Puedes hacer esto solo para ti, pero también podrías compartir el resultado de tu trabajo con el resto de la comunidad LyX. Envía un mensaje a la lista de desarrolladores de LyX para más información sobre cómo proceder.

En resumen, esto es lo que deberías hacer (xx indica el código del idioma):

- Inspecciona el código fuente de LyX. (Véase información en la web.)
- Copia el archivo `lyx.pot` a la carpeta de los archivos `**.po`. Después lo renombras como `xx.po`. (Si `lyx.pot` no está por ningún sitio, puede rehacerse

[1] Si usas con fluidez un idioma distinto del inglés, ¡unirse a estos equipos es una estupenda forma de corresponder a la comunidad LyX!

con el comando de consola `make lyx.pot` en ese directorio, o puedes usar como muestra un archivo `po` de algún otro idioma).

- Edita `xx.po`.[2] Para algunos nombres de menú y otros artilugios hay además atajos de teclado que deberían traducirse. Dichas teclas se marcan con una '|' delante, y deberían traducirse de acuerdo con las palabras y frases del idioma. Además tendrías que rellenar la información al principio del nuevo archivo `po`, correo electrónico, etc., para que la gente sepa dónde enviarte sugerencias o divertidas diatribas.

Si estás haciendo esto solo para ti, entonces:

- Genera `xx.mo`. Se puede hacer con `msgfmt -o xx.mo < xx.po`.
- Copia el archivo `mo` a tu directorio local, en la carpeta apropiada para los mensajes de las aplicaciones en el idioma xx, con el nombre `lyx.mo` (p. e. `/usr/local/share/locale/xx/LC_MESSAGES/lyx.mo`.

Sin embargo, ya se ha dicho, lo mejor sería poder añadir el nuevo archivo `po` a la distribución de LyX, para que otros puedan usarlo. Esto requiere hacer cambios en LyX, así que envía un correo a la lista de desarrolladores si estás interesado en hacerlo.

4.1.1.1. Mensajes ambiguos

A veces resulta que un mensaje en inglés tiene diversas traducciones en un idioma dado. Un ejemplo es la palabra `To`, que en alemán se puede traducir por Nach o Bis, según el sentido exacto que tenga «to» en inglés. GNU `gettext` no maneja tales traducciones ambiguas. Por lo tanto, debes añadir alguna información contextual al mensaje: en vez de `To` se pone `To[[as in 'From format x to format y']]` y `To[[as in 'From page x to page y']]`. Ahora las dos apariciones de `To` son diferentes para `gettext` y pueden traducirse correctamente por Nach y Bis, respectivamente.

Por supuesto que la información contextual debe eliminarse del mensaje original si no se usa en la traducción. Por eso hay que ponerla entre dobles corchetes al final del mensaje (véase el ejemplo). El mecanismo de traducción de LyX asegura que todo lo que va entre corchetes al final de los mensajes se quita antes de mostrarlos.

4.1.2. Traducción de la documentación

La documentación en línea (menú Ayuda) puede (¡debería!) traducirse. Si hay versiones traducidas de la documentación disponible[3] y `locale` está adecuadamente configurado, LyX las cargará «automágicamente». LyX busca las traducciones en

[2] Es un archivo de texto, se puede editar en cualquier editor de texto. Pero hay programas especializados para editar estos archivos, como `Poedit` (para todas las plataformas) o `KBabel` (para KDE). Además, `Emacs` tiene un 'modo' para editar archivos `po`, veáse https://www.gnu.org/software/gettext/manual/html_node/PO-Mode.html#PO-Mode.

[3] Hasta marzo de 2008, al menos algunos de los documentos han sido traducidos a catorce idiomas, y el Tutorial está disponible en algunos más.

`LyXDir/doc/xx/DocName.lyx`, donde `xx` es el código para el idioma en uso actualmente. Si no hay versión traducida se presentará por omisión la versión inglesa del documento. Ten en cuenta que las versiones traducidas deben tener los mismos nombres de archivo (el `DocName` de antes) que el original. Si te animas a traducir documentación (¡de paso, una excelente manera de leerla atentamente!), hay algunas cosas que deberías tener en cuenta:

- Consulta la web de traducción de la documentación en https://www.lyx.org/Translation. Así podrás comprobar el estado actual de traducción de documentos a tu idioma. También si hay alguien que esté coordinando la traducción a tu idioma. Si no hay nadie comunícanos, por favor, si estás interesado en ello.

Una vez que te has decidido a empezar, he aquí algunos consejos que te pueden ahorrar inconvenientes:

- ¡Únete al equipo de documentación! En AYUDA ▷ INTRODUCCIÓN, que por cierto, es el primer documento que se debería traducir, hay información sobre cómo hacerlo.

- Infórmate sobre las normas tipográficas de tu idioma. La tipografía es un antiguo arte que ha desarrollado durante siglos una gran variedad de convenciones en diversas partes del mundo. Estudia también la terminología profesional de los tipógrafos en tu país. Inventar tu propia terminología sólo provocará confusión. *(¡Cuidado! La tipografía es adictiva!)*

- Haz una copia del documento para trabajar sobre ella. Puedes guardarla como archivo personal de traducción en tu directorio `~/.lyx/doc/xx/`.

- De vez en cuando el documento original (del equipo LyX) será actualizado. Usa el visor de fuentes en https://www.lyx.org/trac/timeline para ver los cambios realizados. De esta manera puedes ver fácilmente qué partes del documento traducido necesitan actualizarse.

Si encuentras un error en el documento original, ¡corríjelo y notifica los cambios al equipo de documentación! (¿No te has olvidado de contactar con el equipo de documentación, verdad?)

4.2. Cosas sobre teclados internacionales

Las dos secciones siguientes describen con detalle la sintaxis de los archivos `.kmap` y `.cdef`. Deberían servir de ayuda para diseñar tus propios mapas de teclado si los suministrados no satisfacen tus necesidades.

4.2.1. El archivo .kmap

Un archivo `.kmap` asocia pulsaciones de teclas con caracteres o secuencias de ellos. Como sugiere el nombre, sirve para trazar un mapa del teclado. En esta sección se describen las palabras clave `kmap`, `kmod`, `kxmod` y `kcomb` del archivo `.kmap`.

`kmap` Asocia un carácter a una cadena

`\kmap` *char string*

Esto asocia *char* a *string*. Anotar que en *string*, la doble comilla (") y la barra invertida (\) deben escaparse con una barra (\) previa.

Un ejemplo de una declaración de KMAP para hacer que el símbolo / sea generado por la tecla & es:

```
\kmap & /
```

`kmod` Especifica un carácter acentuado

`\kmod` *char accent allowed*

Esto hará que el carácter *char* sea un acento *accent* sobre el(los) carácter(es) permitido(s) *allowed*. Este es el mecanismo de tecla muerta[4].

Si pulsas *char* y después una tecla que no está entre las permitidas (*allowed*), en la salida obtendrás el carácter *char* seguido por el otro carácter, el no permitido. Hay que tener en cuenta que RETROCESO cancela una tecla muerta, de modo que si pulsas *char* RETROCESO, el cursor no retrocede una posición sino que anula el efecto que *char* hubiera tenido sobre la siguiente pulsación.

En el ejemplo siguiente, el carácter ' va a generar un acento agudo permitido sobre los caracteres a, e, i, o, u, A, E, I, O, U:

```
\kmod ' acute aeiouAEIOU
```

`kxmod` Especifica una excepción para el carácter acentuado

`\kxmod` *accent char result*

Define una excepción para el *accent* sobre *char*. El *accent* debe haber sido asignado a una pulsación con una declaración previa de `\kmod` y *char* no debe estar en el conjunto *allowed* de *accent*. Si tecleas la secuencia *accent char* da como resultado *result*. Si dicha declaración no existe en el archivo `.kmap` y tecleas *accent char*, obtendrás *accent_key char*, donde *accent_key* es el primer argumento de la declaración `\kmod`.

El comando siguiente generará äi cuando teclees agudo-i ('i):

[4] El término *tecla muerta* se refiere a una tecla que no genera ningún carácter por sí mismo, pero seguida de otra tecla genera el carácter acentuado deseado. Por ejemplo, un caráter con diéresis, ü, se genera de esta manera.

```
\kxmod acute i "\\'{\\i}"
```

`kcomb` Combina dos caracteres acentos

`\kcomb` *accent1 accent2 allowed*

Esto es bastante esotérico. Permite combinar el efecto de *accent1* y *accent2* (¡en ese orden!) sobre los caracteres permitidos *allowed*. Las teclas para *accent1* y *accent2* deben haber sido declaradas con un comando `\kmod` *previamente* en el archivo.

Considera este ejemplo del archivo `greek.kmap`:

```
\kmod ; acute aeioyvhAEIOYVH \kmod : umlaut iyIY \kcomb acute umlaut iyIY
```

Esto permite pulsar ;:I y obtener el efecto de `\'{\"{i}}`. En este caso un retroceso cancela la última tecla muerta, así que si tecleas ;: RETROCESO I obtienes `\'{i}`.

4.2.2. El archivo .cdef

Después de realizados el mapa `.kmap`, un archivo `.cdef` traza el mapa de las cadenas que los símbolos generan para los caracteres en la tipografía actual. La distribución de LyX actualmente incluye al menos los archivos `iso8859-1.cdef` and `iso8859-2.cdef`.

En general el archivo `.cdef` es una secuencia de declaraciones de la forma:

char_index_in_set string

Por ejemplo, para asociar \'{E} al correspondiente carácter en el conjunto iso-8859-1 (233), se usa:

```
233 "\\'{e}"
```

con \ y " escapados en *string*. Anotar que el mismo carácter puede aplicarse a más de una cadena. En el archivo `iso-8859-7.cdef` tienes

```
192 "\\'{\\\"{i}}"
192 "\\\"{\\'{i}}"
```

Si LyX no puede encontrar una declaración para la cadena por una tecla o una secuencia con tecla muerta, intentará si aparece como un carácter acentuado y probará a dibujar un acento sobre el carácter en la pantalla.

4.2.3. Teclas muertas

Hay una segunda manera de añadir soporte para caracteres internacionales mediante las denominadas teclas muertas. Una tecla muerta trabaja en combinación con una letra para generar un carácter acentuado. Aquí explicaremos como crear una sencillísima tecla muerta para ilustrar cómo funciona.

Supongamos que necesitas el carácter circunflejo «^». Podrías asociar la tecla ^ (antes MAYÚSCULAS-6) al comando LyX `accent-circumflex` en tu archivo `lyxrc`. Ahora, cuando pulses la tecla ^ seguida de una letra, obtendrás esa letra con circunflejo sobre ella. Por ejemplo, la secuencia «^E» genera «ê». Si pruebas «`^t`», sin embargo, LyX no imprime nada puesto que T nunca lleva un circunflejo. Pulsar ESPACIO tras una tecla muerta imprime el acento solo. ¡Advierte este último punto! Si asocias una tecla a una tecla muerta necesitarás reasociar el carácter de esa tecla a otra tecla. Asociar la tecla , a una cedilla es una mala idea, puesto que solo obtendrás cedillas en lugar de comas.

Una forma habitual de asociar teclas muertas es usar META-, CTRL-, y MAYÚSCULAS- en combinación con un acento, como «~» o «,» o «^». Otra forma incluye el uso de `xmodmap` y `xkeycaps` para configurar la tecla especial `Mode_Switch`. Esta tecla actúa de alguna forma como MAYÚSCULAS y permite teclas a caracteres acentuados. También puedes transformar teclas en muertas asociándolas a algo como `usldead_cedilla` y después asociando esta tecla simbólica al correspondiente comando LyX.[5] Puedes hacer casi cualquier cosa con la tecla `Mode_Switch`: una de las teclas CTRL-, una tecla de función de reserva, etc. En cuanto a los comandos LyX que generan acentos, consulta la entrada para `accent-acute` en *Reference Manual*. Ahí encontrarás la lista completa.

4.2.4. Guardar la configuración de idioma

Puedes editar tus preferencias para que el entorno de idioma adecuado sea automáticamente configurado al iniciar LyX, en el diálogo HERRAMIENTAS ▷ PREFERENCIAS.

[5] Nota de JOHN WEISS: Esto es exactamente lo que hago en mis archivos `~/.lyx/lyxrc` y `~/.xmodmap`. Tengo mi tecla BLOQ DESPL configurada como `Mode_Shift` y unas cuantas de estas teclas simbólicas «`usldead_*`» asociadas a cosas como BLOQ DESPL-^ y BLOQ DESPL-~. Así es como genero mis caracteres acentuados.

5. Instalación de nuevas clases de documento, formatos y plantillas

En este capítulo describimos los procedimientos para crear e instalar nuevos archivos de plantillas y formatos de LyX, así como repasar la instalación correcta de nuevas clases de documentos LaTeX.

En primer lugar, permite que digamos algo sobre la relación entre LyX y LaTeX. Lo que hay que entender es que, en cierto sentido, LyX no sabe nada de LaTeX. Desde el punto de vista de LyX, LaTeX solo es uno más de los varios «formatos de soporte» en los que es capaz de generar salida. Otros son DocBook, texto sencillo y XHTML. LaTeX es, por supuesto, un formato particularmente importante, pero muy poca de la información que LyX tiene sobre LaTeX está realmente contenida en el propio programa.[1] Esa información, incluso para las clases estándar como `article.cls`, está almacenada en 'archivos de formato (*layout*)'. De igual manera LyX no sabe gran cosa sobre DocBook o XHTML. Esa información está en los archivos de formato.

El archivo de formato para una determinada clase de documento se puede considerar como un manual de instrucciones para traducir las estructuras de LyX —párrafos y sus estilos, ciertos tipos de recuadros, etc.— a las estructuras correspondientes de LaTeX, DocBook o XHTML. Por ejemplo, casi todo lo que LyX sabe sobre `article.cls` está contenido en el archivo `article.layout` y en otros archivos incluidos en este. Por esta razón, quien intente escribir archivos de formato debería empezar por estudiar los archivos existentes. Un buen sitio para empezar es `stdsections.inc`, que está incluido en `article.layout`, `book.layout` y muchos otros archivos de formato para clases de documento. En este archivo se definen las secciones y demás: `stdsections.inc` indica a LyX como traducir los párrafos con estilos como Sección, Subsección, etc., a los comandos y etiquetas correspondientes en LaTeX, DocBook y XHTML. El archivo `article.layout` solo incluye, básicamente, varios de estos archivos `std*.inc`.

Pero definir la correspondencia LyX-LaTeX no es lo único que hacen los archivos de formato. Su otra tarea es definir cómo aparecerán en pantalla las estructuras de LyX. El hecho de que los archivos de formato tengan estas dos tareas es a veces fuente de confusión, porque son totalmente independientes. Indicar a LyX cómo traducir cierto estilo de párrafo a LaTeX no le dice cómo mostrarlo en pantalla; a la inversa, indicar cómo mostrar en pantalla un estilo de párrafo no tiene nada que ver con su traducción a LaTeX (solo permite indicar a LaTeX cómo mostrarlo). Así pues, en general, cuando

[1] Algunos comandos son lo bastante complejos como para ser incluidos en el código de LyX. Pero en general los desarrolladores consideran esto una mala cosa.

defines una nueva construcción en LyX, siempre tienes que hacer dos cosas: 1) indicar la traducción a LaTeX y, 2) indicar la presentación en pantalla.

Gran parte de lo dicho es cierto, también, en relación con otros formatos de salida, aunque XHTML es un poco diferente porque en este caso LyX *es capaz*, en cierta medida, de usar la información de la presentación de un párrafo en pantalla para generar la presentación (en forma de CSS) del párrafo en un navegador. Incluso en este caso, no obstante, la distinción entre lo que hace LyX internamente y la forma en que las cosas se muestran externamente sigue en vigor, y ambas se pueden controlar separadamente. Véase sec. 5.4 para más detalles.

5.1. Instalación de nuevos archivos LaTeX

Algunas instalaciones quizá no incluyan un paquete LaTeX que te gustaría usar en LyX. Por ejemplo, podrías necesitar FoilTeX, un paquete para preparar transparencias o diapositivas para proyectores. Las modernas distribuciones de LaTeX como TeXLive (2008 o posterior) o MiKTeX proveen una interfaz de usuario para eso. Por ejemplo, en MiKTeX inicia su programa «Administrador de paquetes» para obtener una lista de los disponibles. Para instalar uno, haz clic derecho sobre él o usa el botón de la herramienta de instalación.

Si tu distribución LaTeX no proporciona tal «administrador de paquetes», o si el paquete no está disponible en ella, sigue los siguientes pasos para instalarlo manualmente:

1. Consigue el paquete de CTAN o de otro sitio.

2. Si el paquete contiene un archivo con la extensión «`.ins`» (es el caso de FoilTeX) abre una consola, cambia a la carpeta del archivo y ejecuta el comando `latex foiltex.ins`. Así desempaquetamos y disponemos de todos los archivos para instalar. La mayoría de paquetes LaTeX no están empaquetados y podemos saltar este paso.

