Centro Integrado de Formación Profesional

AVILÉS

Principado de Asturias

UNIDAD 4:
DOCUMENTACIÓN DE APLICACIONES

DESARROLLO DE INTERFACES

2º CURSO

C.F.G.S. DESARROLLO DE APLICACIONES MULTIPLATAFORMA

REGISTRO DE CAMBIOS

Versión	Fecha	Estado	Resumen de cambios
1.0	06/11/2023	Aprobado	Primera versión completa
1.1.	21/11/2023	Aprobado	Modificaciones en ayuda integrada

ÍNDICE

REGISTRO DE CAMBIOS 2
ÍNDICE 2
UNIDAD 4: DOCUMENTACIÓN DE APLICACIONES 3
4.1. Ficheros de ayuda. Formatos. 4
4.1.1. Microsoft Windows 4
4.1.2. Mac OS X 5
4.1.3. GNU / Linux 5
4.2. Herramientas de generación de ayuda 7
4.3. Ayuda genérica y sensible al contexto 14
4.4. Tablas de contenido 15
4.5. Incorporación de ayuda a la aplicación 16
4.6. Tipos de manuales 19
4.6.1. Manual de instalación 19
4.6.2. Documentación de configuración 19
4.6.3. Manual de usuario 20
4.6.4. Guía de referencia 21
4.6.5. Guía rápida 21
4.6.6. Manual de administración 22
4.7. Confección de tutoriales multimedia 23
ÍNDICE DE FIGURAS 25

BIBLIOGRAFÍA – WEBGRAFÍA 25

UNIDAD 4: DOCUMENTACIÓN DE APLICACIONES

La documentación de aplicaciones constituye el conjunto de información que se basa en qué hacen las aplicaciones, cómo lo hacen y para quién. La documentación consiste en un material que explica las características técnicas y funcionalidades que presenta la aplicación. Al menos debe proporcionar información sobre cómo se instala, la forma correcta de hacerla funcionar y cómo se mantiene.

La documentación de las aplicaciones es un aspecto sumamente importante, tanto en lo que se refiere al desarrollo como al mantenimiento de la aplicación. A veces en desarrollo no se le da mucha importancia lo que hace que se pierda la posibilidad de reutilizar parte del programa en otras aplicaciones sin necesidad de conocer el código punto por punto.

La importancia de la documentación podría compararse con la importancia de tener una póliza de seguro; cuando todo va bien no se tiene la precaución de confirmar si está vigente o no.

La documentación de una aplicación comienza en el momento en que se empieza a construir la aplicación y finaliza justo antes de entregarla al cliente. Asimismo, la documentación que se entrega al cliente debe coincidir con la versión final de los programas que componen la aplicación.

El estilo de redacción de los manuales de documentación debe cumplir estos requisitos:

- Ser concreto.
- Ser preciso.
- Definir los términos usados.
- Utilizar párrafos cortos.
- Utilizar títulos y subtítulos.
- Utilizar formas activas en vez de pasivas.
- Evitar frases largas que presenten distintos hechos.
- Evitar referirse a una información sólo con el número de referencia.

4.1. FICHEROS DE AYUDA. FORMATOS.

Los ficheros de ayuda contienen la información para ayudar al usuario a utilizar la aplicación. A continuación, se van a ver los principales formatos en los siguientes sistemas operativos:

- Microsoft Windows
- Mac OS X
- UNIX - Gnu/Linux

4.1.1. MICROSOFT WINDOWS

Dentro del sistema operativo Microsoft Windows, el primer formato creado fue en 1997 y es el conocido como Microsoft Compiled HTML Help. Posteriormente evolucionó al Microsoft Assistance Markup Language, que incorpora más garantías de seguridad.

4.1.1.1. Microsoft Compiled HTML Help

Microsoft Compiled HTML Help es un formato desarrollado por Microsoft que se lanzó en 1997 para los archivos de ayuda del sistema. Se introdujo por primera vez con el lanzamiento de Windows 98 y todavía se utilizó en las plataformas Windows XP. En 2003 Microsoft anunció que existían riesgos de seguridad asociados a este formato, por lo que se decidió no seguir utilizándolo a partir de Windows Vista y crear uno nuevo, el Microsoft Assistance Markup Language. En cualquier caso, todavía se pueden utilizar en Windows 10+.

Para crear los archivos de ayuda HTML (extensión chm) se utilizan herramientas de autoría de ayuda. Microsoft proporciona una herramienta, el Help Workshop, que puede descargarse gratuitamente. A partir de los archivos con el texto de la ayuda, un compilador de línea de comandos (hhc.exe) los compila y crea un archivo CHM. También hay una serie de herramientas de autor de terceros disponibles.

Un archivo con extensión chm constituye un conjunto de documentos de HTML y otros datos, como imágenes y JavaScript, comprimido en un solo archivo. Los archivos CHM contienen un número de características muy útiles que ayudan al usuario a encontrar lo que busca:

- Tabla de contenido
- Índice de palabras clave
- Funcionalidad de búsqueda con texto completo

4.1.1.2. Microsoft Assistance Markup Language

Microsoft Assistance Markup Language (conocido normalmente como MAML) es un lenguaje de marcas basado en XML. Este lenguaje lo desarrolló el equipo de asistencia al usuario de Microsoft a fin de ofrecer ayuda para el sistema operativo Microsoft Windows Vista. Algunas de las características de MAML han sido disponibles en .NET Framework 2, pero se han añadido más opciones con el lanzamiento de .NET Framework 3 en adelante.

El aspecto más significativo del MAML es que desplaza la producción de la asistencia al usuario con el concepto autoría estructurada (similar al DocBook). Los documentos y elementos que los constituyen se definen por el contexto en el que se encuentran. El énfasis se pone en el contenido y en las tareas que un usuario realiza con el ordenador, no en las características del software.

