

# How to capture video from given window in C++ (unmanaged) using ByteScout Screen Capturing SDK

Write code in C++ (unmanaged) to capture video from given window with this step-by-step tutorial

Capture video from given window is easy to implement in C++ (unmanaged) if you use these source codes below. ByteScout Screen Capturing SDK is the tool for developers who want to add screen capturing in their application. Can record screen into video and into single screenshots. Output formats are WMV, AVI, WebM for video and PNG for screenshots. You can adjust output video size, quality, resolution, framerate, video and audio codecs. Includes special privacy features for blacking out sensitive information on screen. Can also capture video from web camera, can add overlays with text or images and you can use it to capture video from given window with C++ (unmanaged).

The SDK samples like this one below explain how to quickly make your application do capture video from given window in C++ (unmanaged) with the help of ByteScout Screen Capturing SDK. In your C++ (unmanaged) project or application you may simply copy & paste the code and then run your app! Code testing will allow the function to be tested and work properly with your data.

Free trial version of ByteScout Screen Capturing SDK is available for download from our website. Get it to try other source code samples for C++ (unmanaged).

C++ (unmanaged) - CaptureFromGivenWindow.cpp

```
// CaptureFromEntireScreen.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#import "BytescoutScreenCapturing.dll"

using namespace BytescoutScreenCapturingLib;
using namespace std;

void usage(ICapturer* capturer);
void setParams(int argc, _TCHAR* argv[], ICapturer* capturer);

int _tmain(int argc, _TCHAR* argv[])
{
    ::CoInitialize(0);

    // Create Capturer instance

    CLSID clsid_ScreenCapturer;
```

```

    CLSIDFromProgID(OLESTR("BytescoutScreenCapturing.Capturer"),
&clsid_ScreenCapturer);

    ICapturer* capturer = NULL;
    ::CoCreateInstance(clsid_ScreenCapturer, NULL, CLSCTX_ALL,
__uuidof(ICapturer), (LPVOID*) &capturer);

    if (!capturer)
    {
        _ftprintf(stdout, _T("Screen Capturer is not installed properly.));
        ::CoUninitialize();
        return 1;
    }

    capturer->put_RegistrationName(_T("demo"));
    capturer->put_RegistrationKey(_T("demo"));

    // Set capturing type
    capturer->put_CapturingType(catWindow);

    // Specify window to capture by its title.
    // Capturer will take the first window containing "Internet Explorer" in its
title.
    capturer->put_WindowToCapture(_T("Internet Explorer"));

    // You can also capture only a part of the window by specifying capturing
rectangle
    //capturer->put_CaptureRectLeft(25);
    //capturer->put_CaptureRectTop(25);
    //capturer->put_CaptureRectWidth(320);
    //capturer->put_CaptureRectHeight(240);

    // Set output video width and height
    capturer->put_OutputWidth(640);
    capturer->put_OutputHeight(480);

    // WMV and WEBM output use WMVVideoBitrate property to control output
video bitrate
    // so try to increase it by x2 or x3 times if you think the output video
are you are getting is laggy
    // capturer->put_WMVVideoBitrate(capturer->WMVVideoBitrate * 2);

    // uncomment to enable recording of semitransparent or layered windows
(Warning: may cause mouse cursor flickering)
    // capturer->CaptureTransparentControls = true;

    // set border style
?apturer.CaptureAreaBorderType = cabtDashed;

    // Set output file name
    capturer->OutputFileName = _T("Output.wmv");

    // Start capturing
    HRESULT hr = capturer->Run();

    // IMPORTANT: if you want to check for some code if need to stop the
recording then make sure you are
    // using Thread.Sleep(1) inside the checking loop, so you have the loop like

```

```

// Do
// Thread.Sleep(1)
// While StopButtonNotClicked

if (FAILED(hr))
{
    // Error handling
    CComBSTR s;
    capturer->get_LastError(&s);
    _ftprintf(stdout, _T("Capture failed: %s\n"), CString(s));
}
else
{
    _tprintf(_T("Starting capture - Hit a key to stop ...\n"));

    int i = 0;
    TCHAR *spin = _T("|/-\\");

    // Show some progress
    while (!_kbhit())
    {
        _tprintf(_T("\rEncoding %c"), spin[i++]);
        i %= 4;
        Sleep(50);
    }

    // Stop after key press
    capturer->Stop();

    _tprintf(_T("\nDone. "));
    getchar();
}

// Release Capturer
capturer->Release();
capturer = NULL;

::CoUninitialize();

return 0;
}

```

C++ (unmanaged) - stdafx.cpp

```

// stdafx.cpp : source file that includes just the standard includes
// CaptureFromEntireScreen.pch will be the pre-compiled header
// stdafx.obj will contain the pre-compiled type information

```

```
#include "stdafx.h"

// TODO: reference any additional headers you need in STDAFX.H
// and not in this file
```

C++ (unmanaged) - stdafx.h

```
// stdafx.h : include file for standard system include files,
// or project specific include files that are used frequently, but
// are changed infrequently
//

#pragma once

#ifndef _WIN32_WINNT           // Allow use of features specific to Windows XP or
later.
#define _WIN32_WINNT 0x0501   // Change this to the appropriate value to target
other versions of Windows.
#endif

#include
#include

#include
#include
#include
```

---

FOR MORE INFORMATION AND FREE TRIAL:

[Download Free Trial SDK \(on-premise version\)](#)

[Read more about ByteScout Screen Capturing SDK](#)

[Explore documentation](#)

[Visit \[www.ByteScout.com\]\(http://www.ByteScout.com\)](#)

or

[Get Your Free API Key for \[www.PDF.co\]\(http://www.PDF.co\) Web API](#)