3. Tienes que decidir si el paquete estará disponible para todos los usuarios o no.

   a) En sistemas *nix (Linux, OSX, etc.), si el paquete va estar disponible para todos los usuarios del sistema, hay que instalarlo en el árbol 'local' de TeX, de lo contrario instálalo en tu directorio TeX de usuario. La ubicación de estos árboles, si no existen ya, depende del sistema. Para encontrarlos mira en el archivo `texmf.cnf`.[2] La ubicación del árbol 'local' de TeX está definida por la variable `TEXMFLOCAL`; generalmente algo como /usr/local/share/texmf/ o /usr/local/texlive/XXXX donde XXXX es el año de la distribución TeXLive instalada. La ubicación del árbol 'user' de TeX está definida por `TEXMFHOME` y generalmente es $HOME/texmf/

[2]Usualmente es el directorio $TEXMF/web2c, aunque se puede ejecutar el comando `kpsewhich texmf.cnf` para localizarlo.

o $HOME/.texliveXXXX.. (Si estas variables no están predefinidas, debes hacerlo.) Probablemente necesites permiso de administrador para crear o modificar el árbol 'local', pero no para el árbol 'user'.
En general, es recomendable instalar en el árbol 'user' porque no será modificado o sobrescrito al actualizar el sistema. Además, se guardará copia de él cuando hagas copia de seguridad de tu directorio 'home' (que debe ser lo habitual).

b) En Windows, para que el nuevo paquete esté disponible a todos los usuarios, cambia a la carpeta donde está instalado LaTeX y en la subcarpeta ~\tex\latex, (En MiKTeX, debería ser ~:\Programs\MiKTeX\tex\latex) crea una nueva carpeta `foiltex` y copia en ella todos los archivos del paquete. Para hacer el paquete disponible solo a un usuario o si no dispones de permisos, haz lo mismo pero en el directorio de LaTeX, p. ej., en MiKTeX 2.8 en WinXP
~:\Documents and Settings\<username>\Application Data\
MiKTeX\2.8\tex\latex
en Vista sería
~:\Users\<username>\AppData\Roaming\2.8\MiKTeX\tex\latex .

4. Ahora hay que decir a LaTeX que hay archivos nuevos. Esto depende de la distribución de LaTeX:

   a) Para TeXLive ejecuta el comando `texhash` en una consola. Si el paquete se instaló para todos los usuarios harán falta permisos de administrador.

   b) Para MiKTeX, si el paquete se instaló para todos los usuarios, inicia el programa "Settings (Admin)" y pulsa el botón "Refresh FNDB". Si no, inicia el programa "Settings" y haz lo mismo.

5. Finalmente, hay que decir a LyX que hay nuevos paquetes disponibles. Usa el menú Herramientas ▷ Reconfigurar y reinicia LyX.

Ya está instalado el paquete. En nuestro ejemplo, la clase de documento `Slides (FoilTeX)` estará ahora disponible en Documento ▷ Configuración ▷ Clases de documento.

Si quieres usar una clase de documento LaTeX que no aparece listada en Documento ▷ Configuración ▷ Clases de documento, debes crear un archivo de formato ('layout') para ella. Este es el tema de la sección siguiente.

5.2. Tipos de archivos de formato

Esta sección describe los diversos tipos de arhivos de LyX que contienen información sobre el formato. Los archivos `.layout` describen estilos de párrafo y de carácter, y determinan cómo los debería presentar LyX y cómo deberían traducirse a LaTeX, DocBook, XHTML o cualquier otro formato de salida que se vaya a usar.

Intentaremos aquí dar una minuciosa descripción del proceso; sin embargo, hay tantos tipos diferentes de documentos soportados por clases de LaTeX que no podemos aspirar a tratar todas las distintas posibilidades o problemas que puedas encontrar. La lista de usuarios de LyX es frecuentada por gente con mucha experiencia en el diseño de formatos deseosa de compartir sus conocimientos.

Cuando te pongas a escribir un nuevo formato, es de gran ayuda echar un vistazo a los formatos de ejemplo suministrados con LyX. Si escribes un 'layout' de LyX para una clase de documento LaTeX o un módulo que podría ser útil también a otros, no dudes en compartir tu labor enviándola a sección 'Layouts' en wiki LyX o incluso a la lista de desarrolladores de LyX para poderlo incluir en la distribución.[3]

5.2.1. Módulos de formato

Hemos hablado hasta ahora de 'archivos de formato'. Pero hay distintos tipos de archivos con información sobre formatos. Estrictamente, los archivos de formato tienen la extensión `.layout` y proporcionan a LyX información sobre las clases de documento. Sin embargo, a partir de LyX 1.6, esta información puede proporcionarse también en *módulos*, que tienen la extensión `.module`. Los módulos son a los paquetes LaTeX lo que los formatos a las clases LaTeX, y algunos módulos —como el módulo Notas finales— dan soporte específico a un paquete determinado —en este caso `endnotes`—. En cierto sentido, los módulos de formato son similares a los 'archivos incluidos'[4] —como `stdsections.inc`—, ya que, como estos, los módulos no son específicos para una clase de documento dada, sino que se pueden usar con muchos formatos diferentes. La diferencia es que el uso de un archivo incluido con `article.cls` requiere editar dicho archivo, mientras que los módulos se seleccionan en el diálogo Documento ▷ Configuración.

La construcción de módulos es la forma más fácil de iniciarse en la edición del formato, puesto que puede ser tan sencillo como añadir un nuevo estilo de párrafo o un recuadro flexible. No obstante, los módulos pueden, en principio, contener cualquier cosa que pueda contener un archivo 'layout'.

Después de crear un módulo nuevo y copiarlo en la carpeta `layouts/`, tendrás que reconfigurar LyX y reiniciar después para que el módulo aparezca en el menú. Sin embargo, los cambios que hagas al módulo se verán inmediatamente si abres Documento ▷ Configuración, seleccionas lo que sea y después pulsas «OK». *Es muy recomendable guardar el trabajo antes de hacer eso.* De hecho, *es muy recomendable no intentar editar los módulos mientras se está trabajando simultáneamente en un documento.* Aunque los desarrolladores, por supuesto, se esfuerzan en mantener la estabilidad de LyX en estas situaciones, errores sintácticos y similares en el archivo del módulo podrían causar extraños comportamientos.

[3] Hay que advertir que LyX se acoge a la licencia GPL, por tanto cualquier material aportado debe tener la misma licencia.

[4] Estos pueden tener cualquier extensión pero, por convenio, se usa la extensión `.inc`.

5.2.1.1. Formato local

Los módulos son a LyX lo que los paquetes a LaTeX. Sin embargo, a veces puede suceder que necesites un recuadro específico o un estilo de carácter solo para un documento, y escribir un módulo que estará disponible también para otros documentos tiene poco sentido. En estos casos, lo que necesitas es el «Formato local» de LyX.

Se encuentra en Documento ▷ Configuración ▷ Formato local. El gran cuadro de texto permite introducir todo lo que podría incluirse en un archivo de formato o en un módulo. De hecho, el formato local de un documento se puede considerar como un módulo que le pertenece solo a él. Por tanto, en particular, hay que introducir una etiqueta `Format`. Cualquier formato es aceptable, pero debería usarse normalmente el actual formato. (En LyX 2.3 el formato actual es 66).

Una vez introducido algo en el panel `Local Layout`, LyX habilitará el botón «Validar» de debajo. Pulsando ese botón LyX determinará si lo introducido es información válida para el formato elegido. LyX informará del resultado pero, desafortunadamente, no indicará qué errores pudiera haber habido. Sin embargo, si LyX se ha arrancado en una terminal se verán ahí los errores. El formato local no se podrá guardar hasta que se haya introducido algo válido.

Las advertencias al final de la sección anterior se aplican también aquí. Es mejor no enredar con formato local mientras se está trabajando, especialmente si no se ha guardado el documento. Dicho esto, usar formato local en un documento de prueba puede ser una forma muy conveniente de probar ideas de formato, o incluso de empezar a desarrollar un módulo.

5.2.2. Formato para archivos .sty

Probablemente te vas a encontrar con dos situaciones a la hora de querer soportar una nueva clase de documento LaTeX, según se trate de archivos de estilo (`.sty`) o de archivos de clases LaTeX 2_ε (`.cls`). Dar soporte a un nuevo estilo es bastante fácil. Dar soporte a una nueva clase es un poco más difícil. Trataremos el primer caso en esta sección y el otro en la siguiente. Lo mismo cabe decir, por supuesto, para el soporte de un nuevo DocBook DTD.

El caso más sencillo es aquel en el que la nueva clase de documento se suministra como archivo de estilo que se usará conjuntamente con una clase ya soportada. Para nuestro ejemplo supondremos que el archivo de estilo se llama MYCLASS.STY y que se usará conjuntamente con REPORT.CLS, que es una clase estándar.

Empieza por copiar el archivo 'layout' existente en tu directorio local:[5]

```
cp report.layout ~/.lyx/layouts/myclass.layout
```

Después edita `myclass.layout` y cambia la línea:

```
\DeclareLaTeXClass{report}
```

[5] Evidentemente, cuál es tu directorio local variará según la plataforma, y además LyX permite especificarlo al inicio usando la opción `-userdir`.

por la línea

```
\DeclareLaTeXClass[report, myclass.sty]{report (myclass)}
```

Después añade:

```
Preamble
    \usepackage{myclass}
EndPreamble
```

cerca del principio del archivo.

Inicia LyX y selecciona Herramientas ▷ Reconfigurar. Después reinicia LyX y haz una prueba con un documento nuevo. En el diálogo Documento ▷ Configuración deberías ver «report (myclass)» en la lista de clases de documento. Es probable que algunos de los comandos de secciones y demás en tu nueva clase funcionen de distinta manera a como lo hacen en la clase estándar —`report` en este ejemplo—, de modo que si quieres, ahora puedes enredar un rato con los ajustes de las distintas secciones. La información del formato de las secciones se encuentra en `stdsections.inc`, pero no necesitas copiar ni cambiar este archivo. Simplemente añade tus cambios a tu archivo 'layout' después de la línea `Input stdclass.inc`, que incluye `stdsections.inc`. Por ejemplo, podrías añadir estas líneas:

```
Style Chapter
    Font
        Family Sans
    EndFont
End
```

para cambiar la tipografía de los encabezados de capítulo a sans-serif. Esto sobrescribirá (o, en este caso, añadirá) la declaración para el estilo Capítulo.

Tu nuevo paquete puede proporcionar además comandos o entornos no presentes en la clase base. En este caso, los añadirás al archivo 'layout'. Véase sección sec. 5.3 para información sobre cómo hacerlo.

Si myclass.sty se puede usar con diferentes clases de documento, e incluso si no es así, lo más sencillo sería escribir un módulo que puedes cargar con la clase base. El módulo más simple posible podría ser:

```
#\DeclareLyXModule{My Package}
#DescriptionBegin
#Support for mypkg.sty.
#DescriptionEnd
Format 66
Preamble
    \usepackage{mypkg}
EndPreamble
```

Un módulo más complejo podría modificar el comportamiento de alguna estructura ya existente o definir algunas nuevas. De nuevo te remitimos a sección sec. 5.3 para los detalles.

5.2.3. Formato para archivos .cls

En este caso hay dos opciones. Una, que el archivo de clase esté él mismo basado en una clase de documento existente. Por ejemplo, muchas clases de tesis están basadas en BOOK.CLS. Para ver si el tuyo lo está, busca una línea parecida a

```
\LoadClass{book}
```

en el archivo. Si está, entonces puedes proceder en gran parte como en la sección anterior, aunque la línea DeclareLaTeXClass será diferente. Si tu nueva clase es tesis y está basada en book, la línea debería poner:[6]

```
\DeclareLaTeXClass[thesis,book]{thesis}
```

Si, por el contrario, la nueva clase no está basada en una ya existente, probablemente tendrás que ir pensando en tu propio formato. Recomendamos firmemente copiar un archivo de formato existente que use una clase de LaTeX parecida y modificarlo, si es posible. O al menos, usa un archivo existente como punto de partida para ver qué items son los que te deben preocupar. Los detalles se discuten a continuación.

5.2.4. Creación de plantillas

Una vez escrito un archivo de formato para una clase nueva de documento, podrías pensar en escribir además una *plantilla* para ella. Una plantilla funciona como una especie de tutorial para tu formato, mostrando cómo podría usarse, aunque sea con contenido irrelevante. Naturalmente, puedes echar un vistazo a las plantillas incluidas en LyX para obtener ideas.

Las plantillas se crean exactamente igual que un documento normal: usando LyX. La única diferencia es que los documentos corrientes tienen todos los ajustes posibles, incluso el esquema de tipografías y el tamaño del papel. Generalmente, un usuario no desea que una plantilla sobrescriba sus ajustes preferidos para dichos parámetros. Por esta razón, el diseñador de una plantilla debería quitar comandos como `\font_roman` o `\papersize` de los archivos de plantillas de LyX. Esto se puede hacer con cualquier editor de texto sencillo, por ejemplo `vi` o `notepad`.

Los archivos editados de plantillas que has creado se ubican en `UserDir/templates/`, copia los que uses del directorio global de plantillas en `LyXDir/templates/` al mismo sitio y redefine la ruta a las plantillas en el diálogo HERRAMIENTAS ▷ PREFERENCIAS ▷ RUTAS.

De paso advertiremos que hay una plantilla, `defaults.lyx`, que tiene un papel especial: esta plantilla se carga cada vez que abres un documento nuevo con ARCHIVO ▷ NUEVO, con el fin de proporcionar ajustes predeterminados por omisión. Para (re)crear esta plantilla desde dentro de LyX, lo que debes hacer es abrir un documento, establecer los ajustes deseados y usar el botón GUARDAR COMO PREDETERMINADOS.

[6] Y será mucho más fácil si guardas el archivo como `thesis.layout`: LyX supone que la clase de documento tiene el mismo nombre que el archivo de formato.

5.2.5. Actualización de antiguos archivos de formato

El formato de los archivos de formato cambia con cada nueva versión de LyX, de modo que los antiguos archivos de formato deben ser convertidos. Si LyX lee un archivo de formato con formato antiguo lanzará la herramienta de conversión `layout2layout.py`, que lo convierte a un archivo temporal con el formato actual. El archivo original se deja sin tocar. Si lo usas a menudo y lo quieres convertir permanentemente para evitar que LyX tenga que hacerlo cada vez, ejecuta el convertidor a mano:

1. renombra el archivo `myclass.layout` a `myclass.old`

2. ejecuta el comando
   `python LyXDir/scripts/layout2layout.py myclass.old myclass.layout`
   donde `LyXDir` es el nombre de tu directorio LyX en el sistema.

La conversión manual no puede manejar archivos incluidos, así que estos tendrán que convertirse separadamente.

5.2.6. Cite engine files

A specific form of layout files are the so-called `*.citeengine` files that are located in the `citeengines/` sub-directory. Their purpose is to define the specifics of LaTeX packages aimed at bibliography generation, such as natbib, jurabib or biblatex, but also the way how normal BibTeX citations (without additional packages) are handled in LyX is defined in such a file.

More specifically, it is defined which packages LyX needs to load, which citation commands are available, how these are to be displayed in LyX (in the workarea, the dialogs, the context menus) as well as in the XHTML and plain text output. Furthermore, the files specify available style variants (author-year, numerical, etc.) and their specifics. The cite engine files are also used to generate the options that are available in Document ▷ Settings ▷ Bibliography ▷ Style engine.

Even though a cite engine file is essentially a normal layout file that could theoretically include any layout information, it usually primarily includes some specific parameters such as `MaxCiteNames`, `CiteFramework`, `CiteEngine` and `CiteFormat` blocks. The syntax of the latter two is described in sec. 5.3.13 y sec. 5.3.14, as well as in the files themselves.

5.3. Estructura del archivo 'layout'

Ha llegado por fin el momento de ponerte manos a la obra y crear o editar tu propio archivo de formato; las siguientes secciones describen aquello a lo que te vas a enfrentar. Nuestro consejo es ir despacito, probando y guardando a menudo. En realidad no es para tanto, pero la multitud de opciones puede llegar a abrumarte si

intentas hacer muchas cosas a la vez. Resulta más sencillo usar formatos existentes de LyX como ejemplo o referencia y modificarlos según tus necesidades.

Todos los rótulos o etiquetas en los archivos de formato son insensibles a las mayúsculas; esto significa que `Style`, `style` y `StYlE` son el mismo comando. Los valores posibles se imprimen entre corchetes detrás del nombre de la característica. Si en una descripción de una clase de texto no se especifica una característica, el valor por omisión se escribe en estilo *énfasis*. Si el argumento tiene un tipo de dato como «string» o «float», el valor por omisión se muestra de esta manera: `float=`*`default`*.

5.3.1. Declaración de la clase de documento y clasificación

Las líneas de un archivo de formato que empiezan con `#` son comentarios. Hay una excepción a esta regla: todos los formatos deben comenzar con líneas como:

```
#% Do not delete the line below; configure depends on this[7]
#  \DeclareLaTeXClass{Article (Standard Class)}
#  \DeclareCategory{Articles}
```

La segunda y tercera líneas se usan en la (re)configuración de LyX. El archivo 'layout' es leido por el guión de LaTeX `chkconfig.ltx` de un modo especial tal que `#` se ignora. La primera línea es solo un comentario, la segunda contiene la declaración obligatoria de la clase de texto y la tercera línea contiene la clasificación opcional de la clase. Si estas líneas aparecen en un archivo llamado `article.layout`, entonces definen una clase de texto de nombre `article` (el nombre del archivo de formato) que usa la clase de documento LaTeX `article.cls` (por omisión se usa el mismo nombre para ambos). La palabra «Article (Standard Class)» se utiliza como descripción de la clase de texto y es la que aparece en el diálogo Documento ▷ Configuración. La categoría («Articles» en el ejemplo) también se usa en el diálogo Documento ▷ Configuración: las clases de texto se agrupan por estas categorías (generalmente son géneros, así son categorías típicas «Articles», «Books», «Reports», «Letters», «Presentations», «Curricula Vitae» etc.). Si no se ha declarado categoría la clase se pondrá en el grupo «Sin categoría».