La estructura del MAML se divide en segmentos relacionados con un tipo de contenido:

- Resolución de problemas conceptuales
- Preguntas frecuentes
- Glosario
- Procedimiento de referencia
- Contenido reutilizable
- Tarea
- Tutorial

Cuando aparece un tema, se producen tres niveles de transformación:

- Estructura
- Presentación
- Representación

La transformación estructural contiene contenido reutilizable. La lógica condicional se aplica para determinar la estructura que debe tener el contenido cuando se muestra y el contenido del propio texto.

La transformación de la presentación permite que el contenido creado en MAML pueda transformarse en muchos formatos diferentes, incluyendo HTML, XAML, el formato de texto enriquecido y otro material impreso.

La transformación de la representación aplica hojas de estilo y muestra el contenido final a los usuarios.

4.1.2. MAC OS X

El formato nativo para MAC es el conocido como Apple Help. Este tipo de ayuda proporciona asistencia al usuario basada en el formato de archivos HTML. Gestiona y muestra lo que se conoce como libros de ayuda. Un libro de ayuda es la colección de archivos HTML que constituyen la ayuda al usuario de un producto de software, además de un índice basado en los archivos indexados generados. Al crear un libro de ayuda, los usuarios pueden acceder a la ayuda de su interfaz de usuario y verlo directamente en el visor de ayuda de Apple.

4.1.3. GNU / LINUX

La ayuda en entornos Linux está ligada a la ayuda que ofrecen los comandos de sistema de UNIX.

Para mostrar la ayuda basta con ejecutar:
man <nombre orden>

Cada página de esta salida es un documento independiente.

La mayoría de las aplicaciones gráficas sobre UNIX (sobre todo las construidas con los entornos de desarrollo GNOME y KDE) proporcionan al usuario final la documentación en HTML e incluyen visores de HTML incrustados, como Yelp, el cual se utiliza para la lectura de la ayuda dentro de la aplicación.

4.2. HERRAMIENTAS DE GENERACIÓN DE AYUDA

A la hora de desarrollar un código, es posible ayudarse de comentarios que se detallan sobre este mismo código para ofrecer documentación sobre los temas tratados. Se conoce como metainformación y su formato depende del lenguaje de programación o Framework utilizado.

Por ejemplo, en C# es posible hacer constar una serie de comentarios sobre un determinado código de la siguiente manera:

```
///<summary>
…
///</summary>
```

En este apartado se verán herramientas gratuitas para generar archivos de ayuda.

JavaDoc

JavaDoc es la herramienta oficial de Java para simplificar el proceso de documentación, permitiendo documentar el código a medida que se escribe.

A modo de ejemplo, a continuación, se detallan siete buenas prácticas para generar una buena documentación de API desde los ficheros de código fuente usando JavaDoc:

1. Seguir la especificación de comentarios para la versión de Java de la aplicación.

La herramienta utiliza lo que se conoce como "doclet" JavaDoc para analizar la estructura interna de los archivos fuente y producir los archivos de salida correspondientes. El doclet estándar genera resultados HTML, pero es posible producir resultados diferentes utilizando diferentes doclets.

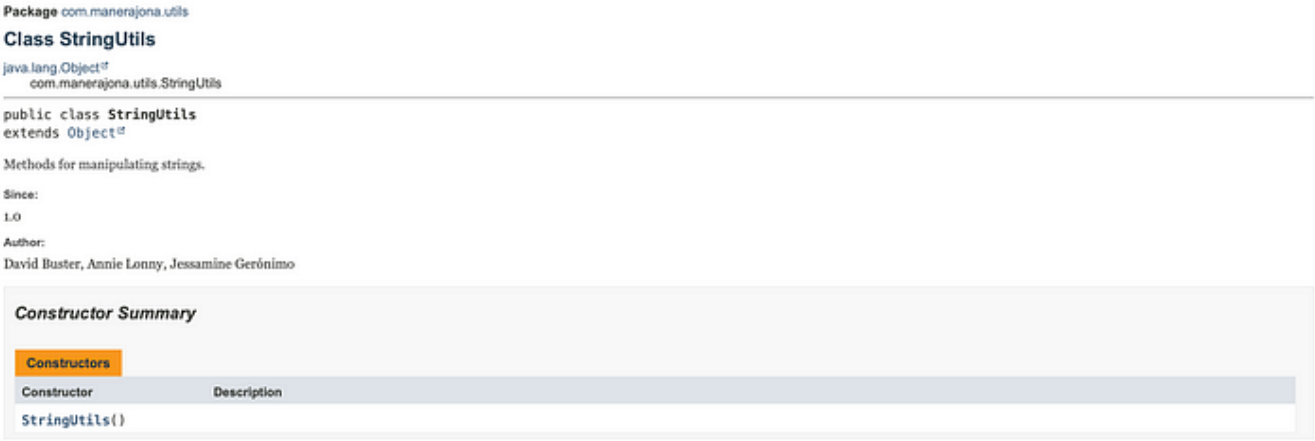
Los comentarios reconocidos por el doclet estándar están definidos por la especificación de comentarios para la versión de JDK correspondiente. Hay que tener en cuenta que podría variar entre distintas versiones de Java. Para la 21 se puede encontrar aquí:
https://docs.oracle.com/en/java/javase/21/docs/specs/javadoc/doc-comment-spec.html

2. Añadir comentarios a nivel de clase con @author y @since

Siempre está bien incluir autores y colaboradores cuando se crea una clase utilizando la etiqueta @author. En algunos casos, cuando se crean artefactos como bibliotecas, también es una buena práctica incluir la versión en la cual se ha introducido por primera vez la clase con la etiqueta @since

```java
/**
 * Methods for manipulating strings.
 *
 * @author David Buster
 * @author Annie Lonny
 * @author Jessamine Gerónimo
 * @since 1.0
 */
public class StringUtils { ... }
```

El código anterior genera la siguiente documentación HTML:



Package com.manerajona.utils

Class StringUtils

java.lang.Object
com.manerajona.utils.StringUtils

```
public class StringUtils
extends Object
```

Methods for manipulating strings.

