

**MP0488.**

**Desarrollo de interfaces de usuario  
UF6. Testing, confección de informes y  
documentación de aplicaciones**

**6.3. Tipos de informes,  
estructura y herramientas**

# Índice

---

☰	Objetivos	3
☰	Informes incrustados y no incrustados	4
☰	Estructura general de un informe	6
☰	Herramientas gráficas integradas y externas al IDE	8
☰	Librerías para generación de informes: clases, métodos y atributos	9
☰	Integrar JavaHelp	11
☰	Crear informes y gráficos con IReport	12
☰	Resumen	13

# Objetivos

---

---

En esta lección perseguimos los siguientes objetivos:

1

Distinguir los tipos de informes existentes.

2

Aprender la estructura básica de los informes, independientemente del lenguaje de programación o aplicación (IDE) que se esté utilizando.

---

¡Ánimo y adelante!

# Informes incrustados y no incrustados

Vamos a conocer las posibilidades que se nos plantean a la hora de incluir informes en nuestras aplicaciones.

Los informes de una aplicación se pueden dividir en dos grandes grupos, desde el punto de vista de su diseño e inclusión en la misma:

## INFORME INCRUSTADO

Es un informe que se ha importado a un proyecto, o que se ha creado en el mismo, que genera una clase contenedora que contiene el propio informe. En el caso de que sea una aplicación compilada, el informe y su clase contenedora se incrustan en el ensamblado, al igual que cualquier otro recurso, módulo, paquete o clase del proyecto. Tenemos el ejemplo de Microsoft Visual Studio y su clase contenedora, denominada *ReportDocument*. En JAVA disponemos de la clase *JasperReports*... Y un largo etcétera.

## INFORME NO INCRUSTADO

Es un informe externo al proyecto, aunque se habilite la interacción del programa con el propio informe, y se accede al mismo mediante una ruta de directorio de archivos que se obtiene mediante formatos HTML/XML e incluso, si forma parte de un grupo de informes expuestos a través de herramientas externas. El ejemplo lo tenemos con *Crystal Reports*, el generador de informes de mayor influencia, que puede trabajar y generar informes no incrustados sin añadir ninguna clase contenedora, cargando el informe en tiempo de ejecución.

Elegiremos un informe **incrustado**....

- 1 Si lo que necesitamos es que la implementación del proyecto sea simplificada al máximo, utilizaremos la opción de informes incrustados. Pero debemos tener en cuenta que no todos los lenguajes y sus correspondientes entornos de desarrollo soportan la creación de informes, o tienen la herramienta o asistente para llevar a cabo su diseño.
- 2 No necesitamos archivos ni rutas para trabajar, ya que así se evitan problemas de acceso. Además, se limita el tamaño de los archivos.
- 3 Siempre y cuando los informes no se expongan a modificaciones tras el compilado del proyecto.

```
Private Sub Button1_Click(ByVal sender As System.Object, _  
    ByVal e As System.EventArgs) Handles Button1.Click  
    Dim warnings As Warning() = Nothing  
    Dim streamids As String() = Nothing  
    Dim mimeType As String = Nothing  
    Dim encoding As String = Nothing  
    Dim extension As String = Nothing  
    Dim deviceInfo as string  
    Dim bytes As Byte()  
  
    deviceInfo = "<DeviceInfo><SimplePageHeaders>True</SimplePageHeaders></DeviceInfo>"  
  
    bytes = ReportViewer1.LocalReport.Render("Excel", Nothing, _  
        mimeType, encoding, extension, streamids, warnings)  
  
    Dim fs As New FileStream("c:\informes.xls", FileMode.Create)  
    fs.Write(bytes, 0, bytes.Length)  
  
    MessageBox.Show("Reporte exportado a informes.xls", "Info")  
End Sub
```

Ejemplo de código: uso de un objeto *LocalReport* para cargar y exportar un informe en C#.

### ¿Podemos elegir informes **no incrustados**?

1

Sabiendo que son más sencillos y seguros, pero requieren más trabajo.

2

Si podemos modificar su diseño sin necesidad de volver a compilar el proyecto, lo que nos facilita el acceso y la modificación de los mismos.

3

Ya que ofrecen muchas posibilidades y ventajas frente a la escalabilidad.

```
Private Sub Cargar_Informe(ByVal ValorParametro As String, ByVal RutaRPT As String)  
    Dim Rpt As New CrystalDecisions.CrystalReports.Engine.ReportDocument  
    Rpt.Load(RutaRPT)  
    Rpt.SetParameterValue("Informe", ValorInforme)  
    CrystalReportViewer1.ReportSource = Rpt  
    CrystalReportViewer1.Show()  
    Application.DoEvents()  
End Sub
```

Formulario con el que controlamos el *CrystalReportViewer*.