Supongamos que has escrito tu propia clase de texto, que utiliza la clase de documento `article.cls`, pero en la que has cambiado el aspecto de los encabezados de sección. Si lo pones en un archivo `myarticle.layout`, la cabecera de este archivo debería ser:

```
#% Do not delete the line below; configure depends on this
#  \DeclareLaTeXClass[article]{Article (con Mis Encabezados)}
#  \DeclareCategory{Articles}
```

Esto declara la clase de texto `myarticle`, asociada con la clase de documento LaTeX `article.cls`, y cuya descripción es «Article (con Mis Encabezados)». Si la clase de texto depende de varios paquetes, puedes declararla así:

[7] *N. del T.*: No borrar la línea siguiente; la configuración depende de ella

```
#% Do not delete the line below; configure depends on this
#  \DeclareLaTeXClass[article,foo.sty]{Article (con Mis Encabezados)}
#  \DeclareCategory{Articles}
```

Esto indica que la clase de texto utiliza el paquete foo.sty. Finalmente, también es posible declarar clases para código DocBook. Una declaración típica podría ser:

```
#% Do not delete the line below; configure depends on this
# \DeclareDocBookClass[article]{SGML (DocBook Article)}
#  \DeclareCategory{Articles}
```

Anotar que estas declaraciones pueden llevar también un parámetro opcional que declare el nombre de la clase de documento (pero no una lista).

Así, para ser lo más explícito posible, la forma de la declaración del formato es:

```
#  \DeclareLaTeXClass[class,package.sty]{descripción del formato}
#  \DeclareCategory{category}
```

La clase solo debe especificarse si el nombre de archivo de la clase LaTeX y el nombre de archivo del formato son diferentes; si el nombre del archivo de clase no se especifica, LyX simplemente supondrá que es el mismo que el del archivo de formato.

Cuando la clase de texto se ha modificado según tus necesidades, lo que debes hacer es copiarla a `LyXDir/layouts/` o a `UserDir/layouts`, ejecutar HERRAMIENTAS ▷ RECONFIGURAR, y reiniciar LyX. Ahora la nueva clase de texto debería estar disponible junto con las demás.

Una vez instalado el 'layout', puedes editarlo y ver los cambios sin tener que reconfigurar ni reiniciar LyX. [8] Se puede forzar la recarga del formato en uso mediante la función LyX LAYOUT-RELOAD. No hay atajo predeterminado para esta función —aunque por supuesto, puedes asociarle tú mismo una secuencia de teclas—. Normalmente usarás esta función introduciéndola simplemente en el *mini-buffer*.

Aviso: Esto es mucho más que una «característica avanzada». Es *muy* recomendable que no intentes editar tu formato mientras estés trabajando en un documento importante. Usa un documento de prueba. Los errores sintácticos y similares en tu archivo de formato podrían provocar comportamientos extraños. En particular, tales errores podrían provocar que LyX vea el formato actual como inválido e intente cambiar a algún otro.[9] El equipo LyX procura que LyX se mantenga estable en estos casos, pero es mejor prevenir que curar.[10]

[8] En versiones de LyX anteriores a 1.6 no era así. Como resultado, la edición de archivos de formato podía llevar mucho tiempo, pues constantemente había que reconfigurar y reiniciar.

[9] En realidad, los errores en la sintaxis pueden causar incluso el cierre de LyX, porque ciertos tipos de error pueden hacer que LyX sea incapaz de leer *cualquier* información de formato. Por favor ten cuidado.

[10] De todos modos, haz copias de seguridad regularmente. Y sé bueno con mamá.

5.3.2. Declaración de un módulo

Un módulo debe empezar con una línea como la siguiente:

```
#\DeclareLyXModule[endnotes.sty]{Endnotes}
```

El argumento imprescindible es, entre llaves, el nombre del módulo tal como deberá aparecer en DOCUMENTO ▷ CONFIGURACIÓN ▷ MÓDULOS. El argumento entre corchetes es opcional: declara cualesquiera paquetes LaTeX de los que depende el módulo. También es posible usar la forma FROM->TO como argumento opcional, que declara que el módulo solo puede usarse cuando existe una cadena de conversión entre los formatos '`from`' y '`to`'.

La declaración del módulo debe proseguir con líneas como las siguientes:[11]

```
#DescriptionBegin
#Adds an endnote command, in addition to footnotes.
#You will need to add  \theendnotes in TeX code where you
#want the endnotes to appear.
#DescriptionEnd
#Requires: somemodule | othermodule
#Excludes: badmodule
```

La descripción se usa en DOCUMENTO ▷ CONFIGURACIÓN ▷ MÓDULOS para proporcionar al usuario información sobre las acciones del módulo. La línea `Requires` sirve para identificar otros módulos con los que se debe usar este; la línea `Excludes` identifica los módulos con los que este no se puede usar. Ambas son opcionales, y, como se ve, varios módulos deben separarse con el símbolo tubería: |. Anotar que los módulos requeridos se tratan de forma disyuntiva: debe usarse *al menos uno* de los módulos requeridos. Y no se puede usar *ningún* módulo excluido. Aquí, los módulos se identifican por sus nombres de archivo sin la extensión `.module`. Por tanto, `somemodule` es realmente `somemodule.module`.

5.3.3. The CiteEngine file declaration

A cite engine file must begin with a line like the following:

```
#\DeclareLyXCiteEngineModule[biblatex.sty]{Biblatex}
```

The mandatory argument, in curly brackets, is the name of the module, as it should appear in DOCUMENT ▷ SETTINGS ▷ BIBLIOGRAPHY. The argument in square brackets is optional: It declares any LaTeX packages on which the cite engine depends.

The cite engine declaration should then be followed by lines like the following:[12]

[11] Preferiblemente en inglés si el módulo se va a publicar en LyX. Esta descripción aparecerá en la lista de mensajes para ser traducidos en la siguiente actualización de la interfaz.

[12] Preferably in English if the module should be published with LyX. This description will appear in the list of messages to be translated and will be thus translated with the next interface update.

```
# DescriptionBegin
#   Biblatex supports many author-year and numerical styles.
#   It is mainly aimed at the Humanities. It is highly
#   customizable, fully localized and provides many features
#   that are not possible with BibTeX. The use of 'biber' as
#   bibliography processor is advised.
# DescriptionEnd
```

The description is used in DOCUMENT ▷ SETTINGS ▷ BIBLIOGRAPHY to provide the user with information about the cite engine.

5.3.4. Número de formato

La primera línea no comentada de cualquier archivo 'layout', 'inc' o 'module' *debe* contener el número de formato del archivo:

`Format` [`int`] El número de formato del archivo de formato ('layout').

Esta etiqueta se introdujo en LyX 1.4.0. Los archivos anteriores no tienen un número explícito y se consideran como `Format 1`. El número para la versión actual de LyX es formato 66. Cada versión de LyX es capaz de leer los formatos de versiones anteriores, de la misma forma que puede leer archivos producidos con versiones de LyX anteriores. Sin embargo, no está previsto convertir a formatos anteriores.

5.3.5. Parámetros generales de clases de texto

Estos son los parámetros generales que describen completamente una clase de documento. (Esto *no* significa que deban aparecer en archivos `.layout` en vez de en módulos. Un módulo puede contener cualquier etiqueta de formato).

`AddToHTMLPreamble` Añade información para el bloque `<head>` cuando el documento se exporte a XHTML. Típicamente, debería usarse para exportar información de estilo CSS, aunque se puede usar para cualquier cosa que pueda aparecer en `<head>`. Debe acabar con "`EndPreamble`".

`AddToPreamble` Añade información al preámbulo del documento. Debe terminar con «`EndPreamble`».

`CiteEngine <engine>` Defines the possibilities for displaying citation references. See subsection 5.3.13 for details. Must end with "`End`". Primarily used in cite engine files (see subsection 5.2.6).

`CiteFormat` Define formatos a usar en la presentación de información bibliográfica. Véase la sección sec. 5.3.14 para detalles. Debe terminar con "`End`". Primarily used in cite engine files (see subsection 5.2.6).

CiteFramework [*bibtex*,biblatex] Determines whether Biblatex or BibTEX is used to generate a Bibliography. Primarily used in cite engine files (see subsection 5.2.6).

ClassOptions Describe varias opciones globales soportadas por la clase de documento. Véase la sección sec. 5.3.6 para una descripción. Debe terminar con «End».

Columns [*1*, 2] Establece el número de columnas por omisión. Se puede cambiar en el diálogo DOCUMENTO ▷ CONFIGURACIÓN.

Counter Esta secuencia define un nuevo contador. Véase la sección sec. 5.3.11 para detalles. Debe terminar con «End».

DefaultFont Establece la tipografía por omisión para presentar el documento. Véase la sección sec. 5.3.12 para cómo declarar tipografías. Debe terminar con «EndFont».

DefaultModule [<module>] Especifica un módulo que se incluirá por omisión en esta clase de documento; deberá declararse por su nombre de archivo sin la extensión .module. El usuario puede, aún así, quitar el módulo, pero estará activo al principio. (Esto se aplica solo cuando se crean archivos nuevos, o si esta clase se elige para un documento existente.)

DefaultStyle [<style>] Este es el estilo que será asignado a párrafos nuevos, generalmente STANDARD. Si no se especifica, se escogerá por omisión el primer estilo definido, pero es muy recomendable usar esta instrucción.

ExcludesModule [<module>] Indica que el módulo en cuestión (que debería especificarse por el nombre de archivo sin la extensión .module) no puede utilizarse con esta clase de documento. Esto podría usarse en un archivo de formato específico de una publicación para, digamos, impedir el empleo del módulo theorems-sec, que numera los teoremas y demás por secciones. Esta etiqueta *no* puede utilizarse en un módulo. Los módulos tienen su propia forma de excluir otros módulos (véase sec. 5.2.1).

Float Define un nuevo flotante. Véase la sección sec. 5.3.9 para detalles. Debe terminar con «End».

HTMLPreamble Da información para el bloque <head> cuando esta clase de documento se exporte a XHTML. Advertir que esto sobrescribirá completamente cualquier declaración anterior de HTMLPreamble o AddToHTMLPreamble. (Usa AddToHTMLPreamble si solo quieres añadir material al preámbulo). Debe acabar con "EndPreamble".

HTMLTOCSection [<style>] El estilo a usar para el índice general, bibliografía, etc., cuando el documento se exporte a HTML. Para article, normalmente será Section; para book Chapter. Si no se especifica, LyX intentará deducir qué formato usar.

IfCounter [<counter>] Modifica las propiedades del contador dado. Si el contador no existe, esta sección se ignora. Debe acabar con "End".
Véase la sección sec. 5.3.11 para más detalles.

Input [<filename>] Permite incluir otro archivo de definición de formato en el tuyo con el fin de evitar la duplicación de comandos. Ejemplos habituales son los archivos de formatos estándar, como `stdclass.inc`, que contiene la mayoría de los diseños básicos.

InsetLayout [<type>] Esta sección (re)define el formato de un recuadro. Puede aplicarse a uno existente o a uno nuevo, definido por el usuario, p. e., un nuevo estilo de texto. Véase la sección sec. 5.3.10 para más información. Debe terminar con «End».

LeftMargin [string] Una cadena que indica la anchura del margen izquierdo en la pantalla, por ejemplo «MMMMM». (Esto no es una 'longitud', como «2ex».

MaxCiteNames [integer] An integer that determines the maximal number of names displayed in an author-year citation before the citation switches to "FirstAuthor et al.". Primarily used in cite engine files (see subsection 5.2.6).

ModifyStyle [<style>] Modifica las propiedades del estilo de párrafo dado. Si el estilo no existe, esta sección se ignora. Debe acabar con "End".

NoCounter [<counter>] Este comando elimina un contador existente, generalmente uno definido en un archivo incluido.

NoFloat [<float>] Este comando borra un flotante existente. Es particularmente útil si quieres suprimir un flotante que ha sido definido en un archivo de entrada.

NoStyle [<style>] Este comando borra un estilo existente.

OutputFormat [<format>] El formato de archivo (tal como se define en las preferencias de LyX) producido por esta clase. Principalmente, es útil cuando `OutputType` es `literate` y se quiere definir un nuevo tipo de documento 'literate'. Esta cadena se redefine a «docbook» o «latex» si se encuentra el correspondiente parámetro `OutputType`.

OutputType [*latex*, docbook, literate] Indica qué clase de documentos de salida que usan esta clase se generarán.

PackageOptions [string string] Especifica opciones, dadas en el segundo `string`, para el paquete nombrado en el primer `string`. Por ejemplo, «`PackageOptions natbib square`» hará que `natbib` se cargue con la opción `square`. (Para TeXpertos, esto hace que LyX interprete: `\PassOptionsToPackage{natbib}{square}` antes de cargar `natbib`).

PageStyle [*plain*, empty, headings] El estilo de página predeterminado de la clase. Puede cambiarse en el diálogo DOCUMENTO ▷ CONFIGURACIÓN.

Preamble Establece el preámbulo LaTeX para el documento. Anotar que esto anulará completamente cualesquiera declaraciones Premable o AddToPreamble anteriores. (Usa AddToPreamble si solo quieres añadir material al preámbulo). Debe terminar con «EndPreamble».

Provides [string] [*0*, 1] Si la clase ya proporciona, o no, la característica string. Una característica es en general el nombre de un paquete (amsmath, makeidx, ...) o una macro (url, boldsymbol,...). Véase en el capítulo Apéndice A la lista de características.

ProvidesModule [string] Indica que este formato proporciona la funcionalidad del módulo mencionado, que deberá especificarse con su nombre de archivo sin la extensión .module. Esto se usará típicamente si el formato incluye el módulo directamente, preferiblemente a usar la etiqueta DefaultModule para indicar que debería utilizarse. Podría ser empleado en un módulo que provea una implementación alternativa de la misma funcionalidad.

ProvideStyle [<style>] Crea un nuevo estilo (entorno) de párrafo si no existe. Si el estilo existe se ignora esta sección. Debe acabar con "End".

Requires [string] Si la clase requiere la característica string. Múltiples características deben separarse con comas. Multiple features must be separated by commas. Anotar que solo se pueden demandar características soportadas. (Véase la lista en el capítulo Apéndice A). Si necesitas un paquete con opciones específicas puedes usar además PackageOptions.

RightMargin [string] Una cadena que indica la anchura del margen derecho en la pantalla, por ejemplo, «MMMMM».

SecNumDepth [int=3] Establece qué divisiones se numeran. Corresponde al contador LaTeX secnumdepth.

Sides [*1*, 2] Si debe imprimirse en una o en ambas caras del papel. Se puede cambiar en el diálogo DOCUMENTO ▷ CONFIGURACIÓN.

Style [<name>] Esta secuencia define un nuevo estilo de párrafo. Si el estilo ya existe, en ese caso se redefinirán algunos de los parámetros. Véase la sección sec. 5.3.7 para detalles. Debe terminarcon «End».

TitleLatexName [string="maketitle"] El nombre del comando de entorno a usar con TitleLatexType.

TitleLatexType [*CommandAfter*, Environment] Indica qué clase de diseño se emplea para definir el título de un documento. CommandAfter significa que la macro

con nombre `TitleLatexName` se insertará después del último formato que tiene «`InTitle 1`». `Environment` corresponde al caso en que todos los formatos que tienen «`InTitle 1`» debería ser incluido en el entorno `TitleLatexName`.

`TocDepth` [`int=3`] Establece qué divisiones se incluyen en el índice general. Corresponde al contador LaTeX `tocdepth`.

5.3.6. Sección `ClassOptions`

La sección `ClassOptions` puede contener las siguientes entradas:

`FontSize` [`string="10|11|12"`] La lista de los tamaños de carácter disponibles para la tipografía principal del documento, separados por «`|`».

`Header` Para establecer la línea DTD con las clases con salida basada en XML. P. e.: PUBLIC «-//OASIS//DTD DocBook V4.2//EN».

`Other` [`string=""`] Algunas opciones de clase de documento, separadas por coma, que se añadirán a la parte opcional del comando `\documentclass`.

`PageStyle` [`string="empty|plain|headings|fancy"`] Lista de los estilos de página disponibles, separados por «`|`».

La sección `ClassOptions` debe terminar con «`End`».

5.3.7. Estilos de párrafo

Una descripción de estilo de párrafo tiene una aspecto como éste:[13]

```
Style name
  ...
End
```

donde se permiten los comandos siguientes:

`AddToToc` [`string=""`] This paragraph will appear in the table of contents of the given type. An empty string disables. See also the OutlinerName and the IsTocCaption commands. Default: disabled.

`Align` [*`block`*`, left, right, center`] Alineación del párrafo.

`AlignPossible` [*`block`*`, left, right, center`] Una lista separada con comas de alineaciones permitidas. (Algunos estilos de LaTeX prohiben ciertos alineamientos, ya que no tendrían sentido. Por ejemplo una alineación a la derecha o al centro de una lista numerada no es posible).

[13] Anotar que esto definirá un nuevo diseño o modificará uno existente.

`Argument` [`int`] Define el número de argumento de un comando o entorno asociado al estilo actual. Es útil para cosas como encabezados de sección y solo tiene sentido con LaTeX. Cada argumento (opcional o requerido) de un comando o entorno —excepto para el argumento requerido que se asocia con el contenido del propio párrafo— tiene una definición separada, en la que el número especifica el orden de los argumentos. La definición debe finalizar con `EndArgument`. Así, un comando con dos argumentos opcionales tiene:

```
Argument 1
...
EndArgument
Argument 2
...
EndArgument
```

En la definición de `Argument` son posibles las siguientes especificaciones:

- `LabelString [string]` Secuencia que aparecerá tanto en el menú (a insertar este argumento) como en el botón de inserción del argumento (a menos que especifiques también un `MenuString` separado). Para el menú puedes definir un acelerador agregando el carácter respectivo a la secuencia, separado con «|» (p. e. «`Short Title|S`»).
- `MenuString [string]` Secuencia separada para el menú Puedes definir un acelerador agregando el carácter respectivo a la secuencia, separado con «|» (p. e. «`Short Title|S`»). Esta especificación es opcional. Si no se da se usará `LabelString` para el menú.
- `Tooltip [string]` Texto explicativo que se muestra en un recuadro sugerencia al planear sobre el recuadro del argumento.
- `Mandatory` [*0*, 1] Declara si es argumento obligatorio (1) u opcional (0). Los argumentos obligatorios se emitirán vacíos si no se dan, mientras que los opcionales no serán emitidos. Por omisión, los argumentos obligatorios se delimitan con `{...}` y los opcionales con `[...]`
- `Requires [int=0]` define otro argumento (por su número) que este argumento requiere para emitirse si él mismo es emitido. P. e., en comandos LaTeX, los argumentos opcionales requieren a menudo argumentos opcionales previos para ser emitidos (al menos vacíos), como en `\command[][argument]{text}`. Esto se puede conseguir con la declaración `Requires 1` dentro de `Argument 2`.
- `LeftDelim [string]` define un delimitador izquierdo personalizado (en vez de `{` o `[`). Un salto de línea en la salida se puede indicar con `<br/>`.
- `RightDelim [string]` define un delimitador derecho personalizado (en vez de `}` o `]`). Un salto de línea en la salida se puede indicar con `<br/>`.

