4. Componentes imagen

wxImage

Este componente encapsula una imagen en una plataforma independiente. Una imagen puede ser creada desde datos, o usando el método *ConvertToImage()*, del componente wxBitmap, que veremos más adelante. Una imagen puede ser cargada desde un fichero en variedad de formatos.

Al existir más de una forma de crear una imagen, veremos más de un constructor, el primero es a partir de un wxBitmap:

```
wxImage(const wxBitmap& bitmap)
```

El segundo, a partir de un archivo, donde además de este, indicaremos el tipo y el índice (el índice servirá para ordenas todas las imágenes que tengamos):

```
wxImage(const wxString& name, long type = wxBITMAP\_TYPE\_ANY, int index = -1)
```

Y por último, mostraremos como crear una imagen nueva, a partir de su anchura y su altura, indicando además si la queremos en negra (true) o sin color (false):

```
wxImage(int width, int height, bool clear=true)
```

Existen manejadores predeterminados para las imágenes, como por ejemplo: **wxBMPHandler**, **wxPNGHandler**, **wxJPEGHandler**, **wxGIFHandler** o **wxTIFFHandler**, entre otros.

Además, este componente cuenta con métodos como: *ConvertToMono()*, para convertir una imagen en monocromática, *Copy()* y *Paste()*, para copiar una imagen o pegarla en un lugar determinado, *Scale()* y *Rescale()*, para escalarla, *Rotate()*, para rotarla, *LoadFile()* y *SaveFile()*, para trabajar con ficheros, *GetWidth()* y *GetHeight()*, que devuelven la anchura y altura de la imagen, y *Size()* y *Resize()*, que trabajan con el tamaño de la imagne.

wxBitmap

Esta clase encapsula el concepto de plataforma dependiente de mapa de bits, ya sea monocromo, en color o soportado por algún canal.

Como en el caso anterior, también existen varios constructores. El primero, es a partir de datos dados, siempre que se le indique también, el tamaño (anchura y altura), el tipo y la profundidad:

```
wxBitmap(void* data, int type, int width, int height, int depth = -1)
```

También podemos crear un nuevo Bitmap, con el siguiente constructor, pasandole la anchura, altura y profundidad:

```
wxBitmap(int width, int height, int depth = -1)
```

Y por último, podemos crearlo a partir de una imagen

```
wxBitmap(const wxImage& img, int depth = -1)
```

Existen diferentes tipos predeterminados, que podemos pasara como parámetro al constructor, según el tipo de imagen que queramos crear, como por ejemplo: **wxBITMAP_TYPE_BMP**, **wxBITMAP_TYPE_JPEG**, **wxBITMAP_TYPE_TIF**, **wxBITMAP_TYPE_PNG**, **wxBITMAP_TYPE_GIF**, **wxBITMAP_TYPE_XPM**, etc.

Los métodos que destacamos de este componente son los siguientes: *ConvertToImage()*, que convierte un wxBitmap en un wxImage, *CopyFromIcon()*, que crea un wxBitmap desde un wxIcon, *GetWidth()* y *GetHeight()*, que devuelven el tamaño del componente, *GetDepth()*, que devuelve la profundidad, y *LoadFile()* y *SaveFile()*, que trabajan con archivos.

wxStaticBitmap

Este componente muestra un wxBitmap. Podemos utilizarlo para mostrar iconos en cuadros de diálogos, pero no como componente para mostrar imágenes de forma común, ya que bajo Windows el tamaño de los wxBitmaps está limitado a 64x64 pixeles.

Para crearlo, debemos de pasar al constructor, la ventana padre, el identificador, la etiqueta, la posición, el tamaño y el estilo.

```
wxStaticBitmap(wxWindow* parent, wxWindowID id, const wxBitmap& label,
               const wxPoint& pos = wxDefaultPosition,
               const wxSize& size = wxDefaultSize, long style = 0)
```

Destacamos pocos métodos para este componente, principalmente *GetBitmap()*, *SetBitmap()*, *GetIcon()* y *SetIcon()*, que permiten trabajar tanto con objetos wxBitmaps como con wxIcon.

wxBitmapButton

Este componente es un botón simple que contiene un bitmap, o lo que sería lo mismo, un wxButton con un wxBitmap como etiqueta. Permite insertar diferentes wxBitmap para los diferentes estados posibles del botón, estos son: normal, desactivado, seleccionado, con el foco y con el ratón encima (sin pulsar).

Para crearlo, necesitamos pasar al constructor, los parámetros habituales, la ventana, identificador, posición, tamaño y estilo, y además el wxBitmap que deseemos.

```
wxBitmapButton(wxWindow* parent, wxWindowID id, const wxBitmap& bitmap,
               const wxPoint& pos = wxDefaultPosition,
               const wxSize& size = wxDefaultSize, long style = wxBU_AUTODRAW)
```

Los estilos que acepta este componente son: **wxBU_AUTODRAW**, que crea un botón con un estilo por defecto, y **wxBU_LEFT**, **wxBU_RIGHT**, **wxBU_TOP** y **wxBU_BOTTOM**, para seleccionar la alineación del wxBitmap en el botón.

Al ser un botón, su único evento es **EVT_BUTTON**, como el de estos.

Los métodos son para ver y modificar los wxBitmap de cada uno de los posibles estados del botón: *GetBitmapDisabled()*, *GetBitmapFocus()*, *GetBitmapHover()*, *GetBitmapLabel()*, *GetBitmapSelected()*, *SetBitmapDisabled()*, *SetBitmapFocus()*, *SetBitmapHover()*, *SetBitmapLabel()* y *SetBitmapSelected()*.