Neste guião irás aprender a **criar ecrãs** de **alta resolução**, a **inserir imagens, som** e a aceitar o **rato** como **INPUT** nos teus programas.

Começa por verificar **qual** a **resolução** do teu **ecrã**.

Procura duas imagens ao teu gosto na Internet. Uma **PNG** (sem fundo) e outra **exatamente** do **tamanho** da resolução do teu **ecrã**.

Cria uma **pasta** de nome **imagens** dentro da pasta **QB64** e **copia** as **imagens** retiradas da Internet para **dentro** dessa **pasta**.

Atribui-lhes um **nome simples** para poderes **trabalhar** com elas mais **facilmente**.

A partir de agora vais **aprender** a **inserir imagens** no teu **programa**. Começa por **definir duas variáveis** do tipo **long** de nome **imagem1** e **imagem2**.

De seguida **define** o **tamanho** do **ecrã** com que queres trabalhar. **Utiliza** a seguinte linha de **código** e **adapta** à **tua resolução** de ecrã.

```
Screen _NewImage(1920, 1080, 32)
```

Atenção: deves alterar os valores a vermelho para a tua resolução de ecrã.

Adiciona a seguinte **linha** de **código** para o ecrã **entrar** em **modo inteiro** quando corres o programa:

```
_Fullscreen
```

Vamos **carregar** uma **imagem**. **Escolhe** a imagem **PNG** e faz:

```
imagem1 = _LoadImage("imagens/car1.png")
```

Atenção: deves alterar o caminho (PATH) para a tua imagem.

Vamos fazer **aparecer** a **imagem** no **ecrã**. Neste caso vamos dizer para **aparecer** no **canto superior esquerdo** que corresponde à coordenada **(0,0)**.

```
_PutImage (0, 0), imagem1
```

Corre o teu **programa**. Deves ver a tua **imagem** aparecer em **modo** de **ecrã inteiro**.

Acrescenta uma **pausa** de **3 segundos**, **limpa** o **ecrã**, **remove** a **imagem** da **memória** e faz **aparecer** o **novo cenário** (a tua **imagem2**). O código para **remover** a **imagem1** da **memória** é:

```
_FreeImage imagem1
```

Corre o **programa** e irás ver a **primeira imagem** durante **3 segundos**, esta irá **desaparecer** e aparecer a tua **segunda imagem**.

PRO: converte as **variáveis imagem1** e **imagem2** num **array** de **duas posições**. Adiciona a **cada posição** do array o **_LoadImage** e faz **aparecer** com **_PutImage** recorrendo às **posições** do **array**.

Guarda o teu programa como **imagens.bas**

Ricardo Sérgio

Cria um **novo programa**.

Adiciona apenas este **código**:

```
Beep
```

Corre o **programa** e irás ouvir um **Beep**, claro está!

Adiciona agora outra **linha**:

```
Play "T120<<e8e8f8g8g8f8e8d8c8c8d8e8d8c12c4"
```

Já **ouves notas**?! Podes **procurar** na **internet** por **músicas** em **Qbasic** para poderes **adicionar** de **futuro** ao teu **projeto**.

Sabendo agora que o seguinte **código**:

```
Sound 262, 5
```

toca a nota **Dó** durante **5 ciclos** de **relógio** em **modo sintetizador**, **acrescenta código** ao teu programa, sabendo que Ré – **294**, Mi – **330**, Fá – **349**, Sol – **392**, Lá – **440**, Si – **494** e Dó (uma sétima acima) – **523**, para ouvires **Dó, Ré, Mi, Fá, Sol, Lá, Si, Dó.**

Cria agora uma **pasta sons** na tua pasta **QB64**.

Faz **download** do **som** a utilizar **aqui**. **Copia** o **ficheiro** para a **pasta sons**.

Copia e **cola** o seguinte **código** para o teu **programa**:

```
Declare Dynamic Library "winmm"
    Function PlaySound% Alias PlaySoundA (lpszName As String, Byval hModule As Integer, Byval dwFlags As Integer)
End Declare
retval% = PlaySound("sons/load_game.wav", 0, 1)
```

Corre o programa e **ouvirás** o som **Wav** a ser reproduzido. Retira outro **WAV** da **internet** e faz **tocar** no teu **programa** a seguir ao **anterior**.

Guarda o teu **programa** como **sons.bas**

Cria um **novo programa**. Neste **programa** vais **utilizar** o **rato** e aprender a **centrar objetos** no **ecrã** de **acordo** com a **resolução** do **ecrã** e também das **dimensões** da **imagem**.

Começa por **definir duas variáveis** do **tipo inteiro** de nome **largura** e **altura**.
Estas **variáveis** vão **guardar** a **largura** e **altura** do teu **ecrã** para serem **utilizadas** em **fórmulas** mais à frente. Para todos os efeitos, pretende-se que ao **alterares** a **dimensão** do teu **ecrã**, **todos** os **objetos** nele **fiquem formatados** tal **como** estavam **anteriormente**. Chama-se a isto **programação dinâmica**.