- `DefaultArg [string]` define un argumento que se inserta si y solo si no se han dado argumentos específicos del usuario, o sea, si no se ha insertado ningún recuadro de argumento (también un recuadro de argumento vacío omite DefaultArg). Múltiples argumentos deben separarse con comas.
- `PresetArg [string]` define un argumento que se inserta en cualquier caso (solo o añadido a argumentos especificados por el usuario). Múltiples argumentos deben separarse con comas.
- `Font` Tipografía usada para el contenido del argumento, véase sec. 5.3.12.
- `LabelFont` Tipografía usada para la etiqueta, véase sec. 5.3.12.
- `Decoration` [*`Classic`*, `Minimalistic, Conglomerate`] describe el estilo de representación para los botones y el marco del recuadro de inserción.
- `AutoInsert [int=0]` Si se pone `1`, este argumento se inserta automáticamente cuando se selecciona el estilo respectivo. Actualmente solo se puede insertar automaticamente un argumento por estilo/formato.
- `InsertCotext [int=0]` Si se pone `1`, este argumento se insertará con una copia del co-texto (texto seleccionado o párrafo completo) como contenido.
- `PassThru` [*`inherited`*, `true, false`] Whether the contents of this argument should be output in raw form, meaning without special translations that LaTeX would require. By default, the `PassThru` status is inherited by the inset or paragraph layout the argument belongs to, true and false change the status for the given argument only.
- `PassThruChars` [`string of characters`] Define caracteres individuales que deberían emitirse en bruto, es decir, sin traducciones especiales que podría requerir LaTeX. Al contrario que PassThru, esto debe ser explícitamente definido para argumentos. Esto es, los argumentos no heredan PassThruChars de su recuadro de inserción o formato padre.
- `IsTocCaption` [*`0`*, `1`] If this is set to 1, the argument will output its content in the corresponding item in the table of contents. See AddToToc.

Por omisión, el texto introducido en el área de trabajo de LyX en el archivo respectivo de formato es el último argumento (obligatorio) de un comando si `LatexType` es `Command`. Sin embargo, los argumentos con el prefijo `post:` se emiten después de este argumento de área de trabajo. La numeración de este post-argumento reinicia a 1, así que el primer argumento siguiente al del área de trabajo es `post:1`. Los post-argumentos se ignoran en cualquier otro `LatexType` distinto de `Command`.

Los argumentos para `\item`s de lista (como en `\item[foo]`) tienen el prefijo `item:` seguido por el número (p. e. `Argument item:1`)

AutoNests Includes a comma-separated list of layouts that should be nested in and after the current layout. Only makes sense for nestable layouts (such as environments). Must be ended by "EndAutoNests". See also IsAutoNestedBy.

BabelPreamble Esto sobrescribirá completamente cualquier anterior declaración BabelPreamble para este estilo. Debe terminar con "EndBabelPreamble". Véase la sección sec. 5.3.8 para detalles sobre su uso.

BottomSep [float=0][14] El espacio vertical con el que el último de una cadena de párrafos con este estilo se separa del siguiente párrafo. Si el párrafo siguiente tiene otro estilo, las separaciones no son simplemente añadidas sino que se pone la máxima.

Category [string] La categoría para este estilo. Se usa para agrupar estilos relacionados en el recuadro de estilo en la barra de herramientas. Se puede poner cualquier cadena en 'string' pero mejor usar categorías existentes con tus propios estilos.

CommandDepth Profundidad del comando XML. Usado solo con formatos tipo XML.

CopyStyle [string] Copia todas las características de un estilo existente en el actual.

DependsOn [<name>] El nombre de un estilo cuyo preámbulo debería salir *antes* que éste. Esto permite asegurar un orden de los retazos de preámbulo si las definiciones de macros dependen de otra.[15]

EndLabeltype [*No_Label*, Box, Filled_Box, Static] El tipo de etiqueta que se pone al final del párrafo (o secuencia de párrafos si LatexType es Environment, Item_Environment o List_Environment). No_Label quiere decir «nada», Box (resp. Filled_Box) es un cuadrado blanco (resp. negro) adecuado para marcas finales de demostraciones, Static es una cadena de texto explícito.

EndLabelString [string=""] La secuencia usada para una etiqueta con un Static EndLabelType.

16

Font La tipografía usada para el cuerpo del texto *y* para la etiqueta. Véase la sección sec. 5.3.12. Anotar que al definir esta tipografía se define automáticamente la de la etiqueta, LabelFont. Así que debería definirse primero ésta si se quiere definir también LabelFont.

ForceLocal [int=0] Se usa para compatibilizar estilos nuevos a versiones estables de LyX. La primera versión estable que soporta esta etiqueta es LyX 2.1.0. El argumento es un número 0, -1 o cualquier valor mayor de cero. Si el indicador

`ForceLocal` de un estilo es mayor que cero, se escribirá siempre en la cabecera del documento. Si un archivo .lyx es leido, las definiciones de estilo de la cabecera del documento se añaden a la clase de documento. Por eso, incluso versiones anteriores de LyX pueden manejar el estilo. El argumento de `ForceLocal` es un número de versión: si el estilo es leido, y el número de versión es menor que el número de versión del estilo existente en la clase de documento, el nuevo se ignora. Si el número de versión es mayor, el estilo nuevo reemplaza al existente. -1 significa un número de versión infinito, o sea, el estilo se usa siempre.

`FreeSpacing` [*0*, 1] Usualmente LyX no permite insertar más de un espacio entre palabras, puesto que un espacio se considera como la separación entre dos palabras, no un carácter o símbolo por sí mismo. Esto es una buena cosa pero a veces puede ser molesta, por ejemplo, para escribir código de programas o de LaTeX puro. Por esta razón se puede habilitar `FreeSpacing`. Anotar que LyX creará espacios protegidos para los espacios adicionales en modos que no sean LaTeX.

`HTML*` Para la salida XHTML. Véase sec. 5.4.1.

`InnerTag` [[FIXME]] (Usado solo con formatos tipo XML.)

`InPreamble` [0, 1] Si `1`, marca el estilo para ser incluido en el preámbulo del documento en lugar de en el cuerpo del documento. Útil para clases de documento que piden informaciones tales como título y autor en el preámbulo. Esto solo funciona para estilos para los que `LatexType` es `Command` o `Paragraph`.

`InTitle` [0, 1] Si es `1`, marca el estilo como parte de un bloque de título (véanse también las entradas globales `TitleLatexType` y `TitleLatexName`).

`IsAutoNestedBy` Includes a comma-separated list of layouts after which this one should be nested. Only makes sense with regard to nestable layouts (such as environments). Must be ended by "`EndIsAutoNestedBy`". See also `AutoNests`.

`IsTocCaption` [*0*, 1] If this is set to 1 and AddToToc is enabled, the paragraph adds a summary of its contents in its item in the table of contents. Otherwise, only the label, if it exists, appears.

`ItemCommand` [`string="item"`] La secuencia de comando LaTeX que declara un ítem en una lista. El comando es para ser definido sin la barra invertida (por omisión es "item", que resulta en `\item` en la salida LaTeX).

`ItemSep` [`float=0`] Esto proporciona espacio extra entre párrafos que tienen el mismo formato. Si se ponen otros formatos en un entorno, cada uno es separado con el comando de entorno `ParSep`. Pero el conjunto de los ítems del entorno son separados adicionalmente con este `ItemSep`. Anotar que esto es un *multiplicador.*

ItemTag [[FIXME]] (Usado solo con formatos tipo XML.)

KeepEmpty [0, 1] Habitualmente, LyX no permite dejar un párrafo vacío, porque llevaría a una salida de LaTeX vacía. Sin embargo, hay algunos casos donde podría ser útil: en una plantilla de carta, los campos requeridos pueden proporcionarse vacíos, así el usuario no los olvida; en algunos casos especiales, se puede usar un estilo que no contiene texto real como una especie de salto.

LabelBottomsep [float=0] El espacio vertical entre la etiqueta y el cuerpo del texto. Solo se usa para etiquetas que están sobre el cuerpo del texto (Top_Environment y Centered_Top_Environment).

LabelCounter [string=""] El nombre del contador para numeración automática. Con el fin de que el contador aparezca con su etiqueta, deberás referenciarla en LabelString. Esto funcionará con LabelTypes, Static, Above y Centered, al menos.
Esto *puede* ponerse también si LabelType es Enumerate, aunque este caso es un poco complicado. Supongamos que declaras "LabelCounter myenum". Entonces, los contadores reales usados son myenumi, myenumii, myenumiii, y myenumiv, como en LaTeX. Estos contadores deben declararse todos por separado.
Véase la sección sec. 5.3.11 para detalles sobre contadores

LabelFont La tipografía usada para la etiqueta. Véase la sección sec. 5.3.12.

LabelIndent [string=""] Texto que indica cuánto se debe sangrar una etiqueta.

LabelSep [string=""] Texto que indica el valor de espacio horizontal entre la etiqueta y el cuerpo del texto. Solo se usa para etiquetas que no están sobre el cuerpo del texto.

LabelString [string=""] La cadena usada para una etiqueta del LabelType Static. Si LabelCounter está establecido, esta cadena puede contener los comandos especiales de formato descritos en la sección sec. 5.3.11.

LabelStringAppendix [string=""] Esta se usa en el apéndice en vez de LabelString. Anotar que toda declaración de LabelString reconfigura también LabelStringAppendix.

LabelTag [FIXME] (Usado solo con formatos tipo XML.)

LabelType [*No_Label*, Manual, Static, Above, Centered, Sensitive, Enumerate, Itemize, Bibliography]

Manual significa que la etiqueta es la primera palabra (hasta elprimer espacio real). Usa espacios protegidos si quieres más de una palabra como etiqueta.

Static significa que la etiqueta es simplemente cualquier `LabelString` declarado como tal. Esto se mostrará «en línea» al comienzo del párrafo. Si `LatexType` es `Environment`, entonces se mostrará solo en el primer párrafo de cualquier secuencia de párrafos con el mismo `Style`.

Above y Centered son casos especiales de `Static`. La etiqueta se imprimirá encima del párrafo, o al inicio de línea o centrado.

Sensitive es un caso especial para las etiquetas de las leyendas de «Figura» y «Tabla». `Sensitive` quiere decir que la cadena (en código) de la etiqueta depende de la clase de flotante: Está codificado para ser 'FloatType N', donde N es el valor del contador asociado al flotante. En el caso en que la leyenda se inserte adosada a un flotante `LabelString` aparecerá como «Senseless!» , (¡Sin sentido!).

Enumerate produce los tipos habituales de etiquetas de enumeración. The number type needs to be set in the Contador, véase sec. 5.3.11.

Itemize produce varias marcas para los distintos niveles. The bullet types displayed can be set via DOCUMENTO ▷ CONFIGURATCIÓN ▷ MARCAS.

Bibliography solo debería emplearse con `LatexType BibEnvironment`.

LangPreamble Esto sobrescribirá completamente cualquier declaración anterior de `LangPreamble` para este estilo. Debe acabar con "`EndLangPreamble`". Véase la sección sec. 5.3.8 para detallessobre su uso.

LatexName [`<name>`] El nombre correspondiente en LaTeX, ya sea de un comando o de un entorno.

LatexParam [`<parameter>`] Un parámetro opcional para el correspondiente `LatexName`. Este parámetro no se puede cambiar desde dentro de LyX (usa `Argument` para parámetros personalizables). Se emitirá como tal tras todos los `Argument` LaTeX.

LatexType [*Paragraph*, `Command, Environment, Item_Environment, List_Environment, Bib_Environment`] Cómo debería traducirse el estilo a LaTeX.[17]

Paragraph no significa nada especial.

Command significa `\`*LatexName*`{...}`.

Environment significa `\begin{`*LatexName*`}...\end{`*LatexName*`}`.

Item_Environment es lo mismo que `Environment`, excepto que un `\item` se genera para cada párrafo de este entorno.

List_Environment es lo mismo que `Item_Environment`, excepto que `LabelWidthString` se pasa como un argumento al entorno. `LabelWidthString` puede definirse en el diálogo EDITAR ▷ CONFIGURACIÓN DE PÁRRAFOS.

Bib_Environment es como Environment pero añade además el argumento obligatorio necesario (la etiqueta más larga) a la declaración begin del entorno bibliografía:
\begin{thebibliography}{99} Por tanto solo es útil para entornos de bibliografía. La etiqueta más larga predeterminada «99» puede cambiarla el usuario en la configuración de párrafo de un ítem bibliográfico.

Poniendo juntas las últimas cosas, la salida LaTeX será una de estas:

```
\LatexName[LatexParam]{...}
```

o:

```
\begin{LatexName}[LatexParam] ... \end{LatexName}.
```

dependiendo del tipo de LaTeX.

LeftDelim [string] Secuencia que se pone al inicio del contenido del estilo. Un salto de línea en la salida puede indicarse con
.

LeftMargin [string=""] Si pones estilos en entornos, los LeftMargin no son simplemente añadidos, sino aumentados en un factor $\frac{4}{\text{depth}+4}$. Anotar que este parámetro se usa también cuando Margin se define como Manual o Dynamic. En ese caso se añade al margen manual o dinámico.
Por ejemplo «MM» significa que el párrafo se sangra con la anchura de «MM» en la tipografía normal. Se puede obtener una anchura negativa anteponiendo «-» a la cadena. Se eligió este método para que el aspecto sea el mismo con cada una de las tipografías de pantalla usadas.

Margin [*Static*, Manual, Dynamic, First_Dynamic, Right_Address_Box]
El tipo de margen que el estilo tiene en el lado izquierdo.

Static significa un margen fijo.

Manual significa que el margen izquierdo depende de la cadena introducida en el diálogo Editar ▷ Configuración del párrafo.... Esto se utiliza para componer bonitas listas sin tabuladores.

Dynamic significa que el margen depende del tamaño de la etiqueta. Esto se usa para las cabeceras automáticas de las listas numeradas. Es obvio que la cabecera «5.4.3.2.1 Cabecera muy larga» debe tener un margen izquierdo más amplio (tan ancho como «5.4.3.2.1» más el espacio) que «3.2 Cabecera muy larga», aunque los «procesadores de texto» corrientes no sean capaces de hacer esto.

First_Dynamic es similar, pero solo la primera fila del párrafo es dinámica, mientras que las demás son estáticas; esto se emplea, por ejemplo, para descripciones.

`Right_Address_Box` significa que el margen se elige de manera que la fila más larga de este párrafo se adapte al margen derecho. Esto se emplea para componer una dirección en el lado derecho de la página.

`NeedProtect` [*0*, 1] Si los comandos frágiles en este estilo deberían ser protegidos, `\protect`. (Nota: es *no* si este comando debería él mismo ser protegido.)

`Newline` [0, *1*] Si las líneas nuevas se transforman en líneas nuevas LaTeX (\\) o no. La transformación puede desactivarse para permitir una edición más confortable de LaTeX en LyX.

`NextNoIndent` [*0*, 1] Si se permite sangrar la primera fila del párrafo siguiente. `1` significa que no se permite hacerlo; `0` significa que puede hacerse si se desea.

`ObsoletedBy` [`<name>`] Nombre de un estilo que ha reemplazado este estilo. Esto se emplea para renombrar un estilo, mientras que se mantiene compatibilidad hacia atrás.

`ParagraphGroup` [0, 1] Determina si párrafos consecutivos del mismo tipo se tratan como si fueran juntos. Esto tiene el efecto de que `GuiLabel` solo se imprime una vez delante de tal grupo. Por omisión, esto es así para `LaTeXType Environment` y `Bib_Environment` y no lo es para los demás tipos.

`ParbreakIsNewline` [*0*, 1] Indica que los párrafos no serán separados por una línea vacía en la salida LaTeX, solo por un salto de línea; junto con `PassThru 1`, esto permite emular un editor de texto simple (como un recuadro ERT).

`ParIndent` [`string=""`] El sangrado de la primera línea de un párrafo. `Parindent` estará fijado para un determinado estilo. La excepción es el estilo `Normal`, puesto que la sangría de un párrafo de estilo `Normal` puede prohibirse con `NextNoIndent`. Además, los párrafos de estilo `Normal` dentro de entornos usan la sangría `Parindent` del entorno, no su propio valor. Por ejemplo, los párrafos `Normal` en una enumeración no se sangran.

`ParSep` [`float=0`] El espacio vertical entre dos párrafos de este estilo.

`Parskip` [`float=0`] LyX permite al usuario elegir entre «sangrado» o «espacio vertical» para separar párrafos. Si se elige «sangrado», este valor es completamente ignorado. Si se elige «espacio vertical», el sangrado de un estilo tipo «Párrafo» de LaTeX se ignora y todos los párrafos son separados con este argumento «parskip». El espacio vertical se calcula con `Parskip * DefaultHeight`, donde `DefaultHeight` es la altura de una fila con la tipografía normal. De esta forma, el aspecto queda igual con diferentes tipografías de pantalla.

