CHAPTER 22 SOUND AND MUSIC

The integration of sound, music, and video into Microsoft Windows has been a significant advancement in the evolution of the operating system. Initially, multimedia support was introduced as the Multimedia Extensions to Windows in 1991. However, with the release of Windows 3.1 in 1992, multimedia support became a fully integrated category of APIs.



Over the years, the availability of CD-ROM drives and sound boards, which were considered rare in the early 1990s, has become a standard feature in new PCs. Nowadays, it is widely recognized that multimedia capabilities enhance the graphical user interface of Windows and add a valuable dimension to the computing experience. These features have extended the traditional role of computers beyond number crunching and text processing.

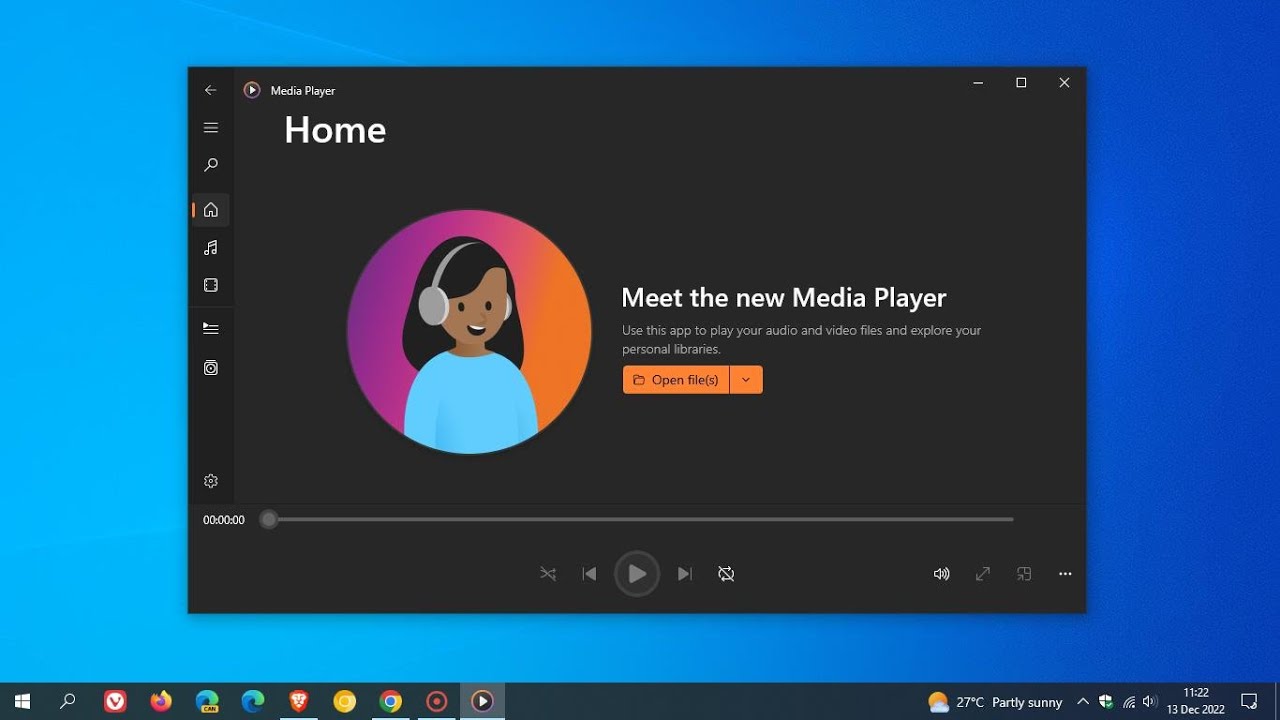


The integration of sound, music, and video into Windows has transformed the way users interact with their computers. It has opened up new possibilities for entertainment, communication, and creativity. From playing audio files and videos to creating multimedia presentations, Windows provides a platform for users to engage with various forms of media.