Since:
1.0

Author:
David Buster, Annie Lonny, Jessamine Gerónimo

Constructor Summary

Constructors

Constructor	Description
StringUtils()	

1. Documentación HTML de clase Java

3. Añadir métodos a nivel de cliente incluidos parámetros de entrada @param y de salida @return y @throws.

Hay que asegurarse de incluir parámetros requeridos por el método usando la etiqueta @param. También hay que incluir posibles salidas, tanto de valor de retorno como de excepción con @return y @throws.

```java
/**
 * Returns a formatted decimal using the specified decimal pattern and argument.
 *
 * @param pattern  The pattern for the string format.
 * @param argument The object to format.
 * @return The formatted decimal.
 * @throws java.lang.IllegalArgumentException If the pattern is incompatible with the
given argument.
 * @since 1.1
 */
public static String formatDecimal(String pattern, Object argument)
    throws IllegalArgumentException { ... }
```

El código genera la siguiente ayuda en HTML:

formatDecimal

```
public static String formatDecimal(String pattern,
                                   Object argument)
                            throws IllegalArgumentException
```

Returns a formatted decimal using the specified decimal pattern and argument.

Parameters:

`pattern` - The pattern for the string format.

`argument` - The object to format.

Returns:

The formatted decimal.

Throws:

`IllegalArgumentException` - If the pattern is incompatible with the given argument.

Since:

1.1

2. Documentación HTML de método Java

4. Clases y métodos relacionados referenciados con @see y {@link}

Añadir enlaces incrustados usando la etiqueta {@link} y referencias con @see para el módulo especificado, paquete, clase o miembro de una clase referenciada.

Ambas etiquetas aceptan la misma sintaxis para la etiqueta modulo/paquete.clase#miembro

```
/**
 * Returns the total with the currency symbol and the formatted decimal amount.
 *
 * @param currency {@link Currency} in which the amount is expressed.
 * @param amount   the total amount in {@link BigDecimal}.
 * @return The total using currency symbol and decimal format, e.g. $100,000.00.
 * @see #formatDecimal(String, Object)
 * @since 1.2
 */
public static String formatTotal(Currency currency, BigDecimal amount) { ... }
```

Se genera la siguiente documentación:

formatTotal

```
public static String formatTotal(Currency currency,
                                 BigDecimal amount)
```

Returns the total with the currency symbol and the formatted decimal amount.

Parameters:

`currency` - `Currency` in which the amount is expressed.

`amount` - the total amount in `BigDecimal`.

Returns:

The total using currency symbol and decimal format, e.g. $100,000.00.

Since:

1.2

See Also:

`formatDecimal(String, Object)`

3. Documentación HTML de referencias

5. Usar HTML

Usar etiquetas HTML en comentarios para generar documentación más comprensible.

```
/**
 * Returns the formatted phone number based on the following patterns:
 * <ul>
 * <li>Pattern for 10-digit numbers: <em>(XXX) XXX-XXXX</em></li>
 * <li>Pattern for 7-digit numbers: <em>XXX-XXXX</em></li>
 * </ul>
 * <p><b>Note: The method will remove any non-digit characters from the input
string.</b>
 *
 * @param phoneNumber the string phone number.
 * @return the formatted phone number.
 * @since 1.3
 */
public static String formatPhoneNumber(String phoneNumber) { ... }
```

lo que genera la siguiente documentación:


formatPhoneNumber

public static String formatPhoneNumber(String phoneNumber)

Returns the formatted phone number based on the following patterns:
- Pattern for 10-digit numbers: *(XXX) XXX-XXXX*
- Pattern for 7-digit numbers: *XXX-XXXX*

Note: The method will remove any non-digit characters from the input string.

Parameters:
phoneNumber - the string phone number

Returns:
the formatted phone number

Since:
1.3

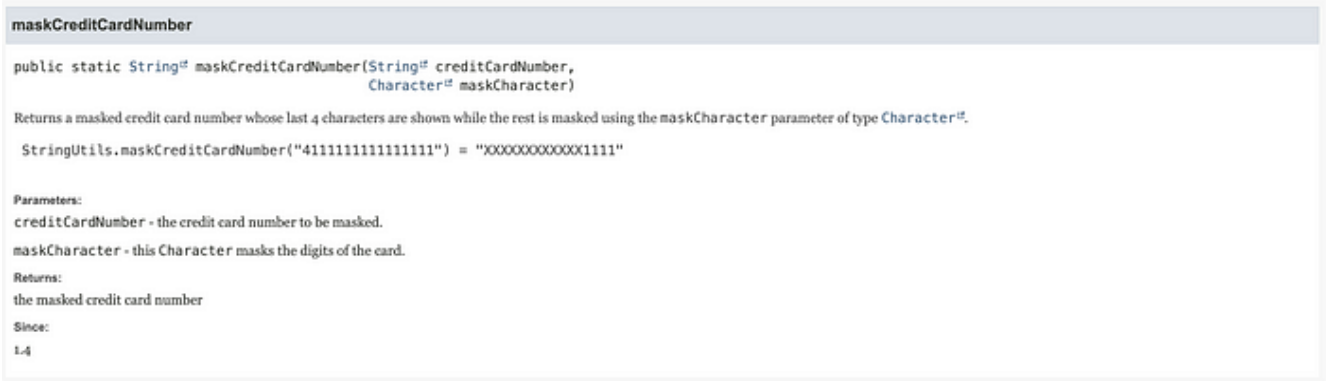
4. Código HTML incrustado en la ayuda

6. Formatea porciones de código con {@code} y <pre></pre>

Añadir porciones de código incrustadas mediante la etiqueta {@code}, lo que muestra el texto en un tipo de letra adecuado a código sin interpretarlo como código HTML o etiquetas anidadas de JavaDoc. La etiqueta <pre></pre> se usa para representar texto pre-formateado en HTML. La diferencia clave con {@code} es que esta soporta saltos de línea y sangrías.

```
/**
 * Returns a masked credit card number whose last 4 characters are shown while the
rest is masked using the
 * {@code maskCharacter} parameter of type {@link Character}.
 * <br>
 * <pre>
 * StringUtils.maskCreditCardNumber("4111111111111111") = "XXXXXXXXXXXX1111"
 * </pre>
 * @param creditCardNumber the credit card number to be masked.
 * @param maskCharacter    this {@code Character} masks the digits of the card.
 * @return the masked credit card number.
 * @since 1.4
 */
public static String maskCreditCardNumber(String creditCardNumber, Character
maskCharacter) { ... }
```

