Real Time Gaming

1. Assumption

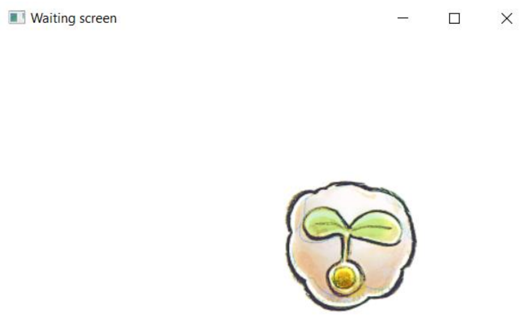
The application is built with Microsoft Visual Studio C++ 2015 with Win32 API library. I do not use MFC library because I want to optimize the code at my best.
The render function is totally reusable for other purposes.

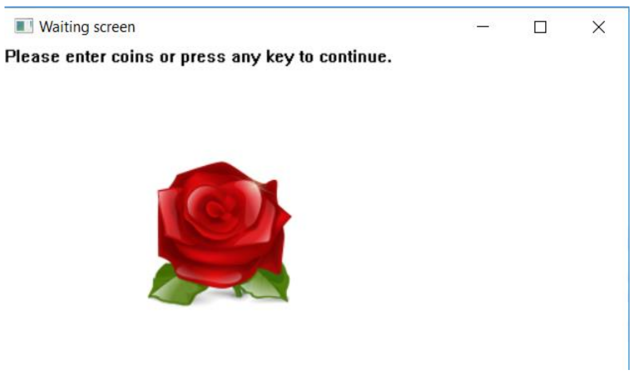
2. Render a frame at location(x,y)

Because the program uses both WM_TIMER and WM_PAINT events, this function passes screen memory hdc as a reference value.
This timer function will run every 1/100 second so the location can be updated regularly. However, the current frame should be updated a little bit later which is (1/100)*100 = 1 second. Frame changes from 1 to 8 then back to 1; therefore, the current frame is ***hImages[(time/100) % 8]*** with time is the number of 1/100 seconds that game has run so far.

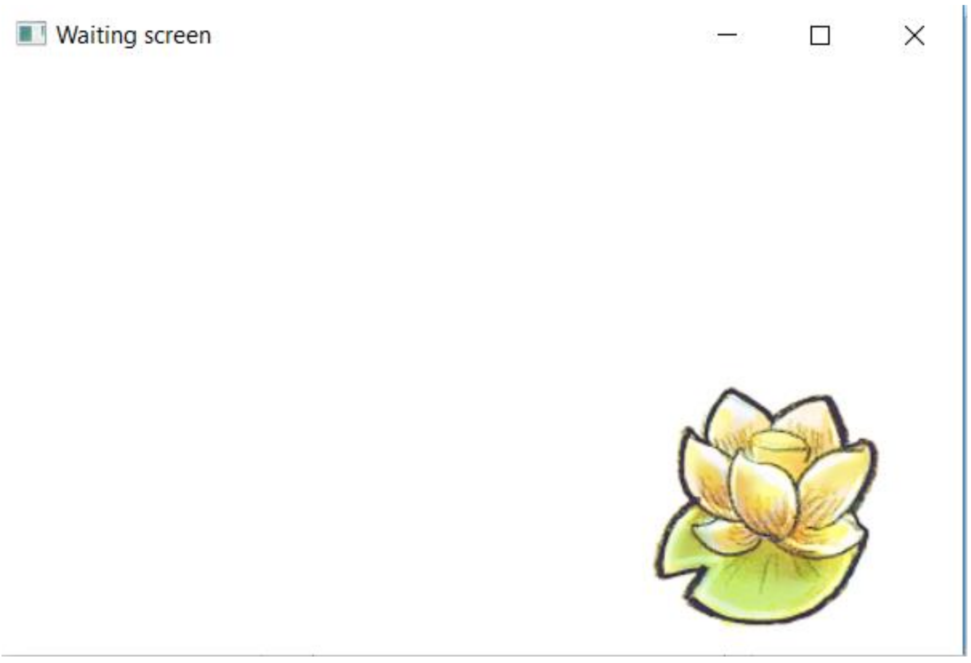
```
// Check whether continue or not
bool CanWeContinue()
{
	return bContinue;
}
// Render a frame at location(x,y), screen memory to draw on hdc, list of frames
hImages, running time, frame velocity (dx,dy)
void Render(HDC &hdc, int &time, HBITMAP* hImages, int &x, int &y, int& dx, int& dy)
{
	// Check continue flag
	if (CanWeContinue())
	{
		// Get screen rectangle
		RECT rect;
		GetClientRect(WindowFromDC(hdc), &rect);
		// Change frame location by adding velocity
		x += dx;
		y += dy;
		// Create a buffer memory for screen memory
		HDC hdcMem = CreateCompatibleDC(hdc);
		BITMAP bm;
		// Get the frame object information: width, height in bm object
		GetObject(hImages[(time / 100) % 8], sizeof(bm), &bm);
		// Draw the frame object on the buffer memory
		SelectObject(hdcMem, hImages[(time / 100) % 8]);
		// Draw the buffer memory on the screen memory
		BitBlt(hdc, x, y, bm.bmHeight, bm.bmHeight, hdcMem, 0, 0, SRCCOPY);
		// Delete the buffere memory
		DeleteDC(hdcMem);
		// Change velocity based on location(x,y), velocity(dx,dy),
screensize(rect)
		if (dx > 0 && x + bm.bmWidth > rect.right)
		{
			dx *= -1;
			}
			if (dx < 0 && x < rect.left)
			{
				dx *= -1;
			}
			if (dy > 0 && y + bm.bmHeight > rect.bottom)
			{
				dy *= -1;
			}
			if (dy < 0 && y < rect.top)
			{
				dy *= -1;
			}
		}
		else
		{
			TextOut(hdc, 0, 0, TEXT("Please enter coins or press any key to 
continue."), 48);
		}
}
```