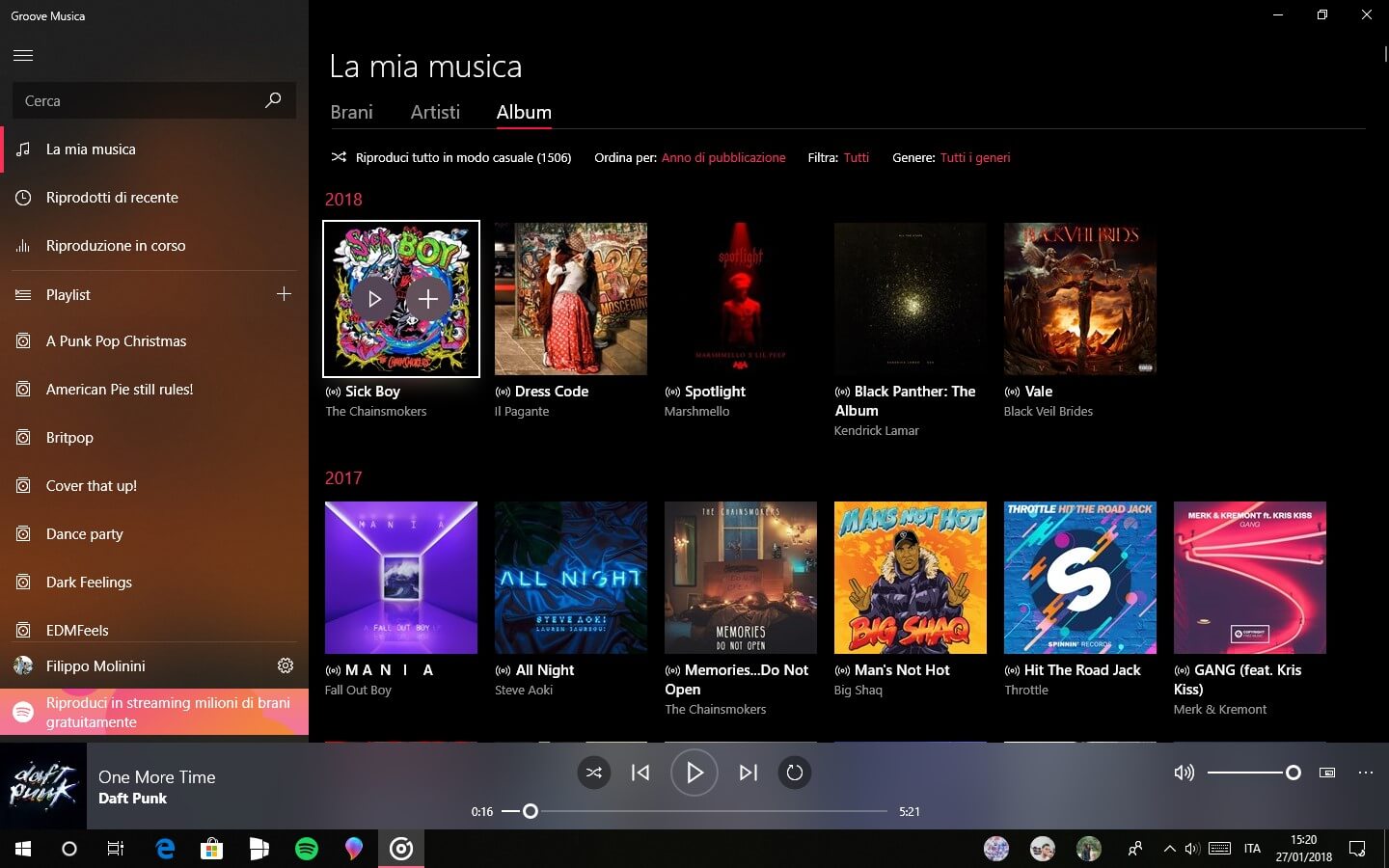
With each new version of Windows, including Windows 10, Microsoft has continued to enhance and expand multimedia capabilities. Windows 10 offers a wide range of features and tools that make it easier for users to enjoy and create multimedia content.