El código genera la siguiente documentación:



5. Inclusión de código en la ayuda

7. Generar documentación y ver sus resultados

El plugin maven-javadoc-plugin simplifica el proceso de generar Javadocs para el proyecto especificado. Se debe añadir el plugin al fichero pom.xml:

```
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-javadoc-plugin</artifactId>
            <version>3.6.2</version>
            <configuration>
                <source>17</source>
                <encoding>UTF-8</encoding>
            </configuration>
        </plugin>
    </plugins>
</build>
```

Si se utiliza Maven en línea de comandos, hay que lanzar la siguiente orden para generar Javadocs en el directorio target/site:

```
mvn javadoc:javadoc
```

Para ver los resultados se abre el fichero generado en target/site/apidocs/index.html

GhostDoc

GhostDoc es una extensión de Visual Studio que permite generar de forma automática documentación XML que contiene toda la información sobre los comentarios de un código determinado, junto con sus principales funciones.

Esta herramienta hace uso de la documentación correspondiente que le facilita Microsoft o el proveedor de la biblioteca utilizada. A la hora de crear la documentación en XML que se

encuentra dentro de las propiedades de un proyecto determinado, se siguen los siguientes pasos:

- En primer lugar, se accede a la opción de compilación (Build).
- A continuación, se selecciona la opción de archivo de documentación XML.
- Por último, se define el nombre junto con la ruta correspondiente.

SandCastle

SandCastle se utiliza también para crear documentación correspondiente a un documento determinado, pero, basándose en estilo MSD de .NET con los comentarios XML que estén asociados.

La utilización de esta herramienta es bastante sencilla y está basada en la línea de comandos. En su wiki se pueden ver los pasos básicos de uso:
https://github.com/EWSoftware/SHFB/wiki

Se compone de dos partes bien diferenciadas que trabajan juntas. Por un lado, la propia herramienta SandCastle y por otra el SandCastle Help File Builder.

La ayuda de referencia de una API se crea combinando los comentarios XML que están incrustados en el código fuente con la sintaxis y estructura de la ayuda propia del .NET Framework incluyendo referencias a esta si hiciera falta. La ayuda conceptual se crea mediante la conversión de documentos XML que contienen un dialecto conocido como Lenguaje de Marcado de Asistencia de Microsoft (MAML).

Sandcastle Help File Builder se creó para proporcionar las características que se usan con mayor frecuencia y proporcionar una GUI independiente y herramientas basadas en línea de comandos para crear un archivo de ayuda de forma automatizada. También está disponible un paquete de integración de Visual Studio para que los proyectos de ayuda se puedan crear y administrar completamente desde ese IDE.

HelpNDoc

Herramienta que permite generar documentos de ayuda en formatos como PDF, CHM o HTML entre otros.

Una vez que está descargada e instalada, los pasos que seguir son:
1. Se crea un nuevo proyecto.
2. A continuación, se seleccionan los datos correspondientes al nombre, idioma y tabla de contenidos.
3. Se realizan las configuraciones pertinentes de esta herramienta.

HelpNDoc es un producto comercial, aunque dispone de una versión gratuita sin limitaciones pero que incluye un banner en la parte inferior de cada página.

Microsoft Help Viewer

Microsoft Help Viewer es una herramienta desarrollada por Microsoft que se puede utilizar para crear archivos de ayuda en formato de Visor de Ayuda de Microsoft. Se encuentra integrada en Visual Studio.

DocFX

DocFX es una herramienta de código abierto desarrollada por Microsoft que permite generar documentación, incluidos archivos de ayuda, a partir de los archivos de código fuente. Es especialmente útil para proyectos basados en Markdown y es compatible con varias plataformas, entre ellas, Java.

Sphinx

Sphinx es una herramienta de generación de documentación de código abierto que admite la creación de archivos de ayuda. Aunque es más conocido en entornos de desarrollo de Python, es bastante versátil y puede utilizarse para proyectos en otros lenguajes.

4.3. AYUDA GENÉRICA Y SENSIBLE AL CONTEXTO

Toda aplicación informática, con independencia del tipo que sea (sistemas operativos, juegos, programas de gestión, programas de ofimática, etc.), viene acompañados de un sistema externo al propio programa, que son las ayudas de la aplicación. La ayuda de una aplicación tiene como objetivo orientar al usuario de la aplicación, con las funcionalidades y uso de la aplicación en la que van incrustada. El objetivo de la ayuda es orientar sobre la forma de usar la aplicación, las posibilidades que ofrece el programa, responder sobre preguntas comunes y toda información adicional que pueda ser útil.

Se puede considerar ayuda genérica la vista anteriormente, es decir, la que aparece en las opciones de menú de Ayuda donde se describe el funcionamiento de la aplicación, sea accediendo a esa opción o mediante la pulsación de la tecla F1. La ayuda genérica de una aplicación va a proporcionar un entorno en el que se le van a presentar todos los contenidos de ayuda que hay diseñados. Con la ayuda genérica se ofrecen todos los contenidos de ella, de forma que se pueda navegar por ellos y seleccionar el tema que más interese. Además de la lista de contenidos, la ayuda genérica suele incluir un buscador, de forma que se pueda buscar un "Término de ayuda" concreto. De este modo, el sistema de ayuda mostrará todas las referencias donde aparezca ese concepto. Elegido el término de ayuda o el contenido deseado, el sistema de ayuda mostrará el texto asociado, donde se podrá visualizar toda la información de ayuda que hay para el mismo.

Como complemento de la genérica, está la ayuda sensible al contexto o contextual, la cual ofrece al usuario, a través de diferentes elementos visuales o de texto, un acceso más rápido y directo a la función que desee realizar.