Atribui o **valor** da tua **resolução** de **ecrã** às **variáveis largura** e **altura** e depois **acrescenta** a seguinte **linha** de **código**:

```
Screen _NewImage(largura, altura, 32)
```

Se **reparares** em **vez** de **números entram** as **variáveis largura** e **altura** o que **permite** no **futuro**, caso pretenda **alterar** a **resolução** do **ecrã** apenas **mexer** no **valor** das **variáveis**.

Descarrega agora um **botão** [aqui](#).

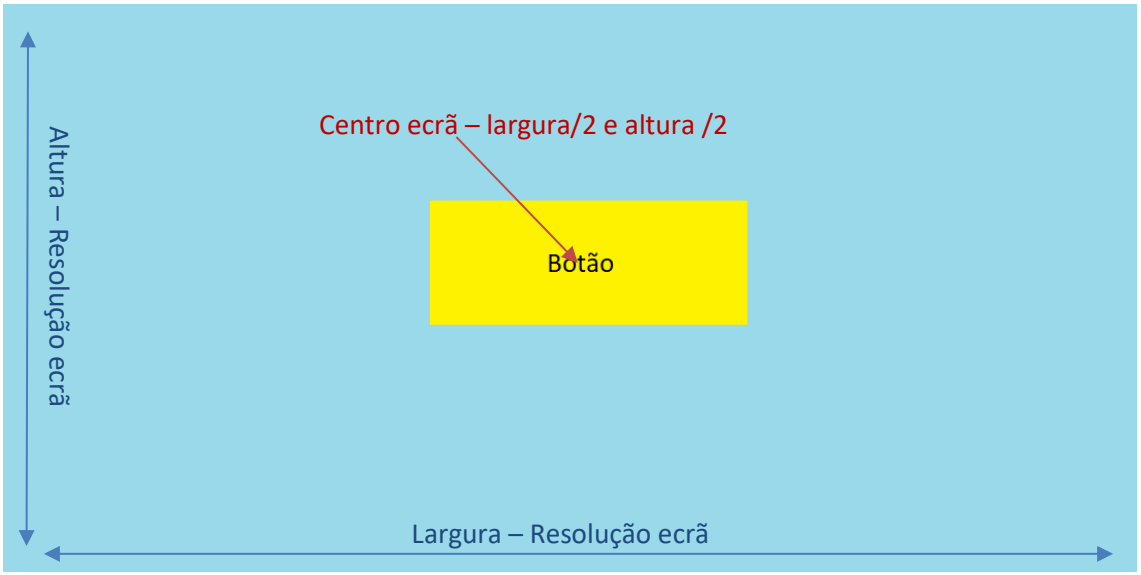
Copia o **botão** para a tua **pasta imagens**.

Cria uma **variável botao** e atribui-lhe a **imagem** do **botão** utilizando a função **_LoadImage** (podes utilizar a função **_PutImage(0,0)** para verificar se a **imagem aparece**).

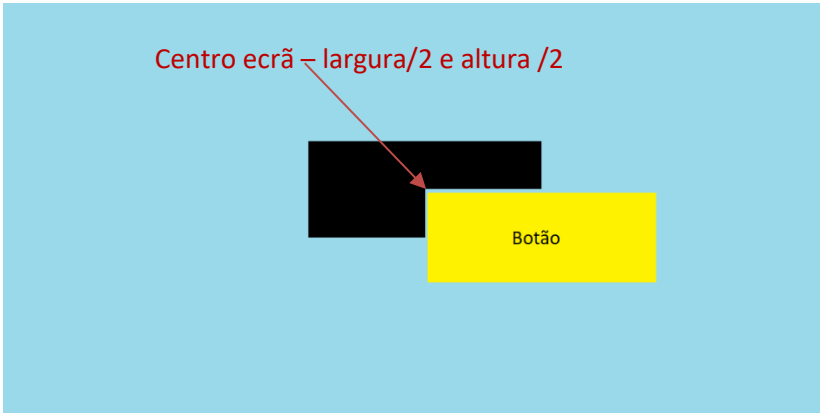
De seguida pretende-se **centrar** o **botão** no **ecrã**. Agora **pensa**… se o **ecrã tiver** uma **resolução diferente do teu colega do lado**, nos dois casos o **botão tem** de **aparecer centrado**, utilizando **exatamente** o **mesmo código**! Por outro **lado**, se **trocares** de **botão** por **um** de **outra dimensão** também este deve, **automaticamente**, ficar **centrado sem** haver necessidade de **mexer** no **código**. **Como fazer isto?!**

Cria mais **duas variáveis** do tipo **integer** de nome **centroBotaoX** e **centroBotaoY**.

Imagina o seguinte **botão centrado**.



Se **reparares** o **centro** do **botão** é o **centro** do **ecrã**. E o **centro** do **ecrã** é **obtido** pela **metade** da **largura** e **metade** da **altura**. O **problema** é que se eu **disser** que o **botão** deve **ficar** nas **coordenadas** do **meio** do **ecrã** ele **ficava assim**:



A parte **preta representa** onde **estava** o **botão** na **imagem anterior**. O **vértice** mais à **esquerda** do **botão** é que **ficava** no **centro**. **Como resolver**?