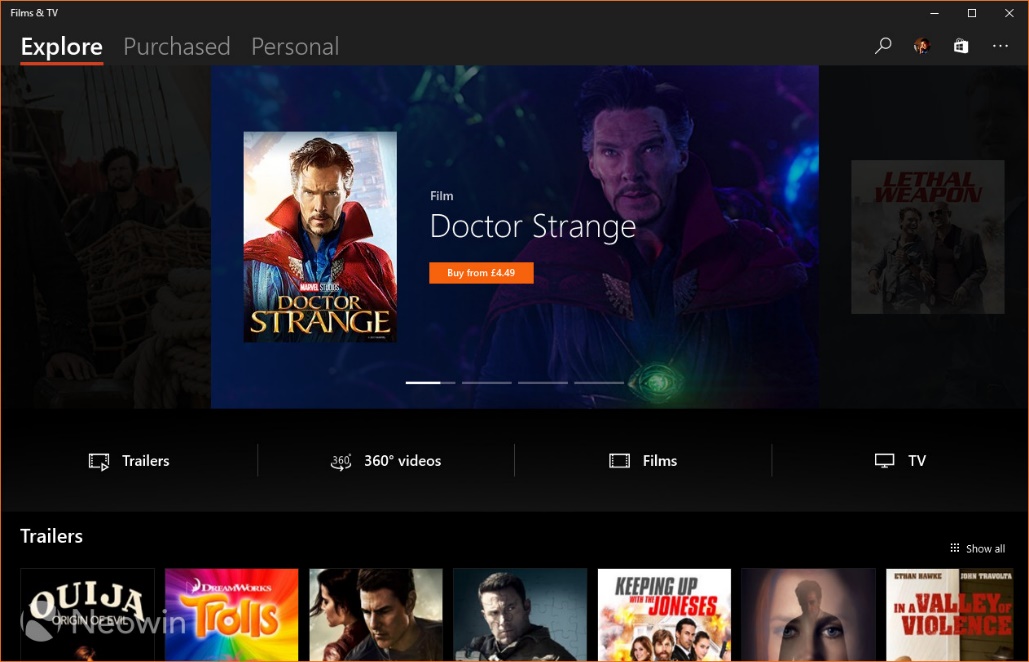
Media Player: Windows 10 includes the built-in Windows Media Player, which allows users to play a variety of audio and video file formats. It provides basic playback controls, playlist management, and the ability to create and manage media libraries.



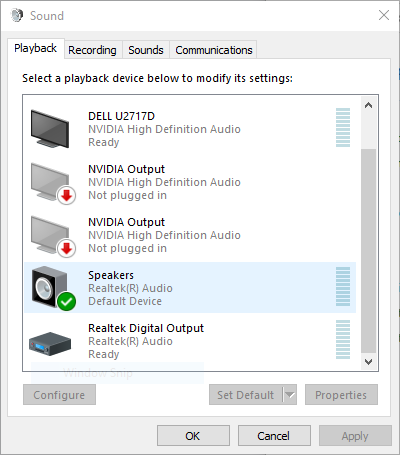
Groove Music: Windows 10 introduced the Groove Music app, which provides access to a vast library of songs and allows users to stream music from the Microsoft Store. It also supports local music playback and offers features like playlists, radio stations, and music recommendations.



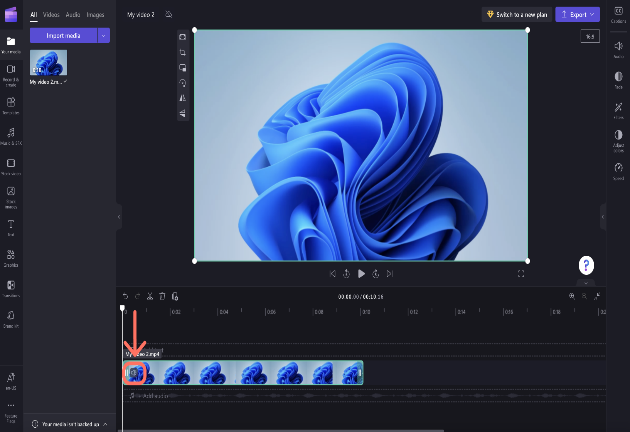
Movies & TV: The Movies & TV app in Windows 10 enables users to play movies and TV shows from their personal collection or purchase and rent content from the Microsoft Store. It supports a range of video formats and provides features such as playback controls, subtitles, and video casting.



Sound Settings: Windows 10 includes comprehensive sound settings that allow users to configure audio playback and recording devices, adjust volume levels, and apply audio enhancements. Users can also set default audio devices for different scenarios and customize sound effects.



Recording and Editing: Windows 10 provides built-in tools for recording and editing audio and video content. The Voice Recorder app allows users to record audio notes, interviews, or lectures, while the Photos app offers basic video editing capabilities, such as trimming, adding music, and applying visual effects.