En este tipo de ayuda, se pueden encontrar los TootTips, los cuales son una serie de elementos que aparecen en la interfaz cuando el ratón se posiciona sobre algún elemento determinado. El objetivo de estos elementos es brindar al usuario alguna ayuda sobre ese elemento en cuestión.

Por supuesto, también se puede acceder a este tipo de ayuda mediante la pulsación del atajo de teclado F1 si la aplicación la soporta. De esta forma, al posicionarse sobre un elemento o seleccionarlo y proceder a la pulsación de F1, el sistema lanzará una ayuda personalizada enfocada en dicho elemento.

4.4. TABLAS DE CONTENIDO

Las tablas de contenidos, más comúnmente conocidas con TOC (Table of Content), es una de las partes que conforma el sistema de ayuda de una aplicación.

Con las tablas de contenidos se puede elaborar la estructura correspondiente de un determinado proyecto, y en las tablas se pueden identificar los títulos de los distintos temas o subtemas que intervienen en el proyecto. Asimismo, existe la opción de que la tabla de contenidos contenga el número de página o no. La tabla de contenidos suele presentar diferentes niveles de temas, de forma que se avanza desde un tema general a otro más específico. La profundidad en los niveles de contenido depende del tamaño de la ayuda y del nivel de detalle de su diseño.

Para facilitar el acceso a los términos de ayuda, la ayuda suele contar con un índice. El índice es una lista de palabras claves ordenadas alfabéticamente. Se puede decir que es similar a un glosario de términos o una guía de definiciones, salvo que el objetivo es aclarar un concepto o un procedimiento de uso de la aplicación en la que está insertada. Por ejemplo, si se realiza la ayuda del lenguaje de programación Java, las palabras reservadas del lenguaje serían palabras clave. Normalmente, para acceder al índice se puede navegar por la ventana de términos indexados y seleccionar el que interese aclarar, o bien, rellenar un cuadro de texto y que el sistema busque el término en toda la ayuda. También es posible tener acceso a un lugar concreto, indicando el número de página en la que se encuentra.

El sistema de búsquedas en la ayuda es una herramienta que permite buscar todas las referencias que aparecen en la ayuda, para un término en concreto. Para buscar un término concreto en el sistema de ayuda, se introduce la palabra a buscar en un cuadro de texto y se activa la búsqueda de esa palabra (bien palabra completa o en títulos). Si se pulsa el botón buscar, el sistema de ayuda busca todas las páginas de ayuda en la que aparece esa palabra. El usuario de la ayuda podrá navegar por los títulos de las páginas o por los temas en los que aparece la palabra que buscaba, y podrá ver el contenido de la ayuda para ese tema o título. En la página de ayuda aparecerán resaltadas todas las ocurrencias de la palabra buscada.

4.5. INCORPORACIÓN DE AYUDA A LA APLICACIÓN

La incorporación de ayuda a la aplicación, en cuanto a Java se requiere, se puede realizar de distintas formas. Hasta hace unos años, se utilizaban bibliotecas como JavaHelp pero Oracle ha decidido dejar de darla soporte. Ahora mismo, se puede utilizar cualquiera de las siguientes alternativas:

- HTML: Una de las soluciones más simples es crear un sistema de ayuda utilizando páginas HTML. De esta forma se puede diseñar y organizar el contenido de ayuda como se desee. Se puede lanzar HTML en un navegador web incluido en la aplicación Java utilizando JavaFX WebView o el componente JEditorPane de Swing.
- Plataformas de Documentación en línea: Se pueden considerar plataformas de documentación en línea, como Read the Docs, GitBook o similar. Crear una documentación en línea para la aplicación y enlazarla desde la aplicación Java es una solución eficaz. Se puede alojar la documentación en la web y acceder a ella desde la aplicación a través de un navegador o un visor de contenido web como los anteriormente citados.

Como caso práctico, se va a ver cómo implementar un sistema de ayuda online mediante el control JavaFX WebView. En este ejemplo, se verá el uso del control, pero desde una aplicación Swing. El contenido de la ayuda puede obtenerse, por ejemplo, de GitBook.

En primer lugar, para poder utilizar los controles JavaFX en un proyecto Swing, hay que incluir las dependencias a sus bibliotecas. En Maven, el código puede ser similar al siguiente teniendo en cuenta que las versiones de las dependencias aumentan con el paso del tiempo:

```
<dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-swing</artifactId>
    <version>17.0.7</version>
    <type>jar</type>
</dependency>
<dependency>
    <groupId>org.openjfx</groupId>
    <artifactId>javafx-web</artifactId>
    <version>17.0.7</version>
    <type>jar</type>
</dependency>
```

A continuación, se crearía la ayuda mediante GitBook. Se puede crear una estructura en la cual la página principal enlace a otras páginas. De esta forma, al acceder a la ayuda completa del sistema, irá al enlace principal mientras que, en el caso de ayuda contextual, se puede ir a otras páginas dependiendo del control en el que se utilice.

En el mismo constructor del Frame sobre el que se trabaje, se puede incluir un método llamado setHelp() que ajustará tanto la ayuda principal como la contextual. Antes de eso, se declararán dentro de la clase varios elementos que se necesitarán para que el sistema funcione:

```
JFXPanel jfxPanel;
JFrame frame;
```

Es decir, se necesita un elemento JFXPanel y un JFrame. El primero irá incluido en el segundo y este último será el que muestre la ayuda en cada momento que se invoque.

Para mostrar la ayuda principal, el método setHelp() tan solo requiere las siguientes líneas:

```
private void setHelp() {
    jfxPanel = new JFXPanel();
    frame = new JFrame("Ayuda");
    frame.setSize(new Dimension(500, 500));
    frame.add(jfxPanel);
}
```

Este método irá ubicado en el constructor de la clase justo después de initComponents(). Posteriormente se verá cómo modificarlo para poder añadir ayuda sensible al contexto. Este código simplemente instancia un JFXPanel y lo añade a un frame de un tamaño concreto (se puede ajustar a gusto del desarrollador).