`PassThru` [0, 1] Si el contenido de este párrafo debería aparecer en la salida en forma cruda, es decir sin las traducciones especiales que LaTeX necesitaría.

PassThruChars [string] Define caracteres individuales que deberían emitirse en bruto, o sea, sin traducciones especiales que LaTeX podría requerir.

Preamble Información a incluir en el preámbulo LaTeX cuando se usa este estilo. Utilizado para definir macros, paquetes a cargar, etc., requeridos por este estilo en particular. Debe terminar con «EndPreamble».

RefPrefix [string] El prefijo a usar cuando se crean etiquetas que se refieren a párrafos de este tipo. Esto permite el uso de referencias con formato.

Requires [string] Si el estilo requiere la característica string (véase la lista en Apéndice A). Si necesitas un paquete con opciones específicas puedes usar además PackageOptions como un parámetro de clase de texto general (véase sec. 5.3.5).

ResetArgs [*0*,1] Reinicia los argumentos de este estilo (como definido mediante la etiqueta Argument). Es útil si has copiado un estilo mediante CopyStyle, pero no quieres heredar sus argumentos (requerido y opcional).

ResumeCounter [*0*,1] Resumes a counter that is usually reset at each new sequence of layouts. This is currently only useful when LabelType is Enumerate.

RightDelim [string] Secuencia que se pone al final del contenido del formato. Un salto de línea en la salida se puede indicar con
.

RightMargin [string=""] Similar a LeftMargin.

Spacing [*single*, onehalf, double, other <valor>] Esto define cuál debería ser el espacio predefinido en el estilo. Los argumentos single, onehalf y double corresponden respectivamente a un multiplicador de 1, 1.25 y 1.667. Si se especifica el argumento other, entonces se debería proporcionar además un argumento numérico que será el valor real del multiplicador. Anotar que, al contrario que otros parámetros, Spacing implica la generación de código LaTeX específico, usando el paquete LaTeX setspace.

Spellcheck [0, *1*] Corrección ortográfica de párrafos de este estilo. Por omisión, sí.

StepMasterCounter [*0*,1] Steps the master counter of a given counter at the beginning of a new sequence of layouts. This is currently only useful when LabelType is Enumerate.

TextFont La tipografía para el cuerpo del texto. Véase la sección sec. 5.3.12.

TocLevel [int=3] El nivel del estilo en el índice general. Esto se usa para la numeración automática de los encabezados de sección.

ToggleIndent [*default*, always, never] Esta etiqueta determina si la sangría de primera línea de este párrafo puede conmutarse en el diálogo de configuración del párrafo. Si se establece *default*, la sangría puede conmutarse si la configuración del documento usa estilo de párrafo «sangrado»; con always, la sangría se puede conmutar siempre, no importa la configuración del documento; con never, la sangría nunca se puede conmutar.

TopSep [float=0] El espacio vertical con el que el primero de una cadena de párrafos con este estilo se separa del párrafo anterior. Si el párrafo anterior tiene otro estilo, la separación no es simplemente añadida, sino que se pone la máxima.

5.3.8. Internacionalización de estilos de párrafo

LyX ha soportado desde hace tiempo la internacionalización de la información de formato, pero, hasta la versión 2.0, esto se aplicaba solo a la interfaz de usuario y no a la salida, digamos, PDF. Así, por ej., autores franceses tenían que acudir a feos trucos para conseguir 'Théorème 1' en vez de 'Theorem 1'. Gracias a Georg Baum, este ya no es el caso.

Si un Style define texto que se mostrará en el documento exportado, puede usar LangPreamble y BabelPreamble para soportar correctamente documentos con idioma distinto del inglés e incluso documentos plurilingües. El siguiente extracto (del archivo theorems-ams.inc) muestra cómo funciona:

```
Preamble
    \theoremstyle{remark}
    \newtheorem{claim}[thm]{\protect\claimname}
EndPreamble
LangPreamble
    \providecommand{\claimname}{_(Claim)}
EndLangPreamble
BabelPreamble
    \addto\captions$$lang{\renewcommand{\claimname}{_(Claim)}}
EndBabelPreamble
```

En principio, cualquier LaTeX legal puede aparecer en las etiquetas LangPreamble y BabelPreamble, pero en la práctica el aspecto será generalmente como el mostrado

[14] Anotar que aquí un 'float' es un número real, como: 1.5.

[15] Anotar que, excepto esta funcionalidad, no hay forma de asegurar ningún orden de los preámbulos. El orden que se ve en una versión dada de LyX puede cambiar sin previo aviso en versiones posteriores.

[16] *Nota de Jean-Marc:* No estoy seguro de que estas configuraciones (Fill_Bottom, Fill_Top) tengan mucho uso, y probablemente se quitarán en próximas versiones.

[17] LatexType es un poco engañoso porque estas reglas se aplican también a clases SGML. Consulta los archivos de clases SGML (archivos de nombre db_*.inc) para ejemplos concretos.

aquí. La clave para una correcta traducción del texto impreso es la definición del comando LaTeX `\claimname` y su uso en `\newtheorem`.

La etiqueta `LangPreamble` proporciona internacionalización basada en el idioma global del documento. El contenido de la etiqueta se incluirá en el preámbulo, igual que con la etiqueta `Preamble`. Lo que la hace especial es el uso de la "función" `_()`, que será reemplazada, cuando LyX genere la salida LaTeX, por la traducción de su argumento al idioma del documento.

La etiqueta `BabelPreamble` es más compleja, puesto que está pensada para dar soporte a documentos plurilingües y ofrece una interfaz al paquete `babel`. Su contenido se añadirá al preámbulo una vez por cada idioma que aparezca en el documento. En este caso, el argumento en `_()` se reemplazará por su traducción al idioma en cuestión; la expresión `$$lang` es reemplazada por el nombre del idioma (el usado por el paquete babel).

Un documento en alemán que también incluya una sección en francés tendría lo siguiente en el preámbulo:

```
\addto\captionsfrench{\renewcommand{\claimname}{Affirmation}}
\addto\captionsngerman{\renewcommand{\claimname}{Behauptung}}
\providecommand{\claimname}{Behauptung}
```

LaTeX y `babel` conspirarán para producir el texto correcto en la salida.

Un punto importante a tener en cuenta aquí es que las traducciones son proporcionadas por el propio LyX, por medio del archivo `layouttranslations`. Esto quiere decir, en definitiva, que `LangPreamble` y `BabelPreamble` realmente solo son útiles en archivos de formato que son proporcionados por LyX, puesto que el texto introducido en archivos de formato creados por el usuario no serán tenidos en cuenta por las rutinas de internacionalización de LyX a menos que el archivo `layouttranslations` se modifique como correponde. Sin embargo, cualquier formato creado con la intención de ser incluido en LyX debería usar estas etiquetas en los lugares apropiados. Ten en cuenta que las traducciones de estilo de párrafo provistas por LyX nunca cambian en actualizaciones menores (p. e. de versión 2.1.x a 2.1.y). Sin embargo es muy probable que en actualizaciones mayores (p. e. de 2.0.x a 2.1.y) se introduzcan nuevas traducciones o correcciones.

5.3.9. Flotantes

Es necesario definir los flotantes (FIGURA, CUADRO, ...) en la propia clase de texto. Los flotantes normales se incluyen en el archivo `stdfloats.inc`, así que no tendrás más que añadir

```
Input stdfloats.inc
```

en tu archivo de formato. Si quieres implementar una clase de texto que proponga algún otro tipo de flotante (como la clase AGU empaquetada con LyX), la información siguiente te servirá de ayuda:

AllowedPlacement [string=!htbpH] Opciones de colocación permitidas para este tipo de flotante. El valor es una secuencia de caracteres: *h* ("here if possible"), *t* ("top of page"), *b* ("bottom of page"), *p* ("page of floats"), *H* ("here definitely") y *!* ("ignore LaTeX rules"). El orden no importa. Si las opciones de colocación no están permitidas usa string *none*.

AllowsSideways [0, *1*] Define si el flotante admite rotación mediante el paquete LaTeX `rotfloat` (sidewaysfloat). Pon `0` si el flotante no soporta esta característica.

AllowsWide [0, *1*] Define si el flotante tiene una variante asterisco que abarca columnas en un párrafo de dos columnas. Pon `0` si el flotante no soporta esta característica.

Extension [string=""] La extensión del nombre de un archivo auxiliar para la lista de figuras (u otra cosa). LaTeX escribe las leyendas en este archivo.

GuiName [string=""] La cadena que se usará en los menús y también para la leyenda. Esto se traduce al idioma actual si se usa.

HTML* Controlan la salida XHTML. Véase la sección sec. 5.4.

IsPredefined [*0*, 1] Indica si el flotante está ya definido en la clase de documento o si necesitamos cargar el paquete LaTeX `float` para definirlo sobre la marcha. Por omisión `0`, que significa: usar `float`. Deberá ponerse `1` si el flotante ya está definido por la clase de documento LaTeX.

ListCommand [string=""] El comando usado para generar una lista de flotantes de este tipo; debe omitirse el precedente '\'. Esto *debe* ponerse si `UsesFloatPkg` es 'false', puesto que no hay una forma estándar de generar este comando. Se ignora si `UsesFloatPkg` es 'true', porque en este caso sí hay una forma estándar de definir el comando.

ListName [string=""] Un título para una lista de flotantes de este tipo (índice de figuras, tablas o lo que sea). Se usa para el nombre del recuadro en LyX; se pasa a LaTeX para usarlo como título allí; y se usa como título en la salida. Será traducido al idioma del documento.

NumberWithin [string=""] Este argumento (opcional) determina si los flotantes de esta clase serán numerados dentro de alguna unidad de sección del documento. Por ejemplo, si `NumberWithin` es «`chapter`», los flotantes serán numerados dentro de los capítulos.

Placement [string=""] La colocación por omisión para la clase dada de flotantes. El valor de `string` deberá ser un estándar LaTeX: `t`, `b`, `p` y `h`, por «top» (arriba), «bottom» (abajo), «page» (página), y «here» (aquí), respectivamente.[18] Por encima de éstos hay un nuevo tipo, `H`, que no corresponde realmente a un

flotante, ya que se refiere a: ponerlo aquí, «here», y en ninguna otra parte. Anotar que, sin embargo, el colocador `H` es especial y, debido a detalles de implementación, no puede usarse en tipos de flotantes no empotrados. Si no comprendes lo que esto significa, usa solo «`tbp`».

`RefPrefix` [`string`] El prefijo a usar cuando se crean etiquetas referidas a flotantes de este tipo. Esto permite el uso de referencias con formato. Puedes quitar cualquier `RefPrefix` puesto por un estilo copiado usando el valor especial "OFF", todas mayúsculas.

`Style` [`string`=""] El estilo usado cuando se define el flotante con `\newfloat`.

`Type` [`string`=""] El «tipo» de la nueva clase de flotante, como programa o algoritmo. Después del apropiado `\newfloat`, comandos tales como `\begin{program}` o `\end{algorithm*}` estarán disponibles.

`UsesFloatPkg` [`0`, *`1`*] Nos dice si este flotante se define usando el paquete LaTeX `float`, bien por el archivo de clase o por un paquete, o bien sobre la marcha por el propio LyX.

Anotar que al definir un flotante con tipo *`type`* automáticamente se define el correspondiente contador con nombre *`type`*.

5.3.10. Recuadros flexibles y formato del recuadro

Los recuadros flexibles son de tres tipos:

- estilo del texto (`CharStyle`): estos definen diseños semánticos correspondientes a comandos LaTeX como `\noun` y `\code`.
- definido por el usuario (`Custom`): estos se pueden usar para definir recuadros plegables personalizados, del tipo de los de código TeX, notas al pie y similares. Un ejemplo obvio es un recuadro de nota final, definido en el módulo del mismo nombre (`endnotes`).
- elementos XML (`Element`): para usar con las clases DocBook.

Los recuadros flexibles se definen usando la etiqueta `InsetLayout`, que se explicará en un momento.

La etiqueta `InsetLayout` también sirve para otra función: puede utilizarse para personalizar el diseño general de muchos tipos diferentes de recuadros. Actualmente, `InsetLayout` permite personalizar los parámetros de diseño de notas al pie, notas al margen, notas, recuadros de código TeX (ERT), ramas, listados de programa, índices, cuadros, tablas, algoritmos, URL, y leyendas, así como para definir recuadros flexibles.

La definición de `InsetLayout` debe comenzar con una línea de la forma:

[18] Anotar que el orden de estas letras en la cadena es irrelevante, como en LaTeX.

```
InsetLayout <type>
```

Aquí, `<type>` indica el recuadro cuyo formato se va a definir, y hay cuatro casos.

1. Se va a modificar el formato para un tipo de recuadro preexistente. En este caso, `<type>` puede ser uno cualquiera de los siguientes: `Algorithm`, `Branch`, `Box`, `Box:shaded`, `Caption:Standard`, `ERT`, `Figure`, `Foot`, `Index`, `Info`, `Info:menu`, `Info:shortcut`, `Info:shortcuts`, `Listings`, `Marginal`, `Note:Comment`, `Note:Note`, `Note:GreyedOut`, `Table`, o `URL`.

2. Se va a definir el formato para un recuadro flexible. En este caso, `<type>` debe de la forma «`Flex:<name>`», donde `name` ser cualquier identificador válido no utilizado por un recuadro preexistente. El identificador puede incluir espacios, pero en este caso debe ponerse todo entre comillas. Ten en cuenta que la definición de un flexible debe incluir *además* una entrada `LyXType` declarando qué tipo de recuadro define.

3. Se va a definir el formato para rama específica de usuario. En este caso, `<type>` debe ser de la forma «`Branch:<name>`», donde `name` puede ser cualquier identificador válido de rama definido en el documento del usuario. Este identificador puede incluir espacios, pero, en ese caso todo debe ir entre comillas. El propósito principal de esta característica es permitir a LaTeX envolver ramas específicas como necesite el usuario.

4. Se va a definir el formato de una leyenda específica de usuario (o clase). En este caso, `<type>` debe ser de la forma «`Caption:<name>`», donde `name` especifica el nombre de la leyenda como aparece en el menú. Repasa la leyenda estándar (`Caption:Standard`), las leyendas específicas de las clases KOMA-Script (`Caption:Above`, `Caption:Below`) o el módulo Multilingual Captions (`Caption:Bicaption`) para aplicaciones.

La definición de `InsetLayout` puede contener las siguientes entradas:

`AddToToc` [`string=""`] This inset will appear in the table of contents of the given type. An empty string disables. See also the OutlinerName and the IsTocCaption commands. This is only implemented for Flex insets. Default: disabled.

`Argument` [`int`] Define el número de argumento de un comando/entorno asociado con el formato actual. La definición debe terminar con `EndArgument`. Véase sec. 5.3.7 para detalles.

`BabelPreamble` Preámbulo para comandos de cambio de idioma; véase sec. 5.3.8.

`BgColor` [`<name>`] El color para el fondo del recuadro. Véase la lista de colores en Apéndice B.

ContentAsLabel [*0*, 1] Si usar no el contenido del recuadro como etiqueta del mismo cuando el recuadro se cierra. Por omisión, falso.

CopyStyle [<type>] Como los estilos de párrafo, véase sec. 5.3.7. Ten en cuenta que debes especificar el tipo completo, p. e. `CopyStyle Flex:<name>`.

CustomPars [*0*, 1] Indica si el usuario puede emplear el diálogo Configuración del párrafo para modificar el párrafo.

Decoration Puede ser `Classic`, `Minimalistic`, o `Conglomerate`, que describe el estilo de presentación para el marco y los botones del recuadro. Las notas al pie usan generalmente `Classic`, los recuadros de código TEX `Minimalistic`, y los de estilos del texto `Conglomerate`.

Display [0, *1*] Solo es útil si `LatexType` es `Environment`. Indica si el entorno se ubicará en su propio espacio en la salida o si aparecerá en línea con el texto circundante. Si se pone falso, se supone que el entorno LATEX ignora los espacios en blanco (incluyendo un carácter de línea nueva) después de las etiquetas `\begin{`*LatexName*`}` y `\end{`*LatexName*`}`. Por omisión, verdadero.

End Requerido al final de las declaraciones de formato del recuadro `InsetLayout`.

Font La tipografía usada tanto para el cuerpo del texto como para la etiqueta. Véase la sec. 5.3.12. Anotar que al definir esta tipografía se define automáticamente el mismo valor para la etiqueta, `LabelFont`, así que define aquella primero y después define `LabelFont` si las quieres diferentes.

FixedWidthPreambleEncoding [*0*, 1] Fuerza una codificación de ancho fijo para el contenido traducido del código `BabelPreamble` y `LangPreamble` generado por este formato. Esto es necesario para paquetes especiales LATEX como `listings` que no trabajan con codificaciones de ancho variable como utf8. Esta configuración se ignora si se usan soportes LATEX compatibles totalmente con Unicode como XeTEX o LuaTEX.

ForceLocalFontSwitch [*0*, 1] Cuando se usa babel, usar siempre un cambiador de fuente local (`\foreignlanguage`), nunca uno global (como `\selectlanguage`).

ForceLTR [*0*, 1] Fuerza el idioma «latex» que genera salida Izquierda-Derecha (latin), p. e. en código TEX o URL. Un parche.

ForceOwnlines [*0*, 1] Fuerza un salto de línea en la salida LATEX antes de iniciarse el recuadro de inserción y después de finalizar. Esto asegura que el recuadro mismo se ubique en sus propias líneas, con fines de análisis.

ForcePlain [*0*, 1] Indica si debería usarse `PlainLayout` o, por el contrario el usuario puede cambiar el estilo de párrafo del recuadro.

FreeSpacing [*0*, 1] Como en estilos de párrafo, véase sec. 5.3.7.

HTML* Controlan la salida XHTML. Véase la sec. 5.4.