Ricardo Sérgio

Se **reparares** fica **exatamente meio botão**, tanto em **altura** como em **largura** fora do **sítio**, neste caso a **mais**. Então a **fórmula** final para **centrar** um **botão é**:

centroBotaoX = (largura - _width(botao)) /2

centroBotaoY = (altura - _height(botao)) /2

O **centro** do botão no eixo do **xx** será a **largura** do **ecrã subtraído** da **largura** do **botão** (acedido através da função _Width) e este resultado a **dividir por 2**. O **mesmo** foi aplicado para a **altura**.

Agora que sabes como centrar o botão adiciona o **código** que o **permita centrar** no **ecrã**. Utiliza a função:

_PutImage (X, Y), imagem

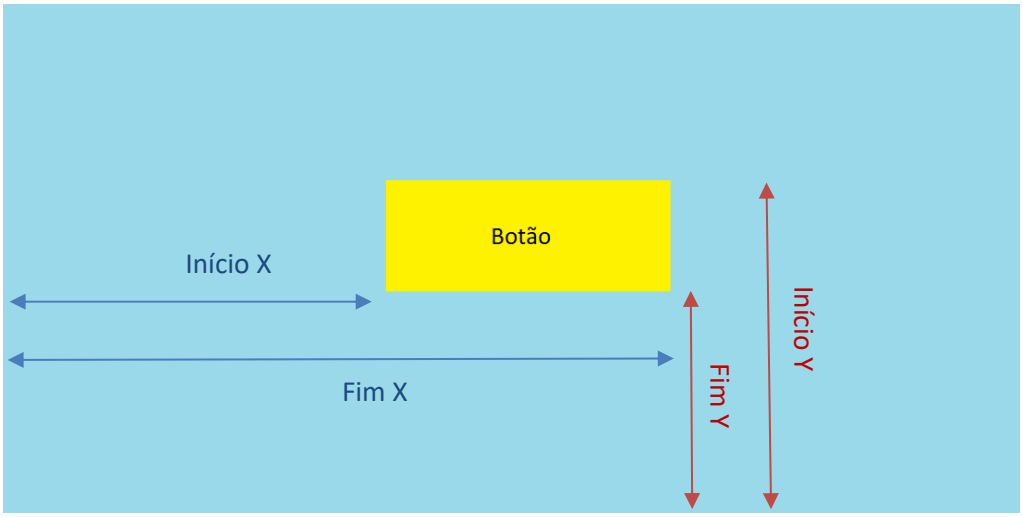
Corre o **programa** e **verifica** que o **botão** aparece **centrado**.

A partir de agora vamos **programar** o **click** no **botão**. Para isso temos de **mapear** o local onde estamos a **clicar**, uma vez que o **QBasic não identifica** a **imagem** como algo **distinto** no **ecrã**. Começa por **acrescentar** o seguinte **código**:

```
Do
    While _MouseInput

    Wend
Loop While INKEY$ = "" 'para ao tocar numa tecla
```

O **código** acima faz um **loop infinito** até que se **pressione** uma **tecla** (apenas para termos forma de parar o programa). Ao **chamar** o **_MouseInput** o **QBasic** está atento aos **movimentos** do **rato** e também aos **clicks**. Mas como saber se **clicamos** no **botão**? Bem, primeiro temos de estar **dentro** da **área** do **botão** e depois haver um **click** com o **botão esquerdo**. Vejamos a **área**:



Início de X – é centroBotaoX

Fim de X – é centroBotaoX + largura do botão

Início de Y – é centroBotaoY

Fim de Y – é centroBotaoY + altura do botão (porque o eixo do Y cresce para baixo)

Então temos de ter um **IF** que **teste** se o **rato** anda **dentro** desta **zona** e foi **clicado**:

```
If _MouseX >= centroBotaoX And _MouseX <= (centroBotaoX + _Width(botao)) And _MouseY >= centroBotaoY And _MouseY <= (centroBotaoY + _Height(botao)) And _MouseButton(1) = -1 Then

End If
```

Neste IF são testados **_MouseX** que é onde se encontra o ponteiro do rato no **eixo** do **xx** e o mesmo processo para o **_MouseY**. Caso eles estejam na área do botão (explicada anteriormente) e o **botão esquerdo** do rato pressionado **_MouseButton(1)** então faz o que estiver dentro do IF.

Acrescenta ao interior do IF forma de **limpar** o **ecrã** e **carregar** uma **nova imagem (pode ser a tua imagem2 do primeiro programa deste guião)**.

Atenção: às vezes é possível tocar ligeiramente fora do botão e ele assumir como botão. Isso acontece porque a imagem em si é mais alta e larga que o botão visível. Podes sempre ajustar retirando ou acrescentando pixels às condições do IF.

PRO: **pesquisa** (ou altera num editor de imagem) o **botão** para dizer **Quit**. Caso **pressiones** esse **botão** em vez de ir para o **jogo sai** do **programa**.

Guarda o teu programa como **rato.bas**

Chama o teu **professor** para **avaliar.**

Ricardo Sérgio