Para lanzar la ayuda principal, lo normal es introducir una entrada de menú asociada al atajo de teclado F1. Dentro de su manejador de evento hay que crear la ventana que mostrará la ayuda. Como se utilizará más veces, se va a integrar en un método privado:

```
private void openWebView(String url) {
    Platform.runLater(() -> {
        WebView webView = new WebView();
        WebEngine webEngine = webView.getEngine();
        webEngine.load(url);
        jfxPanel.setScene(new Scene(webView));
        frame.setVisible(true);
    });
}
```

Ahora simplemente, se ubicará en el ActionPerformed del menú de ayuda:

```
private void mnuAyudaActionPerformed(java.awt.event.ActionEvent evt)
{
    openWebView("url-principal");
}
```

Esto simplemente crea dos objetos de tipo WebView y WebEngine, carga la URL y crea una escena dentro del panel con el webView. A continuación, se muestra el formulario.

Para poder asociar otros controles y tener ayuda contextual, hace falta modificar el código de setHelp de manera que los pueda contemplar. Se añaden las siguientes líneas:

```
Map<JComponent, String> contextualHelpMap = new HashMap<>();
contextualHelpMap.put(btnAccion1, "url-pagina1");
contextualHelpMap.put(btnAccion2, "url-pagina2");
// Se pueden agregar más elementos del mismo modo
setContextualHelp(contextualHelpMap);
```

Esto simplemente instancia un diccionario que asocia a cada control la URL de su ayuda personalizada. A continuación, se llama a un método que recogerá ese diccionario e irá asociando a cada componente su evento al pulsar F1 con la llamada al visor con la URL asociada a cada componente. Su código se muestra a continuación:

```
private void setContextualHelp(Map<JComponent, String> map) {
    for (JComponent comp : map.keySet()) {
        KeyStroke f1KeyStroke = KeyStroke.getKeyStroke(KeyEvent.VK_F1, 0);
        InputMap inputMap = comp.getInputMap(JComponent.WHEN_FOCUSED);
        ActionMap actionMap = comp.getActionMap();
        inputMap.put(f1KeyStroke, "showContextualHelp");
        actionMap.put("showContextualHelp", new AbstractAction() {
        @Override
            public void actionPerformed(ActionEvent e) {
                String helpURL;
                helpURL = map.get(comp);
                openWebView(helpURL);
            }
        });
    }
}
```

.

4.6. TIPOS DE MANUALES

Durante el desarrollo de un proyecto se generan ciertos tipos de documentación que se entregan al usuario de forma que tenga la información funcional, técnica y de operación sobre el producto desarrollado.

4.6.1. MANUAL DE INSTALACIÓN

Contiene todos los detalles necesarios para llevar a cabo la instalación de la aplicación en diferentes entornos. Se contemplan en este tipo de documentación aspectos referidos al software como pueden ser diferentes sistemas operativos, otras aplicaciones instaladas, dependencias de otros programas previamente instalados y que si no están presentes se deben instalar (manualmente o a través del propio instalador de la aplicación) y aspectos referidos a características hardware mínimas exigidas (memoria RAM, disco, tarjeta gráfica, etc.)

Se explican también distintas posibilidades de instalación como puede ser el cambio de directorios por defecto o la generación o no de, por ejemplo, archivos de registro, así como la ubicación de los archivos de configuración, tipo de instalación (completa, estándar, mínima) o diferentes aspectos que pueden modificar la apariencia (skins), los menús que se muestran o distintos ajustes de funcionamiento según las necesidades (seguridad, rapidez, control de procesos, mejor presentación, etc.)

De esta documentación suele haber dos versiones: una reducida para instalación rápida con valores por defecto, sin profundizar y otra detallada donde se explican ampliamente cada uno de los aspectos anteriormente nombrados.

4.6.2. DOCUMENTACIÓN DE CONFIGURACIÓN

Es una parte de la documentación dedicada a explicar todos los parámetros de configuración de la aplicación, así como el efecto que producen cada uno de los cambios en su funcionamiento.

En esta documentación se mezclan aspectos software, hardware, visuales, etc. Por ejemplo, en el IDE NetBeans, la configuración permite modificar la visualización del texto o sus elementos visuales, modificar colores, así como el tipo de letra. Asimismo, permite definir cómo se ejecutarán las aplicaciones o las bibliotecas por defecto que se usarán o los directorios de destino del resultado de los proyectos. En otras aplicaciones, por ejemplo, como máquinas virtuales, se permite definir las características hardware (disco, memoria RAM, etc. de la máquina que se quiere simular.

En el documento referido a la configuración se debe especificar cómo realizar esos cambios en la configuración de la aplicación, así como qué consecuencias tendrán dichos cambios en su funcionamiento y presentación.

Es habitual que se encuentren varios niveles de configuración. Lo más habitual son dos niveles: configuración normal y avanzada. Dependiendo de lo compleja que sea la aplicación puede ocurrir que, además de la guía de configuración normal se genere una para la avanzada en la que se detallen los aspectos y labores más complejos de la administración de la aplicación y que requieren de mayor experiencia y conocimiento por parte de los usuarios.

4.6.3. MANUAL DE USUARIO

En principio, este tipo de manual se puede desarrollar en dos versiones: rápida y detallada. La diferencia entre ambas es el nivel de detalle con el que se explican las posibilidades de la aplicación. En muchos casos se describen las funciones que puede realizar el usuario final desde el punto de vista estrictamente práctico y funcional sin preocuparse de aspectos técnicos (que se suponen salvados o tenidos en cuenta por los administradores para lo que se han ayudado de las guías de instalación y funcionamiento).

En muchos casos se hace uso de esta documentación en la interfaz de usuario de la aplicación para complementarla. Se explican al usuario los pasos que debe seguir para realizar su trabajo a través de un interfaz propio con lo que el usuario final será capaz de identificar esos elementos en la aplicación.