# Estructura general de un informe

## ¿Qué secciones *debería* contener un informe de una aplicación?

En un entorno en el que es cada día más complicado conocer las necesidades de los usuarios en las aplicaciones y sites, los informes que reportan se convierten en un aliado para alcanzar conclusiones sobre el software y sus resultados, con los que podemos analizar los datos obtenidos y saber cómo mejorar la aplicación.

En general, todos los informes siguen una misma estructura y, aunque el contenido sea distinto, continúan las mismas líneas a la hora de estructurarlo, dividiendo el documento en tres partes:

1

**Introducción:** en la que justificaremos la elaboración del documento en cuestión.

2

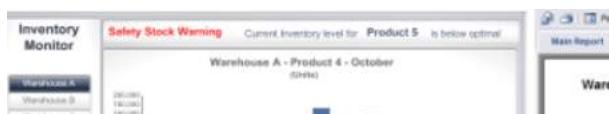
**Desarrollo:** en el que se pueden detallar los procedimientos y la metodología empleada para recopilar la información, los parámetros que se han seguido, etcétera.

3

**Conclusión:** donde se presentan los resultados obtenidos y las valoraciones de los mismos.

## Report Header

Se imprime una sola vez al inicio del informe. Esta sección contiene el logo de la empresa, fechas de realización de prueba y elaboración del informe, nombre del informe...



Ejemplo de un informe con su *header* propio.

## Page Header

Se imprime al inicio de cada página impresa y puede contener anotaciones generales, nombres, direcciones...

**REGISTRO DE LA SESIÓN**  
**ASIGNATURA: |**  
**PROFESOR/A: |**

En algunos casos, también se permite el uso de campos que los usuarios deberán llenar con el nombre, asignaturas/trabajos realizados, fechas de registros de sesiones...

## Details

Multitud de registros conforman el reporte o informe en sí, ya que realmente se trata de la parte en la que se alojan los campos del origen de datos.

Los datos del informe varían según las secciones en las que se desee insertar objetos de informes concretos. Además, se separan los datos en grupos del mismo tipo para facilitar la lectura y el análisis.

GTPC	DHCP	DELL	WINDOWS 7	1T75L
<b>GTPC</b>	<b>192.168.1</b>	255.555.555.0	<b>192.168.1.1</b>	<b>DELL</b>
<b>GTPC</b>	<b>192.168.1.18</b>	255.555.555.0	<b>192.168.1.1</b>	<b>DELL</b>
<b>GTPC</b>	<b>192.168.1.220</b>	255.555.555.0	<b>192.168.1.1</b>	<b>DELL</b>
<b>GTPC</b>	<b>192.168.1.242</b>	255.555.555.0	<b>192.168.1.1</b>	<b>DELL</b>

El ejemplo más claro puede ser un inventario, por ejemplo.

---

## Report footer

Es el área donde se imprime una sola vez al finalizar el informe.

Rev.:2  
Fecha: 07/11/16

También se utiliza para insertar las fechas de las últimas revisiones del informe.

---

## Page footer

Se imprime al final de cada página y da cabida a la paginación, totales por pagina, etc.

---

Página 3

Típico final de página, donde mostramos la página en la que estamos del total del informe.

---



Esta estructura no es obligatoria para todos los informes, aunque es muy aconsejable que se mantenga por la sencillez y posibilidad de cambios que permite.

Por otra parte, se pueden definir **filtros** para seleccionar o excluir partes de un conjunto de datos. Los filtros limitan los datos que se muestran al usuario.

# Herramientas gráficas integradas y externas al IDE

Vamos a conocer las herramientas con las que podemos generar informes en diferentes formatos.

Las diferentes herramientas que están contenidas en suites ofimáticas, como las de Open Source Openoffice.org, LibreOffice y la suite propietaria Office de Microsoft, también permiten la obtención de informes o reportes en diferentes formatos, principalmente PDF y XPS.

Veamos otras herramientas con las que podemos generar informes:

1

*Crystal Reports*: aplicación utilizada para diseñar y generar informes desde una fuente de datos. Aplicaciones como *Microsoft Visual Studio* incluyen una versión OEM de Crystal Reports, como herramienta de propósito general para informes/reportes que nos permite incluir datos de varias tablas, sentencias SQL y estructuras condicionales e iterativas, con un formato de salida en archivos .PDF, .RPT, .XLS, .TXT y .CSV, entre otros.

<http://www.crystalreports.com/>

2

*JasperReports*: herramienta de creación de informes que tiene la habilidad de entregar contenido enriquecido al monitor, a la impresora o a ficheros PDF, HTML, XLS, CSV y XML, y es utilizada de forma común con *iReport*, *front-end* gráfico de código abierto para la edición de informes. Está desarrollado en Java y puede ser usado en gran variedad de aplicaciones Java, como J2EE o aplicaciones web para generar contenido dinámico.

<http://community.jaspersoft.com/project/ireport-designer>

3

*Eclipse Birt*: es un sistema para aplicaciones web Java o Java EE. Está formado por dos componentes principales: un diseñador de informes basado en Eclipse, y otro que se puede integrar en el servidor de aplicaciones y que permite generar informes en tiempo de ejecución.

