# Chapter 5—EXTBMPref & EXTCURref

## Introdcution

The EXTBMPref class gives your external components access to the OMNISPIC.DF1 and USERPIC.DF1 data files handled by Omnis. These data files normally reside in the ICONS subdirectory of your Omnis installation. All icons in Omnis are referenced by an icon identifier, or $iconid, which can be modified in the Omnis icon editor.

With the exception of custom cursors and full page bitmaps, each icon in Omnis can have three drawing sizes, 16x16, 32x32 and 48x48. Each icon has a pre-set default size that it uses unless another size is specified. This default size can also be modified using the Omnis icon editor. Some icons also have drawing modes. For example, checkbox icons have various modes, normal, checked, highlighted etc.

## Enumerations

### ePicModes (EXTBMPref only)

An enum defining the drawing modes supported by the icon-drawing function in this class.

* picNormal  
  The icon is drawn in its normal state.
* picChecked  
  The icon is drawn in its checked state.
* picHilited  
  The icon is drawn in its hilited state.
* picCheckedHilited  
  The icon is drawn in its checked and hilited state.

### ePicSize (EXTBMPref only)

An enum defining the drawing size supported by the icon drawing function in this class.

* ePicDef  
  The size of the icon depends on the default size set in the Omnis icon editor.
* ePic16  
  The 16x16 version of the icon is drawn.
* ePic32  
  The 32x32 version of the icon is drawn.
* ePic48  
  The 48x48 version of the icon is drawn.

## EXTBMPref Class Reference

### EXTBMPref::EXTBMPref()

|  |
| --- |
| EXTBMPref::EXTBMPref( qlong pIconID, qlong pDefault = 0, qapp pApp = 0 ) |

The constructor for the external bitmap class. After construction, the class can be used to interrogate the icon or draw the icon. When you have finished manipulating the icon, the class should be destructed.

1. **pIconID** - the icon associated with this class.
2. **pDefault** - the default icon id is used when pIconID is zero.
3. **pApp** - this parameter must be specified for web client components. See ECOgetApp.

### EXTBMPref::~EXTBMPref()

|  |
| --- |
| EXTBMPref::~EXTBMPref() |

The destructor for the external bitmap class. The destruction of the class informs Omnis that you have finished with the icon.

### EXTBMPref::addBmpSize()

|  |
| --- |
| qlong EXTBMPref::addBmpSize( qlong pIconID, ePicSize pSize ) |

Returns a new icon id with the specified pSize added.

1. **pIconId** - The icon id to add a size to.
2. **pSize** - The size to be added to the icon id.
3. **return** - A new icon id with the icon size pSize embedded.

**Note:** This is a static member function.

### EXTBMPref::copyImage()

|  |
| --- |
| HBITMAP EXTBMPref::copyImage( ePicSize pSize = ePicDef )  HBITMAP EXTBMPref::copyImage( qcol pFillColor, ePicSize pSize = ePicDef ) |

Returns a bitmap for the icon this class refers.

1. **pFillColor** - When calling copyImage specifying a fill color, the transparent pixels of the image are replaced with the given color.
2. **pSize** - The icon size to return.
3. **return** - Returns a new HBITMAP object if successful.

**Note:** The returned HBITMAP must be destroyed with GDIdeleteBitmap.

### EXTBMPref::copyImage()

|  |
| --- |
| HBITMAP EXTBMPref::copyImage( qcol pFillColor, ePicSize pSize = ePicDef ) |

Returns a bitmap for the icon this class references. The transparent color of the bitmap is replaced with the given color.

1. **pFillColor** - The replacement color for the transparent color of the bitmap.
2. **pSize** - The icon size to return.
3. **return** - Returns a new HBITMAP object if successful.

**Note:** The returned HBITMAP must be destroyed with GDIdeleteBitmap.

### EXTBMPref::copyMask()

|  |
| --- |
| HBITMAPMASK EXTBMPref::copyMask( ePicSize pSize = ePicDef ) |

Returns a bitmap mask for the icon this class refers.

1. **pSize** - The icon size to return a mask for.
2. **return** - Returns a new HBITMAPMASK object if successful.

**Note:** The returned HBITMAPMASK must be destroyed with GDIdeleteBitmap.

### EXTBMPref::draw()

|  |
| --- |
| void EXTBMPref::draw( HDC pHdc, qrect\* pRect, ePicSize pSize = ePicDef, ePicModes pWhich = picNormal, qbool pDisabled = qfalse, qcol pHilited = colNone, qbool pScale = qfalse, qjst pJstHoriz = jstLeft, qjst pJstVert = jstLeft ) |

Draws the icon’s image into a device context.

1. **pHdc** - The drawing device to draw into.
2. **pRect** - The destination drawing rectangle.
3. **pSize** - The icon size to draw.
4. **pWhich** - The icon drawing mode to use.
5. **pDisabled** - If qtrue the image is drawn in a disabled state.
6. **pHilited** - Controls how the icon is highlighted.
7. **pScale** - If qtrue the icon is scaled to the full size of pRect.
8. **pJstHoriz** - The horizontal drawing justification. This is ignored if pScale is qtrue.
9. **pJstVert** - The horizontal drawing justification. This is ignored if pScale is qtrue.