En este tipo de documentación hay que tener en cuenta que el nivel de los usuarios que usan la aplicación puede no ser homogéneo y en algunos casos será necesario realizar varias versiones: principiante, inicial, medio, avanzado y experto. De esta forma, en cada caso se profundiza más en las explicaciones. Así, para niveles iniciales se explicará el funcionamiento de la aplicación, pero desde el punto de vista puramente funcional (describiendo cada acción y los valores que se deben proporcionar o elegir para llevarlas a cabo) y además se elegirán acciones fáciles de realizar. Según se va subiendo de nivel, experiencia y capacidad se describen acciones más complejas que requieren una mayor y más amplia experiencia.

Una estructura frecuente de los manuales de usuario incluye una introducción al producto en cuestión, un índice con los contenidos del manual, la guía en sí misma, una sección de problemas frecuentes y su forma de solucionarlo, los datos de contacto y un glosario. Un ejemplo de las secciones de este documento se muestra a continuación:

- Portada
- Página de título y de copyright.
- Prólogo, el cual muestra detalles de los documentos relacionados e información sobre como navegar por el manual de usuario.
- Páginas de contenidos, donde se muestra un índice con los contenidos de manual.
- Una guía de cómo usar, al menos, las funciones básicas de la aplicación.
- Una sección de solución de problemas, donde se detallan problemas y posibles soluciones que pueden surgir, junto con la forma de solucionarlos.
- Una sección FAQ (Frequently Asked Questions).

- Donde encontrar más ayuda para uso de la aplicación.
- Un glosario y un índice.

4.6.4. GUÍA DE REFERENCIA

Esta guía tiene un objetivo similar al del manual de usuario diferenciándose la extensión y la cantidad de información facilitada. Tienen como objetivo indicar todas las funcionalidades del producto de forma que al cliente le pueda servir de referencia.

Un ejemplo típico de manual de referencia sería la guía de referencia para la utilización de un entorno de desarrollo, donde es posible programar en diferentes lenguajes de programación. Para que el usuario pueda utilizar esta aplicación, el manual de usuario le va a indicar los pasos necesarios para crear un nuevo proyecto, como navegar por las diferentes ventanas de la aplicación, las distintas opciones que presenta el menú, junto con su utilidad. Una guía de referencia, en este caso, sería el manual donde se detalla la sintaxis, palabras reservadas, tipos de sentencias, etc., específicos del lenguaje de programación con el que se vaya a trabajar.

4.6.5. GUÍA RÁPIDA

Documentación complementaria a la de usuario o administrador que incluye referencia a las guías comentadas. En su contenido se detalla de forma breve y sintetizada la forma de realizar una tarea.

Cuando un desarrollador comienza un nuevo proyecto es necesario evaluar las ventajas que puede aportar, la implementación de una guía rápida. Sería apropiada en los siguientes casos:

- Aplicación que requiere una sola configuración: si solo tiene que configurarse una vez durante el proceso de instalación, la aplicación está operativa de forma rápida. La guía rápida permite al usuario o usuario final los conocimientos básicos para utilizar la aplicación.
- Aplicación con funcionalidad limitada: si tiene un número muy limitado de tareas no será necesario una documentación muy extensa.
- Documentación muy extensa: si una aplicación tiene un gran volumen de documentación porque es una aplicación muy compleja, una guía rápida permitirá la ejecución de las tareas principales, de forma que cuando un usuario necesite ejecutar otras más complejas, puede consultar el manual o la ayuda, pero ya está capacitado para ejecutar las tareas básicas.
- Usuarios que no puedan dedicar mucho tiempo a la formación, en aplicaciones que migran a una versión nueva, etc.

Para desarrollar la guía rápida, hay que decidir el contenido que va a incluir. Si la guía rápida forma parte de una documentación más amplia, primero se desarrolla esta. Teniendo una visión global del producto, es posible decidir los aspectos más importantes que puedan permitir a un usuario utilizar la aplicación. Las tareas esenciales serán las que se incluyan en la guía rápida.

Existen muchas plantillas para el desarrollo de guías rápidas, sin embargo, cada aplicación va a requerir un diseño específico. Para el diseño gráfico de la guía de referencia rápida, se deben tener presentes estos cuatro aspectos: contraste, repetición, alineación y proximidad:

- El contraste permite establecer relaciones jerárquicas entre la información, por ejemplo, usar contraste entre los títulos y el texto ayuda al usuario a entender mejor la estructura del documento. También sirve para llamar la atención.
- La repetición de un diseño consistente en todo el documento da integridad a la totalidad de este. Por ejemplo, la repetición de un logotipo, de un determinado color, etc.
- Es necesaria la alineación del texto, ya que mantener la simetría entre los párrafos, imágenes, títulos y otros elementos, permite una vista más relajada
- La proximidad se refiere a la agrupación de objetos interrelacionados. En una guía rápida se deben agrupar las tareas importantes en la columna principal, manteniendo las tareas subordinadas en la columna lateral.

4.6.6. MANUAL DE ADMINISTRACIÓN

Permite generar la guía para conocer los detalles de la instalación, de la administración del sistema y de la configuración de los distintos componentes de la aplicación o sistema. Este tipo de manuales se utiliza por los administradores TI ya que disponen de los máximos privilegios a nivel operativo en esa materia dentro de la organización.

4.7. CONFECCIÓN DE TUTORIALES MULTIMEDIA

Un tutorial multimedia incorpora un conjunto de recursos de diferente tipo para ofrecer información visualmente atractiva, sobre un determinado tema, software, herramienta, etc. Cualquier tutorial multimedia que quiera tener calidad, necesita de una buena información base, teniendo siempre presente el tipo de usuario o usuaria a la que va destinado. La información se va a complementar con imágenes, vídeos, sonidos, animaciones, gráficos, enlaces externos o cualquier otro recurso que permita completar y mejorar la calidad del tutorial que se está implementando.