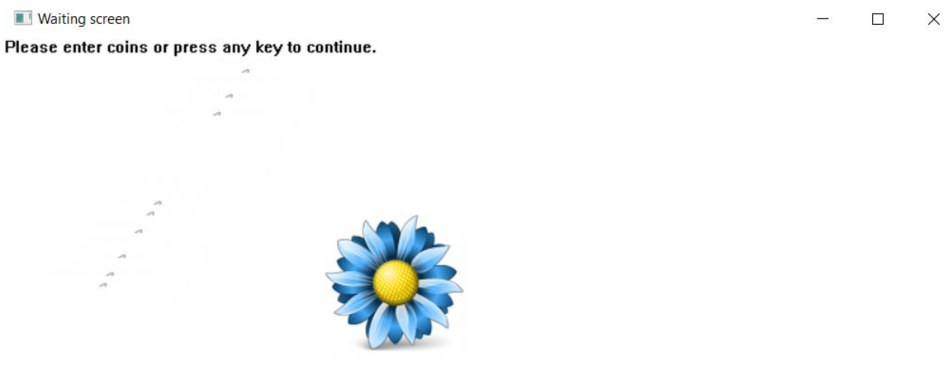
3. Screenshotsd



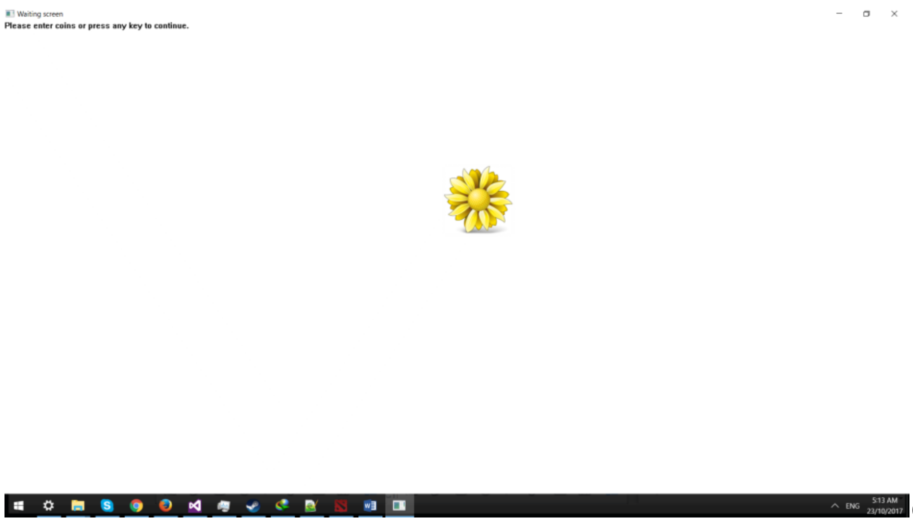


Please enter coins or press any key to continue.





Please enter coins or press any key to continue.



Please enter coins or press any key to continue.

d

4. Conclusion

What makes this app different from others?

- The most concise code at my best
- Reusable
- Using low-level programming library: Win32 API
- Well-commented