InToc [*0*, 1] Incluye o no el contenido de este inset en la cadena generada para el panel 'Navegador de contorno' for all table of contents, regardless of the AddToToc setting. El no sería, por ejemplo, para que el contenido de una nota al pie en un encabezado de sección no aparezca en el índice que se despliega en el navegador de contorno, sin embargo, normalmente sí se incluiría el contenido de un estilo de carácter. Predeterminado es false: no incluir.

IsTocCaption [*0*, 1] If this is set to 1 and AddToToc is enabled, the inset adds a summary of its contents in its item in the table of contents. Otherwise, only the label appears.

KeepEmpty [0, 1] Como en estilos de párrafo, véase sec. 5.3.7.

LabelFont La tipografía para la etiqueta. Véase la sec. 5.3.12. Anotar que esta definición nunca puede aparecer antes de **Font**, para que sea efectiva.

LabelString [string=""] Lo que se mostrará sobre el botón u otra parte como etiqueta del recuadro. Algunos tipos de recuadro (**código TEX** y **Ramas**) modifican esta etiqueta sobre la marcha.

LangPreamble Preámbulo dependiente del idioma; véase sec. 5.3.8.

LatexName [<name>] El nombre LaTeX correspondiente al asunto. Cualquier nombre de entorno o de comando.

LatexParam [<parameter>] El parámetro opcional para el correspondiente **LatexName**, incluyendo posibles pares de corchetes, **[]**. Este parámetro no se puede cambiar desde dentro de LyX (usa **Argument** para parámetros personalizables). Se emitirá como tal después de todos los **Argument** LaTeX.

LatexType [Command, Environment, None] Cómo debería traducirse el estilo en LaTeX.[19]

None significa nada especial

Command significa *LatexName*{...}

Environment significa \begin{*LatexName*}...\end{*LatexName*}

Poniendo juntas las últimas cosas, la salida LaTeX sería:

```
\LatexName[LatexParam]{...}
```

o:

```
\begin{LatexName}[LatexParam] ... \end{LatexName}
```

dependiendo del tipo de LaTeX.

LeftDelim [string] Secuencia que se pone al comienzo del contenido del formato. Un salto de línea en la salida se indica con
.

LyxType Puede ser charstyle, custom, element, o end (indicando una definición ficticia que finaliza definiciones de estilos de texto, etc). Esta entrada es requerida y solo es significativa para recuadros flexibles. Entre otras cosas, determina en qué menús aparecerá este recuadro. Si LyxType es charstyle MultiPar es falso y ForcePlain to true. MultiPar se puede poner como verdadero, o ForcePlain to false, para recuadros de estilos charstyle, si poniéndolo *después* de LyxType.

MultiPar [*0*, 1] Indica si se permiten múltiples párrafos en este recuadro. Además establecerá para CustomPars el mismo valor y para ForcePlain el valor opuesto. Estos pueden modificarse a otros valores si se usan *después* de MultiPar.

NeedProtect [*0*, 1] Si los comandos frágiles deben o no ser \protect'ed en este recuadro. (Nota: es *no* si el comando mismo debe ser protegido.*)*

NoInsetLayout [<layout>] Borra un InsetLayout existente.

ObsoletedBy [<layout>] Nombre de un InsetLayout que ha recolocado este InsetLayout. Se usa para renombrar un InsetLayout, mientras se mantiene compatibilidad hacia atrás.

ParbreakIsNewline [*0*, 1] Igual que para estilos de párrafo, véase sec. 5.3.7. Por omisión es falso.

PassThru [0, 1] Igual que para estilos de párrafo, véase sec. 5.3.7.

Preamble Igual que para estilos de párrafo, véase sec. 5.3.7.

RefPrefix [string] Prefijo a usar cuando se crean etiquetas que se refieren a recuadros de este tipo. Esto permite el uso de referencias con formato.

Requires [string] Igual que para estilos de párrafo, véase sec. 5.3.7.

ResetArgs [*0*, 1] Reinicia los argumentos LaTeX de este formato (como se definen con Argument tag). Es útil si has copiado un estilo con CopyStyle, pero no quieres heredar sus argumentos (requerido y opcional).

ResetsFont [0, *1*] Si este recuadro debería usar la tipografía de su entorno o la suya propia. Por omisión es la tipografía de su entorno.

RightDelim [string] Secuencia que se pone al final del contenido del formato. Un salto de línea en la salida se indica con
.

Spellcheck [0, *1*] Revisar la ortografía del contenido del recuadro. Por omisión, sí.

[19] LatexType es quizás un poco confuso porque estas eglas se aplican también a clases. Mira los archivos de clase SGML para ejemplos concretos.

5.3.11. Contadores

Es necesario definir los contadores (CHAPTER, FIGURE, ...) en la propia clase de texto. Los contadores estándar están definidos en el archivo `stdcounters.inc`, de modo que no tienes más que añadir

```
Input stdcounters.inc
```

en el archivo de formato para que funcionen. Pero si quieres puedes definir contadores personalizados. La declaración de un contador debe comenzar con:

```
Counter <name>
```

donde `<name>` es el nombre del contador. Hay que terminar con `End`.

Se pueden usar además los parámetros siguientes:

`InitialValue` [`int=1`] Establece el valor inicial para el contador, al cuál se reiniciará cada vez que suceda esto. Normalmente será 1.

`LabelString` [`string=""`] Si se define esta cadena, indica cómo se muestra el contador. Al establecer este valor también se da el mismo valor a `LabelStringAppendix`. Para los valores de 'string' se pueden usar las estructuras siguientes:

- `\thecounter` será reemplazado por la expansión de `LabelString` (o `LabelStringAppendix`) del contador `counter`.
- los valores del contador pueden expresarse usando macros tipo LaTeX como `\`*`numbertype`*`{`*`counter`*`}`, donde *`numbertype`* puede ser:[20] `arabic`: 1, 2, 3,...; `alph` para minúsculas: a, b, c, ...; `Alph` para mayúsculas: A, B, C, ...; `roman` números romanos en minúscula: i, ii, iii, ...[21]; `Roman` para romanos en mayúsculas: I, II, III...; `hebrew` para números hebreos.

Si no se define `LabelString`, se construye un valor por omisión así: si el contador tiene un contador maestro `master` (definido mediante `Within`), se usa la cadena `\themaster.\arabic{counter}`; si no, se usa la cadena `\arabic{counter}`.

`LabelStringAppendix` [`string=""`] Lo mismo que `LabelString`, pero para el apéndice.

`PrettyFormat` [`string=""`] Un formato para usar con las referencias a este contador. Por ejemplo, podríamos querer las referencias a los números de sección en la forma «Sección 2.4». La secuencia debería contener "##". Esto será reemplazado por el número correspondiente del contador. Así pues, para secciones sería: Section ##.

`Within` [`string=""`] Si esto se establece para el nombre de otro contador, el presente contador se reiniciará cada vez que el otro aumente. Por ejemplo, `subsection` se numera dentro de `section`.

[20] En realidad la cosa es un poco más compleja: cualquier *numbertype* distinto de los descritos a continuación generará números arábigos. No sería sorprendenta ver este cambio en el futuro.

[21] *N. del T.*: El estilo spanish de babel los transforma automáticamente en versalitas, I, II..., ya que los romanos en minúscula no se usan en español.

5.3.12. Descripción de la tipografía

Una descripción de una tipografía se ve así:

```
Font o LabelFont o DefaultFont
  ...
EndFont
```

Están disponibles los comandos siguientes:

Color [*none*, black, white, red, green, blue, cyan, magenta, yellow, brown, darkgray, gray, lightgray, lime, orange, olive, pink, purple, teal, violet]

Family [Roman, Sans, Typewriter]

Misc [string] Son argumentos válidos: emph, noun, strikeout, underbar, uuline, uwave, no_emph, no_noun, no_strikeout, no_bar, no_uuline y no_uwave. Cada uno de ellos activa o desactiva el atributo correspondiente. Por ejemplo, emph activa énfasis, y no_emph lo desactiva.
Si esto último parece superfluo, recordemos que la configuración de tipografías para el presente contexto se hereda generalmente del contexto circundante. Por eso no_emph desactivaría el énfasis que estaba vigente, digamos, en un entorno teorema.

Series [Medium, Bold]

Shape [Up, Italic, SmallCaps, Slanted]

Size [tiny, small, *normal*, large, larger, largest, huge, giant]

5.3.13. Citation engine description

The CiteEngine blocks, as used mainly in cite engine files (see subsection 5.2.6), define the citation commands provided by a specific "cite engine". A cite engine, in LyX terms, is way specific way to format citations, using numbers, author names and/or years. Currently, LyX supports three such engine types, namely:

1. default: the default BibTeX way to format citations, a simple numeric style (e. g., "[1]")
2. authoryear: Harvard-styled citations using author names and publication year (e. g., "Smith and Miller (2017b)")
3. numerical: extended numerical citations that also allow for author or title next to the number (e. g., "Smith and Miller [1]")

CiteEngine blocks look like this:

```
CiteEngine default
 cite
 Citep*[][]
 citeyearpar[][]=parencite*
 ...
End
```

The tag following `CiteEngine` denotes the engine. The individual lines respectively define a cite command or cite command paradigm supported by this engine. The line can be as simple as a cite command that is used both to name the respective LyX command and the LaTeX output or more complex in order to differentiate things. The full syntax is:

```
LyXName|alias$*<!_stardesc!_stardesctooltip>[][]=latexcmd
```

- `LyXName`: The name as used in the `*.lyx` file.

  For portability reasons, we try to use the same name for same-formatted commands in different cite packages (thus many names stem from natbib, and thus we need to differentiate a `latexcmd` sometimes, if the LaTeX command names differ).

- `alias`: a (comma-separated) list of commands that fall back to the given `LyXName` in the current engine. This eases the switch of citation packages and engines. The `alias` can be compared to `ObsoletedBy` in layout definitions.

- `latexcmd`: The actual LaTeX command that is output.

`Alias` and `latexcmd` are optional. If no `latexcmd` is given, the `LyXName` will be output to LaTeX.

Note further:

- Capitalization indicates that the command also has a capitalized form (`\Latexcmd` vs. `\latexcmd`). These usually enforce up-casing of name prefixes (*von Goethe* ⇒ *Von Goethe*).

- Brackets `[]` indicate the number of optional arguments (there can be 0–2).

- A star `*` indicates there is a starred version of the command (`\latexcmd*` vs. `\latexcmd`).

  By default, the starred version means: Output all authors even if it should be shortened with "et al." due to the `MaxCiteNames` threshold.

  If the star has a different meaning for a given command, it can be specified in angle brackets: `<!_stardesc!_stardesctooltip>`. Maximal two translatable macro keywords, marked by the prefix `!_`, can be given. The first points to

the string that replaces the "Full aut&hor list" checkbox label in the citation dialog, the second one to an optional tool tip for this checkbox.

Note that these two macros have to be defined in a `CiteFormat` (see next section), dropping the `!` from the prefix, like this:

```
_stardesc Sta&rred command label
_stardesctooltip Tooltip for the starred command checkbox.
```

- A dollar sign `$` indicates that this command features "qualified citation lists". This is a Biblatex-specific feature for multi-reference citations where an individual pre- and postnote can be given to each reference in the list. Please refer to the Biblatex manual for details.

5.3.14. Descripción del formato de cita

Los bloques `CiteFormat` se usan para describir la forma en que debe mostrarse la información bibliográfica, tanto en LyX (en el diálogo de citas y en las ayudas emergentes, por ejemplo) como en la salida XHTML. El aspecto de unode estos bloques podría ser este:

```
CiteFormat
 article ...
 book ...
End
```

or

```
CiteFormat
 cite ...
 citet*[][] ...
End
```

In the first case, the individual linesdefinen cómo se va a mostrar la información bibliográfica asociada con un artículo o libro, respectivamente, y tales definiciones se pueden dar para cualquier 'tipo de entrada' que pudiera estar presente en un archivo BibTeX. LyX define un formato por omsión en el código fuente que se usará si no se da una definición específica. LyX predefine varios formatos en el archivo `stdciteformats.inc`, que se incluye en la mayoría de las clases de documento de LyX.

In the second case, the lines define how a specific citation command (in the example `\cite`, `\citet`) is to be displayed on the citation inset label, in the citation dialog, menu or XHTML output. LyX defines such formats for the citation style variants it supports via Document ▷ Setting ▷ Bibliography... in specific `*.citeengine` files that are shipped with LyX (see subsection 5.2.6).

Las definiciones usan un lenguaje simple que permite reemplazar las claves BibTEX por sus valores. Las claves deben encerrarse entre símbolos `%`, p. ej.: `%author%`. Así, una simple definición sería:

```
misc%author%, "%title%".
```

Esto imprimiría el autor, una coma, a continuación el título entre comillas y un punto para finalizar.

A veces querrás, por supuesto, imprimir una clave solo si existe. Esto puede hacerse mediante una construcción condicional tal como: `{%volume%[[vol.%volume%]]}`, que quiere decir: si la clave `volume` existe, imprime "vol. " seguido de la clave. También es posible tener una cláusula de otro modo en el condicional, como:
`{%author%[[%author%]][[%editor%, ed.]]}`
Aquí, la clave `author` se imprime si existe; por otra parte, se imprime la clave `author` seguida por ", ed.". Advierte que la clave se encierra entre signos `%`; el condicional entero se encierra entre llaves; y las cláusulas condicional y por otra parte se ponen entre cobles corchetes, "`[[`" y "`]]`". No debe haber espacios entre ninguno de ellos.

Next to the entry keys, there are some special keys that can be used for these conditionals:

- `{%dialog%[[true]][[false]]}`: process the "true" part for dialogs and menus, the "false" part for other contexts (workarea, export)
- `{%export%[[true]][[false]]}`: process the "true" part for export and menus, the "false" part for other contexts (workarea, dialog)
- `{%next%[[true]]}`: process the "true" part if another item follows (e. g., in a citation with multiple keys)
- `{%second%[[true]][[false]]}`: process the "true" if this is the second of multiple items, else the "false" part
- `{%ifstar%[[true]][[false]]}`: process the "true" part for starred citation commands (such as `\cite*`), the false part for unstarred
- `{%ifentrytype:<type>%[[true]][[false]]}`: process the "true" if the current entry type matches `<type>`, else the false part (e.g., in a citation definition: `{%ifentrytype:book%[[this is a book]][[this is no book]]}`)
- `{%ifmultiple:<authortype>%[[true]][[false]]}`: process the "true" if the current author type (author, editor etc.) has multiple authors, else the false part (e.g., in a bibliography definition: `{%ifmultiple:editor%[[eds.]][[ed.]]}`)
- `{%ifqualified%[[true]][[false]]}`: process the "true" part if the current citation is a qualified citation list (a specific Biblatex format for multi-reference citations), the false part if this is not the case.

We said that `%author%` prints the author key as it is recorded in the bibliography file. This might not be what you want, since it will result in a string such as "Miller, Peter and Smith, Mary and White, Jane" (since "and" is used by BibTEX to delimit authors). LyX therefore provides some methods to get properly formatted name lists (which will also get translated). The following keys are provided:

1. For name lists with pre- and surname, suitable for the main authors/editors of a bibliography item. The `<nametype>` part denotes the kind of list that is requested (e.g. `<nametype:author>`):
   - `%abbrvnames:<nametype>%`: Provides a name list which is abbreviated (with "et al.") when `MaxCiteNames` is reached.
   - `%fullnames:<nametype>%`: Provides a full name list (never abbreviated with "et al.").
   - `%forceabbrvnames:<nametype>%`: Provides a name list which is always abbreviated (with "et al.") irrespective of `MaxCiteNames`.
2. Alternative name lists with pre- and surname, if the order of pre- and surname inside the bibliography item differs (as in: "Miller, John: Some text, in: Mary Smith, ed.: A volume"):
   - `%abbrvbynames:<nametype>%`: Provides a name list which is abbreviated (with "et al.") when `MaxCiteNames` is reached.
   - `%fullbynames:<nametype>%`: Provides a full name list (never abbreviated with "et al.").
   - `%forceabbrvbynames:<nametype>%`: Provides a name list which is always abbreviated (with "et al.") irrespective of `MaxCiteNames`.
3. And finally name lists which consist of family names only, as used in author-year citation labels. these do not take a `<nametype>` part, but always return either an author list or, if this does not exist, an editor list (as common in author-year labels):
   - `%abbrvciteauthor%`: Provides a name list which is abbreviated (with "et al.") when `MaxCiteNames` is reached.
   - `%fullciteauthor%`: Provides a full name list (never abbreviated with "et al.").
   - `%forceabbrvciteauthor%`: Provides a name list which is always abbreviated (with "et al.") irrespective of `MaxCiteNames`.

The order of pre- and surname in the former two lists can be adjusted by these macros:

- `!firstnameform %surname%, %prename%` (first author in lists of type 1)
- `!othernameform %surname%, %prename%` (other authors in lists of type 1)

- `!firstbynameform %prename% %surname%` (first author in lists of type 2)
- `!otherbynameform %prename% %surname%` (other authors in lists of type 2)

This allows you to configure namings like "Miller, Peter and Mary Smith: ..., in: John Doe and Pat Green, eds.:...".

Hay otra pieza de sintaxis disponible en las definiciones, como esta: `{!<i>!}`. Define una pieza de información para ser usada cuando creamos "texto enriquecido". Obviamente, no queremos etiquetas HTML cuando escribimos texto sencillo, por tanto deben ponerse entre "{!" y "!}".

Además, hay dos clases especiales de definiciones disponibles en un bloque `CiteFormat`. Un ejemplo de la primera sería:

```
!quotetitle "%title%"
```

Esto es una abreviación, o macro, y se puede usar tratándola como si fuera una clave: `%!quotetitle%`. LyX tratará `%!quotetitle%` exactamente como trataría su definición. Por tanto, permítenos un *aviso* obvio. No hagas:

```
!funfun %funfun%
```

ni nada parecido. LyX no caería en un bucle infinito, pero sí en uno muy largo.