Gaming and Streaming: Windows 10 incorporates features specifically designed for gaming and streaming. The Xbox app allows users to record and capture gameplay, stream games to other devices, and communicate with fellow gamers. Additionally, the Game Bar provides quick access to gaming features, including audio settings and broadcasting options.



Virtual Reality and Mixed Reality: Windows 10 includes support for virtual reality (VR) and mixed reality (MR) experiences. The Windows Mixed Reality platform enables users to immerse themselves in virtual environments, play VR games, and enjoy 360-degree videos and photos.



MULTIMEDIA CAPABILITIES

Multimedia capabilities are an essential and integrated part of the Windows operating system.

They encompass sound, music, and video, enhancing user experiences and extending the platform's capabilities.

Windows provides a device-independent multimedia API, which allows programmers to interact with various multimedia hardware devices through consistent function calls.

This device abstraction ensures compatibility and flexibility across different hardware configurations. Some of the key multimedia hardware devices supported by Windows include:

Waveform Audio Devices (Sound Cards): Sound cards convert analog audio signals from microphones and other input devices into digital samples for storage and processing (e.g., in .WAV files). They also convert digital waveforms back into analog sound for playback through speakers.



MIDI Devices: MIDI devices implement the Musical Instrument Digital Interface (MIDI) standard. They generate musical notes in response to MIDI messages and can interface with MIDI input devices such as musical keyboards and external synthesizers.



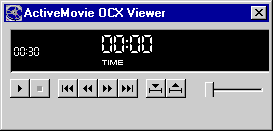
CD-ROM Drives (CD Audio): CD-ROM drives can play standard music CDs, allowing users to listen to audio tracks directly from the CD.



Video for Windows (AVI Video): Video for Windows is a software-based device in Windows that enables the playback of .AVI files (audio-video interleave). It provides support for playing video files and may also leverage video board hardware acceleration if available.



ActiveMovie Control: ActiveMovie Control expands video capabilities by providing support for additional movie formats, including QuickTime and MPEG. It can take advantage of video board hardware acceleration to enhance movie playback performance.