### EXTBMPref::drawMask()

|  |
| --- |
| void EXTBMPref::drawMask( HDC pHdc, qrect\* pRect, ePicSize pSize = ePicDef, ePicModes pWhich = picNormal, qbool pScale = qfalse, qjst pJstHoriz = jstLeft, qjst pJstVert = jstLeft) |

Draws the icons mask image into a device context.

1. **pHdc** - The drawing device to draw into.
2. **pRect** - The destination drawing rectangle.
3. **pSize** - The icon size to draw.
4. **pWhich** - The icon drawing mode to use.
5. **pScale** - If qtrue the icon is scaled to the full size of pRect.
6. **pJstHoriz** - The horizontal drawing justification. This is ignored if pScale is qtrue.
7. **pJstVert** - The horizontal drawing justification. This is ignored if pScale is qtrue.

### EXTBMPref::getBmpSize()

|  |
| --- |
| ePicSize EXTBMPref::getBmpSize( qlong pIconID ) |

Returns the icon size extracted from the icon id passed.

1. **pIconId** - The icon id to extract an icon size from.
2. **return** - The icon’s size.

**Note:** This is a static member function.

### EXTBMPref::getIconId()

|  |
| --- |
| qlong EXTBMPref::getIconId() |

Returns the icon id that was associated with this class at construction.

1. **return** - Returns the icon id.

### EXTBMPref::getRect()

|  |
| --- |
| void EXTBMPref::getRect( qrect\* pRect, ePicSize pSize = ePic16 ) |

Returns a bounding rectangle for the icon. The resulting size depends on the passed size parameter.

1. **pRect** - set to the correct bounding rectangle size.
2. **pSize** - Controls the returned size. This parameter defaults to the 16x16 size.

### EXTBMPref::hasMode()

|  |
| --- |
| qbool EXTBMPref::hasMode( ePicModes pMode = picNormal ) |

Used to determine if the icon this class refers to supports a particular drawing mode.

1. **pMode** - The icon drawing mode to test against. This parameter defaults to the normal drawing mode.
2. **return** - Returns qtrue if the icon does support **pMode**, and return qfalse if it does not.

### EXTBMPref::hasSize()

|  |
| --- |
| qbool EXTBMPref::hasSize( ePicSize pSize = ePic16 ) |

Used to determine if the icon this class refers to has a particular icon size.

1. **pSize** - The icon size to test against. This parameter defaults to the 16x16 size.
2. **return** - Returns qtrue if the icon does support **pSize**, and return qfalse it does not.

**Example:**

###### // This example gets a bitmap from Omnis using icon reference number 1000. // The icon reference is asked how big it should draw by default. The draw // method is called to draw the icon in a rectangle. The icon is centered // both vertically and horizontally in the rect. NOTE: it is not clipped to // the rectangle. It is very important to delete the bitmap reference // when you are finished.

###### WNDpaintStruct paintStruct; WNDbeginPaint( mHWnd, &paintStruct ); EXTBMPref bmpRef( 1000 ); ePicSize defaultSize = EXTBMPref::getBmpSize( 1000 ); bmpRef.draw( paintStruct.hdc, &drawRect , defaultSize, picNormal, qfalse, colNone, qfalse, jstCenter, jstCenter); WNDendPaint( mHWnd, &paintStruct );

### EXTBMPref::transparentColor()

|  |
| --- |
| qcol EXTBMPref::transparentColor() |

Used to get the transparent color of the bitmap image.

## EXTCURref Class Reference (v2.2)

The EXTCURref class gives your external components access to custom cursors in the OMNISPIC.DF1 and USERPIC.DF1. It allows you to create and set custom mouse cursors by specifying the custom cursor ID.

### EXTCURref::EXTCURref()

|  |
| --- |
| EXTCURref::EXTCURref( qlong pCursorID, qlong pDefault = 0, qapp pApp = 0 ) |

The constructor for the external cursor class. After construction, the class can be used to create a HCURSOR. When you have finished with the cursor reference, the class should be destructed. Destructing the class will not destroy the HCURSOR which was created from it.

1. **pCursorID** - the cursor associated with this class.
2. **pDefault** - the default cursor id is used when pCursorID is zero.
3. **pApp** - this parameter must be specified for web client components. See ECOgetApp.

### EXTCURref::~EXTCURref()

|  |
| --- |
| EXTBMPref::~EXTBMPref() |

The destructor for the external cursor class. The destruction of the class informs Omnis that you have finished with the cursor.

### EXTCURref::getCursor()

|  |
| --- |
| HCURSOR EXTCURref::getCursor() |

The getCursor function creates and returns a HCURSOR. You can effect the screen cursor by calling WNDsetCursor or WNDsetWindowCursor.

### EXTCURref::getCursorId()

|  |
| --- |
| qlong EXTCURref::getCursorId() |

Returns the cursor ID.