El segundo tipo especial de definición sería como:

```
B_pptext pp.
```

Esto define un trozo de texto traducible, que permite traducir partes relevantes de la bibliografía o de la cita. Puede incluirse en una definición tratándolo como una clave: `%B_pptext%`. Note that there are two different translation paths: All definitions starting with `B_`, such as in the example above, will be translated to the currently active buffer language (so the translation will match the generated document). All definitions starting with underscore only will be translated to the GUI language. This is the proper translation for strings that only occur in the dialogs or on buttons, such as this one:

```
_addtobib Add to bibliography only.
```

Varios de estos cadenas de caracteres traducible están predefinidos en `stdciteformats.inc` y en los diversos archivos `*.citeengine`. Advertiremos que no son macros en el sentido definido. No se expandirán.

Aquí hay, pues, un ejemplo que usa varias estas características:

```
!authoredit {%author%[[%author%, ]][[{%editor%[[%editor%, %B_edtext%, ]]}]]}
```

Así definimos una macro que imprime el autor seguido de coma, si la clave `author` está definida, o imprime el nombre del editor seguido por el texto `B_edtext` o su traducción (por omisión "ed."), si la clave `editor` está definida. De hecho, esto está definido en `stdciteformats.inc`, por tanto puedes usarlo en tus propias definiciones o redefiniciones, si cargas antes ese archivo.

5.4. Etiquetas para la salida XHTML

Como para LaTeX o DocBook, el formato de salida XHTML en LyX también se controla mediante la información contenida en archivos de formato. En general, LyX provee valores predefinidos sensatos y, como se mencionó anteriormente, incluso construirá reglas de estilo CSS a partir de las otras etiquetas de formato. Por ejemplo, LyX intentará usar la información proporcionada por la declaración `Font` para el estilo Chapter para escribir CSS que formateará adecuadamente los encabezados de capítulo.

En muchos casos, por tanto, no tendrás que hacer nada para obtener una aceptable salida XHTML para tus propios entornos, recuadros personalizados y cosas así. Pero en ciertos casos querrás hacer cambios, y para eso LyX proporciona un número de etiquetas de formato que se pueden usar para personalizar los XHTML y CSS generados.

Advertimos que hay dos etiquetas, `HTMLPreamble` y `AddToHTMLPreamble` que pueden aparecer fuera de las declaraciones de estilos y recuadros. Véase sec. 5.3.5 para detalles sobre esto.

5.4.1. Estilos de párrafo

El tipo de XHTML que genera LyX para un párrafo depende de si se trata de un párrafo normal, de un comando o de un entorno, en los que esto se determina por el contenido de la correspondiente etiqueta `LaTeXType`.

Para un comando o párrafo, la salida XHTML tiene la forma siguiente:

```
<tag attr="value">
<labeltag attr="value">Label</labeltag>
Contents of the paragraph.
</tag>
```

Por supuesto, 'label tags´ se omiten si el párrafo no tiene etiqueta.

Para un entorno que no sea algún tipo de lista, XHTML toma la forma:

```
<tag attr="value">
<itemtag attr="value"><labeltag attr="value">Environment La-
bel</labeltag>First paragraph.</itemtag>
<itemtag>Second paragraph.</itemtag>
</tag>
```

La etiqueta se genera solo para el primer párrafo, como debería ser, por ejemplo, para un teorema.

Para una lista disponemos de una de estas formas:

```
<tag attr="value">
```

```
<itemtag attr="value"><labeltag attr="value">List La-
bel</labeltag>First item.</itemtag>
<itemtag attr="value"><labeltag attr="value">List La-
bel</labeltag>Second item.</itemtag>
</tag>
<tag attr="value">
<labeltag attr="value">List La-
bel</labeltag><itemtag attr="value">First item.</itemtag>
<labeltag attr="value">List La-
bel</labeltag><itemtag attr="value">Second item.</itemtag>
</tag>
```

Observa los diferentes órdenes de `labeltag` e `itemtag`. El orden que obtenemos depende del ajuste de `HTMLLabelFirst`: si `HTMLLabelFirst` es falso (por omisión), se obtiene el primero de ellos, con la etiqueta dentro del ítem; si verdadero, se obtiene el segundo, con la etiqueta fuera del ítem.

Las etiquetas específicas y la salida de los atributos para cada tipo de párrafo puede ser controlado por medio de las etiquetas de formato que vamos a describir. Como mencionamos antes, sin embargo, LyX usa predefinidos adecuados para muchos de estos valores, por lo que generalmente no necesitarás hacer nada para obtener una buena salida XHTML. Partiendo de las etiquetas ya disponibles puedes afinar las cosas a tu gusto.

`HTMLAttr` [`string`] Especifica información de los atributos a entregar con la etiqueta principal. Por ejemplo, "`class='mydiv'`". Por omisión, LyX producirá "`class='layoutname'`", donde `layoutname` es el nombre del formato en LyX, en minúsculas, por ejemplo: chapter. Esto *no* debería contener ninguna información de estilo. Usa `HTMLStyle` para ese propósito.

`HTMLForceCSS` [*`0`*, `1`] Si producir la información CSS predeterminada que LyX genera para este formato, incluso si se proporciona explícitamente información adicional mediante `HTMLStyle`. Poner `1` permite alterar o aumentar el CSS generado, no sobrescribirlo totalmente. Por omisión es `0`.

`HTMLItem` [`string`] Etiqueta a usar para párrafos individuales de entornos, reemplazando `itemtag` en los ejemplos de arriba. Por omisión, `div`.

`HTMLItemAttr` [`string`] Atributos para la etiqueta ítem. Por omisión "`class='layoutname_item'`". Esto *no* debería contener ninguna información de estilo. Usa `HTMLStyle` para ese propósito.

`HTMLLabel` [`string`] Etiqueta a usar para rótulos de párrafos e ítems, reemplazando `labeltag` en los ejemplos de arriba. Por omisión, `span`, a menos que `LabelType` sea `Top_Environment` o `Centered_Top_Environment`, en cuyos casos es `div` por omisión.

HTMLLabelAttr [string] Atributos para la etiqueta del rótulo. Por omisión es "class='layoutname_label'". Esto *no* debería contener ninguna información de estilo. Usa HTMLStyle para ese propósito.

HTMLLabelFirst [*0*, 1] Solo es significativo para entornos tipo lista, esta etiqueta controla si la etiqueta de rótulo se genera antes o dentro de la etiqueta de ítem. Se usa, por ejemplo, en el entorno descripción, donde queremos '<dt>...</dt><dd>...</dd>. Por omisión, 0: la etiqueta de rótulo se genera dentro de la etiqueta de ítem.

HTMLPreamble Información para generar en la sección <head> cuando se usa este estilo. Esto podría, por ejemplo, utilizarse para incluir un bloque <script> definiendo un controlador onclick.

HTMLStyle Información de estilo CSS a incluir cuando se usa este estilo. Advertimos que esto será automáticamente envuelto en un bloque <style> generado por el formato, por tanto solo es necesario incluir el propio CSS. Debe acabar con EndHTMLStyle.

HTMLTag [string] Etiqueta para el rótulo principal, reemplazando tag en losejemplos de arriba. Por omisión es div.

HTMLTitle [*0*, 1] Marca este estilo como el que se va a usar para generar la etiqueta <title> para el archivo XHTML. Por omisión, es falso. El archivo stdtitle.inc lo establece verdadero para el entorno title.

5.4.2. Recuadros XHTML

La salida XHTML de los recuadros también se puede controlar por información en archivos 'layout'.[22] También en este caso, LyX intenta proporcionar valores predefinidos adecuados, y construye reglas de estilo CSS, pero todo puede personalizarse.

Para los recuadros, LyX produce XHTML en la forma:

```
<tag attr="value">
<labeltag>Label</labeltag>
<innertag attr="value">Contents of the inset.</innertag>
</tag>
```

Si el recuadro permite párrafos múltiples —o sea, si MultiPar es verdadero— el contenido del recuadro se generará en forma de párrafos estructurados según los estilos usados para dichos párrafos (normal, cita, y similares). El rótulo de la etiqueta se omite si el párrafo no lo tiene y, por el momento, es siempre span. La etiqueta interior es opcional y, por omisión, no aparece.

Las etiquetas y atributos específicos para cada recuadro se pueden controlar mediante las siguientes etiquetas de formato.

[22] Por ahora, esto solo es cierto para recuadros de "texto" (recuadros en los que puedes escribir) y no para recuadros de "commandos" (recuadros asociados con cuadros de diálogo).

HTMLAttr [string] Especifica información de atributos a generar con la etiqueta principal. Por ejemplo, "class='myinset' onclick='...'". Por omisión, LyX generará "class='insetname'", donde insetname es el nombre del recuadro en LyX, en minúsculas y con los caracteres no alfanuméricos convertidos en guiones bajos, por ejemplo: footnote.

HTMLForceCSS [*0*, 1] Si producir la información CSS predeterminadad que LyX genera para este formato, incluso si se da explícitamente información adicional con HTMLStyle. Si esto es 1 parmite alterar o aumentar el CSS generado, en vez de sobrescribirlo. Por omisión, 0.

HTMLInnerAttr [string] Atributos para la etiqueta interna. Por omisión, es "class='insetname_inner'".

HTMLInnerTag [string] La etiqueta interna, reemplazando innertag en los ejemplos de arriba. Por omisión, ninguna.

HTMLIsBlock [0, *1*] Si este recuadro representa un bloque de texto independiente (como una nota al pie) o bien representa material que está incluido en el texto circundante (como una rama). Por omisión, 1.

HTMLLabel [string] Un rótulo para este recuadro, posiblemente incluyendo una referencia a un contador. Por ejemplo, para notas al pie, podría ser: \arabic{footnote}. Esto es opcional y no hay valor predeterminado.

HTMLPreamble Información que saldrá en la sección <head> cuando se usa este estilo. Esto podría usarse, por ejemplo, para incluir un bloque <script> que defina un controlador onclick.

HTMLStyle Información de estilo CSS a incluir cuando se usa este estilo. Esto será automáticamente envuelto en un bloque <style> generado por el formato, por lo que solo hay que incluir el propio CSS.

HTMLTag [string] La etiqueta a usar para el rótulo principal, reemplazando tag en los ejemplos de arriba. El valor por omisión depende de MultiPar: Si MultiPar es verdadero, es div; si es falso, el valor por omisión es span.

5.4.3. Flotantes XHTML

La salida XHTML para flotantes también se puede controlar mediante información en archivos 'layout'. La salida tiene la forma:

```
<tag attr="value">
Contents of the float.
</tag>
```

La leyenda, si la hay, es un recuadro separado y se generará como tal. Su aspecto puede controlarse con InsetLayout para recuadros de leyendas.

`HTMLAttr` [`string`] Especifica información de los atributos a generar con la etiqueta principal, Por ejemplo, "`class='myfloat' onclick='...'`". Por omisión, LyX generará "`class='float float-floattype'`", donde `floattype` es el nombre en LyX para este tipo de flotante, como determina la declaración de flotante (véase sec. 5.3.9), aunque en minúsculas y con los caracteres no alfanuméricos sustituidos por guiones bajos, por ejemplo: float-table.

`HTMLStyle` Información de estilo CSS a incluir cuando se usa este flotante. Esto será automáticamente envuelto en un bloque `<style>` generado por el formato, por lo que solo hay que incluir el propio CSS.

`HTMLTag` [`string`] La etiqueta a usar para este flotante, reemplazando `"tag"` en el ejemplo de arriba. El valor por omisión es `div` y raramente habrá que cambiarlo.

5.4.4. Formato de la Bibliografía

La bibliografía se puede formatear usando bloques `CiteFormat`. Véase la sec. 5.3.14 para los detalles.

5.4.5. CSS generado por LyX

Hemos mencionado ya que LyX generará reglas de estilo CSS predeterminadas para recuadros y estilos de párrafo, basados en la información suministrada para otros formatos. En esta sección comentaremos qué información usa LyX y cómo la usa.

Actualmente, LyX auto-genera CSS solo para información sobre tipografías, haciendo uso de `Family`, `Series`, `Shape`, y `Size` especificados en la declaración `Font`. (Véase sec. 5.3.12.) La traducción es bastante sencilla y evidente. Por ejemplo, "`Family Sans`" se convierte en "`font-family: sans-serif;`". La correspondencia entre tamaños LyX y tamaños CSS es un poco menos obvia, pero intuitiva en todo caso. Véase la función `getSizeCSS()` en `src/FontInfo.cpp` para los detalles.

6. Inserción de material externo

AVISO: Esta parte de la documentación no ha sido actualizada desde hace tiempo. Esperamos que aún sea adecuada, pero no hay garantías.

El uso de material de procedencia externa a LyX se trata con detalle en el manual *Objetos insertados*. En el presente documento se trata sobre lo que debe suceder entre bambalinas para incluir nuevas clases de material.

6.1. ¿Cómo funciona?

La característica material externo se basa en el concepto *plantilla*. Una plantilla es una especificación de la forma en que LyX interactúa con ciertos tipos de material. De por sí, LyX incluye plantillas predefinidas para figuras Xfig, varios tipos de imágenes raster, diagramas de ajedrez y notación musical LilyPond. Puedes comprobar la lista actual en el menú Insertar ▷ Archivo ▷ Material externo. Además, es posible construir plantillas propias para soportar determinado tipo de material. Más tarde describiremos con más detalle lo que esto implica, y esperamos que nos envíes tus plantillas para que se puedan incluir en próximas versiones de LyX.

Otra idea básica de la característica material externo es distinguir entre el archivo original que sirve como base para el material final y el archivo generado que se incluye en el documento impreso o exportado. Por ejemplo, consideremos el caso de una figura hecha con `Xfig`. El programa Xfig en sí mismo trabaja sobre un archivo original con extensión `.fig`. Dentro de Xfig, se crea o modifica una figura y se guarda como un archivo `.fig`. Para incluir la figura en un documento, se ejecuta `transfig` con el fin de crear un archivo PostScript que puede incluirse fácilmente en el archivo LaTeX. En este caso, el archivo `.fig` es el original, y el archivo PostScript es el archivo generado.

Esta distinción es importante para permitir la actualización del material durante el proceso de escritura del documento. Además, provee la flexibilidad necesaria para soportar múltiples formatos de exportación. Por ejemplo, en el caso de un archivo de texto sencillo, no es una brillante idea incluir la figura como PostScript puro. En vez de esto, es preferible incluir solo una referencia a la figura o bien probar algún convertidor de gráficos a Ascii para que el resultado final se parezca al gráfico real. La gestión de material externo de LyX permite hacer esto porque se basa en parámetros apropiados para los diferentes formatos de exportación soportados.

Además de soportar la generación de diferentes productos de acuerdo con el formato de exportación, existe una profunda integración con las aplicaciones de edición y visualización. En el caso de figuras Xfig, es posible editar el archivo original en Xfig

con un simple clic derecho sobre el recuadro y también previsualizar el PostScript generado con ghostview. Se acabaron los enredos con la línea de comandos y/o el navegador de archivos para localizar y manipular los archivos originales o los generados. De esta manera, se pueden por fin aprovechar las ventajas de muy diversas aplicaciones importantes para la composición de los documentos, y por tanto aumentar el rendimiento.

6.2. El archivo de configuración de plantillas externas

Es relativamente fácil añadir en LyX definiciones personalizadas de plantillas externas. Sin embargo, hay que ser consciente de que hacer esto de manera descuidada generará muy probablemente un agujero de seguridad fácil de explotar. Así pues, antes de hacer esto, lee por favor la discusión sobre seguridad en la sec. 6.4

Una vez dicho esto, te animamos a que envíes cualquiera plantilla interesante que compongas.

Las plantillas externas se definen en el archivo `*.xtemplate` files that are stored in the `LyXDir/lib/xtemplates/` directory. Each template is defined in a file of its own. . Puedes colocar tus templates propias en`UserDir/xtemplates/` or copy existing templates to that directory in order to modify them.

Una plantilla típica tiene un aspecto como este:

```
Template XFig
GuiName "XFig: $$AbsOrRelPathParent$$Basename"
HelpText
An XFig figure.
HelpTextEnd
InputFormat fig
FileFilter "*.fig"
AutomaticProduction true
Transform Rotate
Transform Resize
Format LaTeX
TransformCommand Rotate RotationLatexCommand
TransformCommand Resize ResizeLatexCommand
Product "$$RotateFront$$ResizeFront
         \\input{$$AbsOrRelPathMaster$$Basename.pstex_t}
         $$ResizeBack$$RotateBack"
UpdateFormat pstex
UpdateResult "$$AbsPath$$Basename.pstex_t"
Requirement "graphicx"
ReferencedFile latex "$$AbsOrRelPathMaster$$Basename.pstex_t"
ReferencedFile latex "$$AbsPath$$Basename.eps"
ReferencedFile dvi "$$AbsPath$$Basename.eps"
```

```
FormatEnd
Format PDFLaTeX
TransformCommand Rotate RotationLatexCommand
TransformCommand Resize ResizeLatexCommand
Product "$$RotateFront$$ResizeFront
         \\input{$$AbsOrRelPathMaster$$Basename.pdftex_t}
         $$ResizeBack$$RotateBack"
UpdateFormat pdftex
UpdateResult "$$AbsPath$$Basename.pdftex_t"
Requirement "graphicx"
ReferencedFile latex "$$AbsOrRelPathMaster$$Basename.pdftex_t"
ReferencedFile latex "$$AbsPath$$Basename.pdf"
FormatEnd
Format Ascii
Product "$$Contents(\"$$AbsPath$$Basename.asc\")"
UpdateFormat asciixfig
UpdateResult "$$AbsPath$$Basename.asc"
FormatEnd
Format DocBook
Product "<graphic fileref=\"$$AbsOrRelPathMaster$$Basename.eps\">
         </graphic>"
UpdateFormat eps
UpdateResult "$$AbsPath$$Basename.eps"
ReferencedFile docbook "$$AbsPath$$Basename.eps"
ReferencedFile docbook-xml "$$AbsPath$$Basename.eps"
FormatEnd
Product "[XFig: $$FName]"
FormatEnd
TemplateEnd
```

Como puedes ver, la plantilla se incluye entre `Template` ... `TemplateEnd`. Contiene una cabecera que especifica algunas configuraciones generales y, por cada formato de archivo de documento primario soportado, una sección `Format` ... `FormatEnd`.