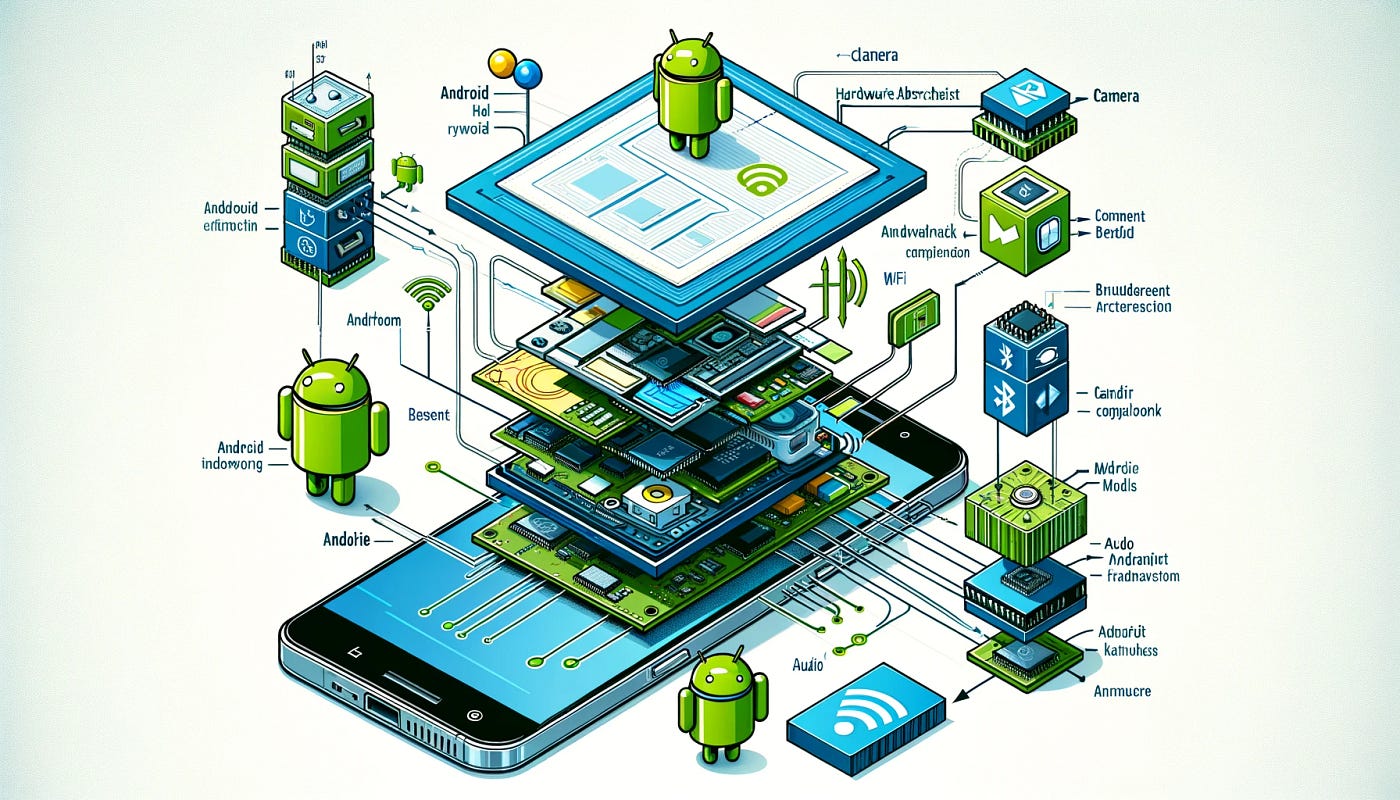


Laserdisc Players and VISCA Video Cassettes: Certain devices, such as laserdisc players and VISCA video cassettes, can be controlled via serial interfaces by PC software. This allows users to manage these devices and perform actions through their computer.

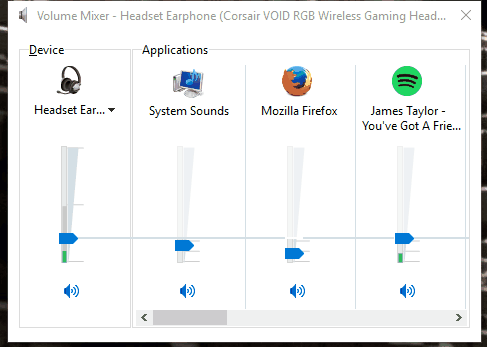


In addition to these hardware-specific features, Windows provides various core functionalities and concepts related to multimedia:

Device Abstraction: The Windows multimedia API abstracts the underlying hardware, providing a unified interface for programmers to access and control diverse multimedia devices. This allows developers to write multimedia applications that can work with different hardware configurations.



Hardware Mixing: Windows often includes a Volume Control application that allows users to blend output from multiple sources, such as waveform audio, MIDI, and CD audio. This enables users to control the relative volume levels of different audio streams.



Hardware Acceleration: Video boards can have dedicated hardware components that accelerate movie playback. This hardware acceleration improves performance and allows for smoother and more efficient video rendering.



Serial Interface Control: Some multimedia devices, like laserdisc players and VISCA video cassettes, can be controlled via serial communication interfaces. This allows users to send commands and manage these devices directly from their computer.

Overall, multimedia support in Windows has evolved significantly since its introduction as the Multimedia Extensions in 1991. With the widespread availability of CD-ROM drives and sound cards, multimedia capabilities have become standard in modern PCs. The integration of sound, music, and video into Windows has transformed the platform, going beyond traditional text and number processing and enabling immersive experiences for users.

Strategic API Design:

Dual-Layer Approach: Windows offers both low-level and high-level multimedia APIs, each serving distinct purposes and catering to different developer needs.

Low-Level Interfaces: Provide direct, granular control over hardware, enabling fine-tuning and optimization for demanding tasks or unique functionalities.

High-Level Interfaces: Simplify common operations, reducing development time and promoting code readability, often at the expense of some flexibility.

Low-Level Interfaces: Unleashing Hardware Potential:

Waveform Audio Mastery: waveIn and waveOut functions enable the recording and playback of digital audio signals, crucial for voice applications, music production software, and sound effects.

MIDI Orchestration: midiOut, midiIn, and midiStream functions control MIDI devices, essential for music creation software, interactive game audio, and sequencing hardware synthesizers.