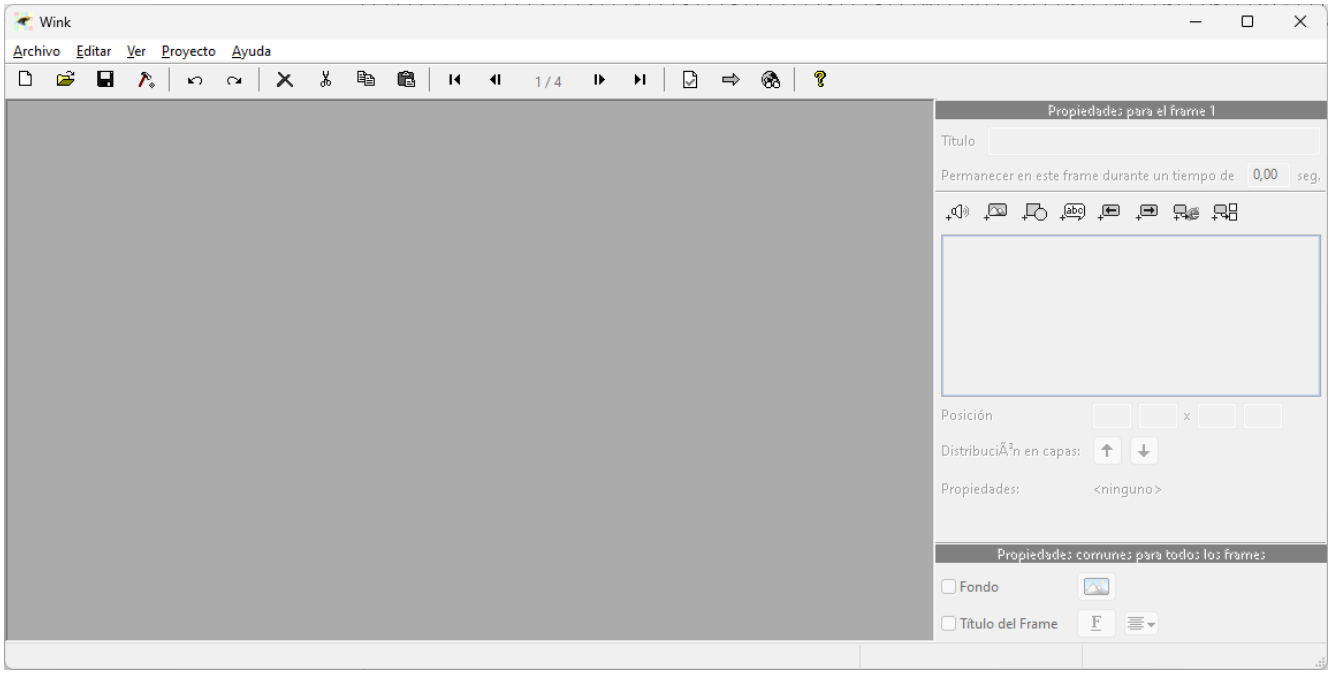
Las capturas de pantalla permiten crear una imagen del escritorio del ordenador, una determinada ventana o un área específica. Esta imagen puede ser manipulada e incrustada en un documento, permitiendo la creación de manuales multimedia a medida. Además de las incluidas en el sistema operativo, existen multitud de herramientas destinadas a la creación de vídeos o capturas de pantalla que pueden ser utilizadas para crear un tutorial. Ejemplos de estas pueden ser Snagit, Hypersnap DX y WinSnap, entre otras. Las herramientas Snagit e Hypersnap Dx son capaces de capturar aplicaciones DirectX además de capturar texto, zonas seleccionadas a mano y varias ventanas/controles de una vez. Snagit añade a eso otras como la captura de vídeo, la adquisición desde cámaras y escáner, de imágenes de páginas web o recursos (iconos, cursores, gráficos) de programas, entre otros. Otra de las funciones más importantes consiste en la captura de ventanas con desplazamiento, el único que no dispone de este tipo de captura es WinSnap, si bien en el resto se produce con distinta eficacia. Otra herramienta para capturar lo que está sucediendo en la pantalla y que también permite audio es OBS Studio. De código abierto, es ideal para crear videotutoriales.

Un ejemplo de este tipo de herramientas es Wink. Esta herramienta permite incluir anotaciones, pausar el tutorial, añadir enlaces a páginas web o a otros apartados del tutorial, etc. Uno de sus puntos fuerte radica en su disponibilidad para diferentes plataformas. Concretamente se encuentra disponible tanto para Windows como para Linux. Se trata de una muy buena solución gratuita para quienes necesiten crear un tutorial sobre el funcionamiento de una aplicación o sobre el trabajo con un software ofimático, por citar algunos ejemplos.

En Wink, para crear un tutorial se pueden capturar imágenes o importarlas, grabar las explicaciones en audio, añadir cuadros de texto, formas, flechas, enlaces, etc. Al terminar la creación del tutorial, se puede exportar como archivo HTML, PDF o PostScript.

La herramienta permite varias formas de captura como son:

- Captura de imágenes
- Captura continua. En este caso es posible configurar el número de frames por segundo lo que redundará directamente en el tamaño del archivo.
- Captura selectiva. Se capturan los eventos relevantes para el tutorial activados al pulsar determinadas teclas o acciones del ratón.



6. Pantalla principal de Wink

Por otra parte, existen soluciones en la nube como H5P para crear tutoriales interactivos. Permite, entre otras muchas funcionalidades, incluir texto, tablas, enlaces, imágenes e incluso preguntas de tipo test dentro de vídeos alojados en plataformas como Youtube.

ÍNDICE DE FIGURAS

1. Documentación HTML de clase Java 8
2. Documentación HTML de método Java 9
3. Documentación HTML de referencias 9
4. Código HTML incrustado en la ayuda 10
5. Inclusión de código en la ayuda 11
6. Pantalla principal de Wink 24

BIBLIOGRAFÍA – WEBGRAFÍA

Manera, J. *7 Best Practices for Java API Documentation*
https://manerajona.medium.com/7-best-practices-for-java-api-documentation-dc6e7e87d33f

Vicente Carro, J.L. (2014). *Desarrollo de interfaces.* Editorial Garceta.

Ferrer Martínez, J. (2015). *Desarrollo de interfaces*. Editorial Ra-Ma