<http://www.eclipse.org/birt/phoenix/>

# Librerías para generación de informes: clases, métodos y atributos

¿Desconoces cómo trabajar con algún tipo de software?

No te preocupes, porque disponemos de un manual.

## Recuentos, totales, subtotales...

Cuando se calculan los resúmenes y se totalizan los datos, tenemos que dividir en grupos y realizar la operación especificada sobre los valores de cada grupo, precisando:

- Campo que deseamos resumir o totalizar.
- Tipo de operación que se va a realizar con ese campo:
  - Media.
  - Unidades x precio-unidad.
  - Subtotal.
  - IVA x subtotal.
  - IVA + subtotal.
- Campo que va a desencadenar un nuevo grupo siempre que cambie su valor.
- Orden de clasificación.

Para insertar subtotales -sólo en campos numéricos- o totales, disponemos de determinadas herramientas que tienen asistentes para facilitar esos cálculos.

## Librería de generación de informes

Generalmente, la mayoría de las API para creación de informes utilizan herramientas integradas para los IDE, aunque lo más aconsejable es reducir el uso de SQL junto con la programación para generación de informes.

En el mercado existen muchas herramientas que integran todas las librerías propietarias necesarias para generar informes:

- *Microsoft SQL Server Reporting Services*.
- *Crystal Reports*.
- *BEA Portal Reporting Solutions*.
- *MicroStrategy Report Services*.
- *Actuate BIRT (Business Intelligence and Reporting Tools)*.

Y otras de Open Source:

- *Pentaho Report Designer.*
- *OpenRPT para PostgreSQL.*
- *Eclipse BIRT.*
- *JasperReports.*

La mayoría de las librerías que realizan la generación de informes establecen la siguiente arquitectura en su elaboración:

- **Modelo de datos.**
- **Acceso a los datos.**
- **Presentación de los datos.**

### Conexión con las fuentes de datos y ejecución de consultas

Las conexiones con las fuentes de datos son las mismas con las que se realizan las consultas, y su principal diferencia radica en la automatización de las tareas de esa generación.

Por ejemplo, mientras en VB.NET tiene un asistente para guiar al programador en la conexión de datos, Python tiene la conexión manual.

Todos los lenguajes de programación permiten ambas formas de conectar la aplicación, la base o fuente de datos para la elaboración de consultas e informes detallados, respectivamente. En el caso de la automatización, los generadores de informes trabajan de manera similar.

# Integrar JavaHelp

---

Vídeos del canal "Desarrollo de Interfaces"

<https://www.youtube.com/channel/UCkbeWlOiGLoQxBjZKdaZrrw/videos>

- En [este video](#) puedes ver un ejemplo de cómo crear un sistema de ayuda con JavaHelp.
- Y en este otro [video](#), puedes ver cómo integrar JavaHelp en una aplicación Swing.

# Crear informes y gráficos con IReport

A continuación, te ofrecemos una serie de vídeos de youtube, donde podrás ver los pasos de la creación de informes con IReport.

Videos del canal "Desarrollo de Interfaces"

<https://www.youtube.com/channel/UCkbeWlOiGLoQxBjZKdaZrrw/videos>

- 1 <https://youtu.be/bZJQIoIji-Q>
- 2 <https://youtu.be/XUrCaoI67Ho>
- 3 <https://youtu.be/mo8akXJIijY>
- 4 <https://youtu.be/EvsjMAvA-tI>
- 5 <https://youtu.be/LUSb-zlPcUc>
- 6 <https://youtu.be/hOsKiac2QZc>
- 7 <https://youtu.be/FKDTjhkXto>

# Resumen

---

Has terminado la lección, repasemos los puntos más importantes que hemos tratado.

- Los informes de una aplicación se pueden dividir en **informes incrustados y no incrustados**. Los informes **incrustados** se han importado a un proyecto o se han creado en él, generando una clase contenedora que contiene el propio informe. Los segundos, los **Informes no incrustados**, son externos al proyecto, aunque se habilita la interacción del programa con los propios informes y se accede a ellos mediante una ruta de directorio de archivos, obteniéndolos mediante formatos HTML/XML, e incluso si forman parte de un grupo de informes expuestos a través de herramientas externas.
- En general, **todos los informes siguen una misma estructura** y, aunque el contenido sea distinto, continúan las mismas líneas a la hora de estructurarlo, dividiendo el documento en tres partes:
  1. *Introducción*: en la que justificaremos la elaboración del documento en cuestión
  2. *Desarrollo*: en el cual se pueden detallar los procedimientos y la metodología empleada para recopilar la información, los parámetros que se han seguido, etcétera.
  3. *Conclusión*: donde se presentan los resultados obtenidos y las valoraciones de los mismos.
- Por otra parte, se pueden definir **filtros** para seleccionar o excluir partes de un conjunto de datos. Los filtros limitan los datos que se muestran al usuario.



**PROEDUCA**