Precision Timing: time functions establish high-resolution timers, ensuring accurate synchronization of multimedia events, particularly MIDI playback and real-time audio processing.

High-Level Interfaces: Streamlining Development:

MCI: The Versatile Orchestrator:

Offers a consistent interface for diverse multimedia devices, promoting code reusability and streamlining multi-device projects.

Its string-based form empowers rapid prototyping and scripting, enabling experimentation and customization.

Supports a comprehensive range of devices, encompassing audio, video, and optical media, fostering multifaceted multimedia experiences.

Beyond the Core: Expanding Possibilities:

*DirectX API: Powering Immersive Experiences:*

* While not extensively covered here, DirectX stands as a cornerstone for game development, multimedia applications, and graphics-intensive software.
* It provides unparalleled hardware acceleration, sophisticated 3D graphics rendering, advanced audio processing, and robust input device support.

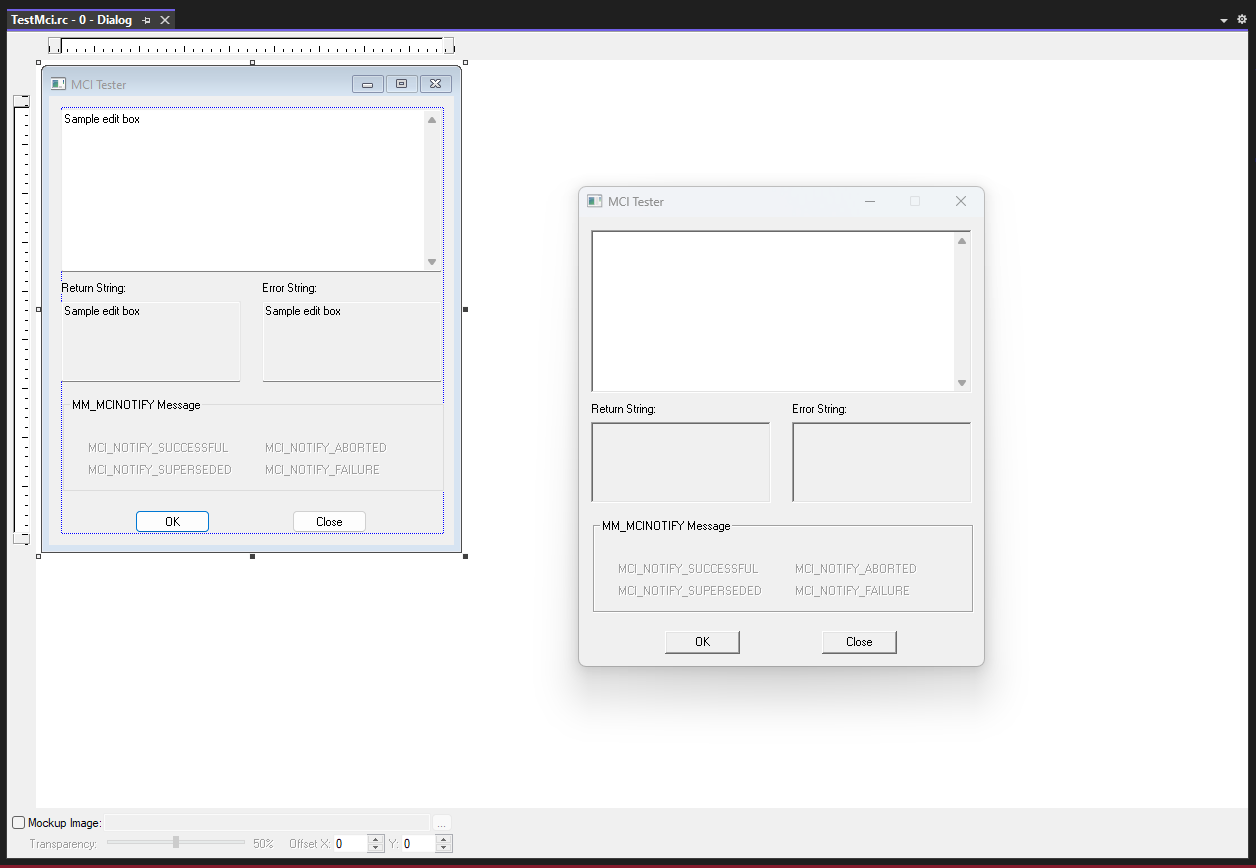
*Convenient Sound Utilities:*

* MessageBeep: Offers a straightforward mechanism for auditory feedback, enhancing user interaction and signaling important events.
* PlaySound: Simplifies the incorporation of sound effects and music into applications, contributing to engaging user experiences.