6.2.1. La cabecera de la plantilla

`AutomaticProduction true|false` Si el archivo representado por la plantilla debe ser generado por LyX. Este comando debe aparecer exactamente una vez.

`FileFilter <pattern>` Un patrón global que se usa en el diálogo del archivo para filtrar los archivos deseados. Si hay más de una posible extensión de archivo (p. e., tgif tiene `.obj` y `.tgo`), se pone algo como `"*.{obj,tgo}"`. Este comando debe aparecer exactamente una vez.

GuiName <guiname> El texto mostrado sobre el botón. Este comando debe aparecer exactamente una vez.

HelpText <text> HelpTextEnd El texto de ayuda que se muestra en diálogo. Da suficiente información sobre lo que puede proporcionar la plantilla. Este comando debe aparecer exactamente una vez.

InputFormat <format> El formato de archivo del original. Debe ser el nombre de un formato conocido por LyX (véase la sec. 3.1). Se pone un «*» si la plantilla puede manejar archivos originales o más de un formato. En este caso, LyX intentará inspeccionar el archivo mismo para deducir su formato. Este comando debe aparecer exactamente una vez.

Template <id> Un nombre único para la plantilla. No debe contener macros de sustitución (véase abajo).

Transform Rotate|Resize|Clip|Extra Este comando especifica qué transformaciones son soportadas por esta plantilla. Puede aparecer ninguna o varias veces. Este comando facilita las correspondientes pestañas en el diálogo de material externo. Cada comando `Transform` debe tener el correspondiente `TransformCommand` o bien `TransformOption` en la sección `Format`. De lo contrario la transformación no será soportada por ese formato.

6.2.2. La sección Format

Format LaTeX|PDFLaTeX|PlainText|DocBook|XHTML El formato de archivo del documento primario para el que es esta definición de formato. No toda plantilla tiene una representación sensible en los formatos de archivo de todos los documentos. No obstante, es preferible definir una sección `Format` para todos los formatos. Se usa un texto postizo si no hay representación disponible. Entonces, al menos, se puede ver una referencia al material externo en el documento exportado.

Option <name> <value> Este comando define una macro adicional, `$$<name>`, para sustitución en `Product`. `<value>` puede contener en sí mismo macros de sustitución. La ventaja sobre el uso de `<value>` directamente en `Product` es que el valor sustituido de `$$<name>` está saneado así que es un argumento opcional válido en el formato del documento. Este comando puede aparecer ninguna o más veces.

Product <text> El texto que se inserta en el documento exportado. Realmente este es el comando más importante y puede ser bastante complejo. Este comando debe aparecer exactamente una vez.

Preamble <name> Este comando especifica un fragmento de preámbulo que se incluirá en el preámbulo LaTeX. Tiene que definirse usando `PreambleDef` ... `PreambleDefEnd`. Este comando puede aparecer ninguna o más veces.

ReferencedFile <format> <filename> Este comando indica los archivos que se crean por el proceso de conversión y que son necesarios para un formato de exportación particular. Si el nombre del archivo es relativo, se interpreta como relativo al documento maestro. Este comando puede darse ninguna o más veces.

Requirement <package> El nombre de un paquete LaTeX requerido. El paquete se incuye mediante `\usepackage{}` en el preámbulo LaTeX. Este comando puede aparecer ninguna o más veces.

TransformCommand Rotate RotationLatexCommand Este comando especifica que el comando LaTeX incluido debería usarse para la rotación. Este comando puede aparecer una vez o ninguna.

TransformCommand Resize ResizeLatexCommand Este comando especifica que el comando LaTeX incluido debería usarse para cambiar el tamaño. Este comando puede aparecer una vez o ninguna.

TransformOption Rotate RotationLatexOption Este comando especifica que la rotación se hace mediante un argumento opcional. Este comando puede aparecer una vez o ninguna.

TransformOption Resize ResizeLatexOption Este comando especifica que el cambio de tamaño se hace mediante un argumento opcional. Este comando puede aparecer una vez o ninguna

TransformOption Clip ClipLatexOption Este comando especifica que el recorte se hace mediante un argumento opcional. Este comando puede aparecer una vez o ninguna.

TransformOption Extra ExtraLatexOption Este comando especifica que se usa un argumento extra opcional. Este comando puede aparecer una vez o ninguna.

UpdateFormat <format> El formato de archivo del archivo convertido. Debe ser el nombre de un formato conocido por LyX (véase Convertidores de Gestión de archivos en el diálogo Herramientas ▷ Preferencias). Este comando debe aparecer exactamente una vez. Si el formato del archivo resultante es PDF, debes especificar el formato `pdf6`. Es el formato PDF usado para incluir gráficos. Los otros formatos PDF definidos son para exportación de documentos.

UpdateResult <filename> El nombre de archivo del archivo convertido. El nombre de archivo debe ser absoluto. Este comando debe aparecer exactamente una vez.

6.2.3. Definiciones de preámbulo

El archivo de configuración de plantillas externas puede contener definiciones de preámbulo adicionales encerradas entre `PreambleDef` ... `PreambleDefEnd`. Se pueden usar por las plantillas en la sección `Format`.

6.3. El mecanismo de sustitución

Cuando la función de material externo recurre a un programa externo, se hace sobre la base de un comando definido en el archivo de configuración de la plantilla. Estos comandos pueden contener varias macros que se expanden antes de su ejecución. La ejecución tiene lugar siempre en el directorio del documento.

Además, cada vez que el material externo se va a mostrar, el nombre será generado por el mecanismo de sustitución, y la mayoría de los demás comandos en la definición de la plantilla soportan también sustitución.

Las macros disponibles son las siguientes:

$$AbsOrRelPathMaster La ruta del archivo, absoluta o relativa al documento LyX maestro.

$$AbsOrRelPathParent La ruta del archivo, absoluta o relativa al documento LyX.

$$AbsPath La ruta absoluta del archivo.

$$Basename El nombre de archivo sin ruta y sin la extensión.

$$Contents("filename.ext") Esta macro expandirá los contenidos del archivo con el nombre `filename.ext`.

$$Extension La extensión de archivo (incluyendo el punto).

$$pngOrjpg Esto será «jpg» si el archivo está en formato JPEG, si no será "png". Esto es útil para evitar conversiones innecesarias para formatos de salida que soporten tanto PNG como JPEG. La plantilla predefinida RasterImage usa esta macro para el formato de salida pdfTeX.

$$FName El nombre de archivo del archivo especificado en el diálogo de material externo. Esto es bien un nombre absoluto, bien relativo al documento LyX.

$$FPath La parte de la ruta de `$$FName` (nombre absoluto o relativo al documento LyX).

$$RelPathMaster La ruta del archivo, relativa al documento maestro LyX.

$$RelPathParent La ruta del archivo, relativa al documento LyX.

$$Sysdir Esta macro expandirá a la ruta absoluta del directorio de sistema. Esto se usa habitualmente para apuntar a los varios guiones de ayuda que se empaquetan con LyX.

$$Tempname Un nombre y ruta completa a un directorio temporal que será borrado automáticamente siempre que el documento se cierra, o se suprime la inserción de material externo.

Todas las macros de ruta contienen un separador de rastreo del directorio, así se puede construir, p. e., el nombre de archivo absoluto con `$$AbsPath$$Basename$$Extension`.

Las macros anteriores son sustituidas en todos los comandos a menos que se indique lo contrario. El comando `Product` soporta adicionalmente las siguientes sustituciones si se habilitan por los comandos `Transform` y `TransformCommand`:

`$$ResizeFront` La parte delantera del comando para cambiar el tamaño.

`$$ResizeBack` La parte trasera del comando para cambiar el tamaño.

`$$RotateFront` La parte delantera del comando para la rotación.

`$$RotateBack` La parte trasera del comando para la rotación.

La cadena de valor del comando `Option` soporta adicionalmente las siguientes sustituciones si éstas se han habilitado por los comandos `Transform` y `TransformOption`:

`$$Clip` La opción recorte.

`$$Extra` La opción extra.

`$$Resize` La opción cambio de tamaño.

`$$Rotate` La opción rotación.

Te puedes preguntar por qué hay tantas macros de ruta. Por dos razones, principalmente:

1. Los nombres de archivo relativo y absoluto deberían permanecer relativo o absoluto, respectivamente. Los usuarios pueden tener razones para preferir cualquiera de ambas formas. Los nombres relativos son útiles para documentos portables que deberían funcionar en máquinas diferentes, por ejemplo. Los nombres absolutos pueden ser requeridos por algunos programas.

2. LaTeX procesa los nombres de archivo relativos de manera diferente a LyX y otros programas en archivos anidados incluidos. Para LyX, un nombre de archivo relativo es siempre relativo al documento que contiene el nombre de archivo. Para LaTeX, es siempre relativo al documento maestro. Estas dos definiciones son idénticas si se tiene solo un documento, pero son distintas si se tiene un documento maestro que incluye documentos parciales. Esto quiere decir que los nombres de archivo relativos deben ser transformados cuando se presentan a LaTeX. Afortunadamente, LyX hace esto automáticamente si se eligen las macros correctas.

Así pues, ¿qué macros de ruta deberían emplearse en las definiciones de plantillas nuevas? La regla no es difícil:

- Usar `$$AbsPath` si se requiere una ruta absoluta.

- Usar `$$AbsOrRelPathMaster` si la cadena sustituida es algún tipo de entrada LaTeX.
- Si no, usar `$$AbsOrRelPathParent` con el fin de preservar la elección del usuario.

Hay casos especiales en los que esta regla no funciona y se necesitan, p. e., nombres relativos, pero normalmente funcionará bien. Un ejemplo de uno de estos casos es el comando `ReferencedFile latex "$$AbsOrRelPathMaster$$Basename.pstex_t"` en la plantilla XFig de arriba: no podemos usar el nombre absoluto porque el copiador para archivos `.pstex_t` necesita el nombre relativo para reescribir el contenido del archivo.

6.4. Discusión sobre seguridad

La función material externo interactúa con muchos programas externos y lo hace de forma automática, así que hemos de tener en cuenta las implicaciones de seguridad que esto conlleva. En particular, puesto que tienes la opción de incluir tus propios archivos y/o cadenas de parámetros y estos se expanden en un comando, parece posible crear un documento malicioso que ejecute comandos arbitrarios cuando un usuario ve o imprime el documento. Esto es algo que definitivamente queremos evitar.

Sin embargo, dado que los comandos de material externo se especifican solo en el archivo de configuración de la plantilla, no hay problemas de seguridad si LyX está adecuadamente configurado solo con plantillas seguras. Esto es así porque los programas externos se invocan con la llamada al sistema `execvp` más bien que con la llamada `system`, de modo que no es posible ejecutar comandos arbitrarios desde la sección del nombre de archivo o de parámetros mediante el shell.

Esto implica además que hay restricciones en las cadenas de comandos que puedes usar en las plantillas de material externo. En particular, las tuberías y la redirección no están fácilmente disponibles. Esto debe ser así para que LyX permanezca seguro. Si quieres usar algunas características de shell, deberías escribir un guión seguro para hacerlo de manera controlada, y después invocar el guión desde la cadena de comandos.

Es posible diseñar una plantilla que interactúe directamente con el shell, pero puesto que esto permitiría a un usuario malicioso ejecutar comandos arbitrarios escribiendo nombres de archivo y/o parámetros astutos, generalmente recomendamos usar solo guiones seguros que trabajen con la llamada `execvp` al sistema en forma controlada. Por supuesto, para usar en un entorno controlado, puede ser tentador caer en el uso de guiones de shell ordinarios. Si lo haces así, sé consciente de que vas a suministrar un agujero se seguridad fácilmente explotable en tu sistema. Evidentemente, es de razón que tales plantillas inseguras nunca serán incluidas en la distribución estándar de LyX, aunque alentamos a la gente a que, a la usanza del software libre, envíe nuevas plantillas. No obstante, LyX nunca tendrá plantillas inseguras tal y como se distribuye por los canales oficiales.

Incluir material externo suministra mucha potencia y has de ser cuidadoso en no introducir riesgos de seguridad a costa de esta potencia. Un sutil error en una simple línea en un guión aparentemente inocente puede abrir la puerta a graves problemas de seguridad. Así, si no comprendes totalmente estos asuntos, recomendamos consultar a un profesional entendido en seguridad o al equipo de desarrollo de LyX si tienes dudas acerca de si una plantilla dada es o no segura. Y hazlo antes de usarla en un entorno no controlado.

A. Lista de funciones soportadas por LyX en archivos de formato

accents	booktabs	feyn	listings	natbib	rotfloat	tfrupee	wasysym
amsbsy	calc	fixltx2e	longtable	nomencl	rsphrase	tipa	wrapfig
amscd	CJK	float	lyxskak	pdfcolmk	setspace	tipx	xargs
amsmath	color	framed	makeidx	pdfpages	shapepar	tone	xcolor
amssymb	covington	graphicx	marvosym	pifont	slashed	txfonts	xy
amstext	csquotes	hhline	mathdesign	pmboxdraw	soul	ulem	yhmath
amsthm	dvipost	hyperref	mathdots	polyglossia	splitidx	undertilde	
array	endnotes	ifsym	mathrsfs	prettyref	subfig	units	
ascii	enumitem	ifthen	mhchem	pxfonts	subscript	url	
bbding	esint	jurabib	multicol	refstyle	textcomp	varioref	
bm	fancybox	latexsym	multirow	rotating	textgreek	verbatim	

B. Nombres de colores disponibles para archivos de formato

Los colores listados son los colores estándar y los que se pueden modificar en las preferencias de LyX.

B.1. Color functions

The following are no real colors, but rather act on color definitions:

ignore El color es ignorado

inherit El color es heredado

none Ningún color en particular, transparente o ausente

B.2. Static colors

These are fixed colors that cannot me customized:

black

white

blue

brown

cyan

darkgray

gray

green

lightgray

lime

magenta

olive

orange

pink

purple

red

teal

violet

yellow

B.3. Dynamic colors

These are the colors allocated to specific elements in HERRAMIENTAS ▷ PREFERENCIAS:

added_space Color espacio añadido

addedtext Color texto añadido

appendix Color marcador de apéndice

background Color de fondo

bottomarea Color área inferior

branchlabel Color etiqueta de ramas

buttonbg Color fondo de botón

buttonframe Color para marcos de botones de recuadro

buttonhoverbg Color fondo de botón en foco

changebar Color barra de cambios

changedtextauthor1 Color autor 1 de texto cambiado

changedtextauthor2 Color autor 2 de texto cambiado

changedtextauthor3 Color autor 3 de texto cambiado

changedtextauthor4 Color autor 4 de texto cambiado

changedtextauthor5 Color autor 5 de texto cambiado

collapsibletext Color texto de recuadro plegable

collapsibleframe Color marco de recuadro plegable

command Color texto recuadros de comando

commandbg Color fondo recuadros de comando

commandframe Color marco recuadros de comando

comment Color etiqueta de comentarios

commentbg Color fondo de comentarios

cursor Color cursor

deletedtext Color texto borrado

deletedtextmodifier Color modificación de texto borrado

depthbar Color barra de anidación en margen

eolmarker Color marcador fin de línea

error Color cuadro de error LaTeX

footlabel Color etiqueta de notas a pie de página

foreground Foreground color

graphicsbg Color fondo de recuadros de inserción de gráficos

greyedoutbg Color fondo de recuadros de notas grises

greyedoutlabel Color etiqueta de recuadros de notas grises

greyedouttext Color texto de recuadros de notas grises

indexlabel Color etiqueta de recuadros de índice

inlinecompletion Color autofinalización en línea

insetbg Color fondo de marcador de recuadro

insetframe Color marco de marcador de recuadro

language Color marcador de palabras en otros idiomas

latex Color texto en modo LaTeX

listingsbg Color fondo de recuadro listado de código

marginlabel Color etiqueta de notas al margen

math Color texto de recuadro de ecuación

mathbg Color fondo de recuadro de ecuación

mathcorners Color marco de recuadro de ecuación fuera de foco

mathframe Color marco de recuadro de ecuación en foco

mathline Color línea de ecuación

mathmacrobg Color fondo de recuadro de macro matemática

mathmacroblend Color mezclado macro matemática

mathmacroframe Color marco de macro matemática

mathmacrohoverbg Color fondo de recuadro de macro matemática bajo el ratón

mathmacrolabel Color etiqueta macro matemática

mathmacronewarg Color plantilla de macro para nuevos parámetros

mathmacrooldarg Color plantilla de macro para antiguos parámetros

newpage Color página nueva

nonunique_inlinecompletion Color autofinalización en línea parte no única

note Color etiqueta de notas

notebg Color fondo de notas

pagebreak Color salto de página/línea

paragraphmarker Color símbolo marcador de fin de párrafo

phantomtext Color texto de recuadro fantasma

preview Color vista preliminar

previewframe Color marco vista preliminar

regexpframe Color marco regexp

scroll Color that indicates when a row can be scrolled

selection Color fondo de texto seleccionado

selectiontext Color de texto seleccionado

shadedbg Color fondo de marco coloreado

special Color texto de caracteres especiales

tabularline Color línea de cuadro/tabla

tabularonoffline Color línea de cuadro/tabla on/off

urllabel Color etiqueta de recuadros URL

urltext Color texto de recuadros URL