Key Considerations for API Selection:

* Project Requirements: The choice of API hinges on the specific needs of the application, balancing performance, ease of development, and hardware access requirements.
* Developer Expertise: Familiarity with low-level programming concepts is essential for effective use of low-level interfaces, while high-level interfaces often align with broader programming knowledge.
* Target Hardware: Understanding the capabilities of the intended multimedia hardware is crucial for optimal API selection and feature utilization.

TESTMCI PROGRAM



The TESTMCI program is a C application that allows users to interactively test and experiment with MCI (Media Control Interface) commands. MCI commands are used to control multimedia devices and perform operations such as playing audio or video files.

The program provides a simple graphical user interface (GUI) based on a modeless dialog box. The main window of the program contains an edit box where users can enter MCI commands. When the user presses Enter or clicks the OK button, the program retrieves the command from the edit box and passes it to the mciSendString function.

The mciSendString function is a key function provided by the MCI API. It takes a command string as input and executes the specified MCI command. In the TESTMCI program, the command string is obtained from the edit box, and the result of the command execution is stored in a string buffer.

After executing the command, the program displays the result in the "Return String" section of the window. This can include information such as status messages or data returned by the MCI command.

Additionally, the program retrieves the error code returned by the mciSendString function. If an error occurs during the execution of the command, the error code is used with the mciGetErrorString function to obtain a textual description of the error. The error description is then displayed in the "Error String" section of the window.

The program supports selecting multiple lines in the edit box. If multiple lines are selected, each line is treated as a separate MCI command and executed sequentially. The program processes each line individually and displays the corresponding result and error description for each command.

The program also includes a section for handling the MM\_MCINOTIFY message. This message is sent by the MCI subsystem to notify the application of changes in the status of multimedia devices or completion of asynchronous operations. The program enables or disables various controls in response to this message.



Overall, the TESTMCI program provides a convenient way for developers or users to experiment with MCI commands and observe their effects. It allows for interactive testing and troubleshooting of multimedia operations using the MCI API.

UNDERSTANDING MCI COMMANDS IN TESTMCI PROGRAM:

The TESTMCI program demonstrates the use of Multimedia Command Strings through two essential functions: mciSendString and mciGetErrorText. Here's an in-depth breakdown of the program's functionality:

mciSendString Function:

* The primary multimedia function used in TESTMCI is mciSendString. This function is responsible for sending command strings to the MCI subsystem for execution.
* When you type a command into the main edit window and press Enter, the program passes the entered string as the first argument to mciSendString.
* If multiple lines are selected in the edit window, the program sends them sequentially to mciSendString.
* The second argument to mciSendString is the address of a string (szReturn) that receives information back from the function.
* The returned information is then displayed in the "Return String" section of the program's window.

mciGetErrorText Function:

* The error code returned from mciSendString is passed to the mciGetErrorText function to obtain a text error description.
* The obtained error description is displayed in the "Error String" section of TESTMCI's window.

CD Audio Control:

* The program showcases the control of a CD-ROM drive to play audio CDs using MCI commands.
* Commands like open cdaudio, play cdaudio, pause cdaudio, and stop cdaudio are utilized to control audio playback.
* The "Return String" section displays information returned by the system in response to these commands.

Understanding Time Formats:

* The program allows users to interact with CD Audio by querying information like the total length of the CD, the number of tracks, and the length of individual tracks.
* Commands like status cdaudio length and status cdaudio number of tracks provide this information.
* The time format for CD Audio is explained, with examples like status cdaudio time format returning "msf" (minutes-seconds-frames).

Setting and Manipulating Time:

* The program demonstrates how to set and manipulate time formats, such as changing the time format to "tmsf" (tracks-minutes-seconds-frames).
* Users can play specific tracks or set the playback range using commands like play cdaudio from with specified time values.

Additional Features:

* The program handles error conditions gracefully, displaying appropriate messages in case of failures.
* It provides a practical and interactive way to explore MCI commands for CD Audio control.