CHAPTER 19: MULTIPLE DOCUMENT INTERFACE

MDI (Multiple-Document Interface):

It's a design approach for applications that manage multiple documents in Microsoft Windows.

It allows users to work with several documents simultaneously within a single application window.

Think of it like having multiple tabs open in your web browser, but for different documents within the same program.

Key concepts:

* Parent window: The main application window that holds all the document windows.
* Child windows: The individual document windows that reside within the parent window.
* Client area: The part of the parent window where the child windows are displayed.

How MDI works:

Opening documents: Each document opens in its own child window within the parent window.

Arranging windows: Users can arrange child windows in various ways:

* Tile them horizontally or vertically to view multiple documents side-by-side.
* Cascade them to overlap like a stack of cards.
* Arrange them manually by dragging and resizing.

Switching between windows: Users can switch between documents by clicking on the desired child window, or using keyboard shortcuts.

Managing windows: The application provides features to manage child windows, such as:

* Arranging them (as mentioned above).
* Minimizing, maximizing, or closing them.

Benefits of MDI:

Efficient multitasking: Enables working on multiple documents simultaneously without cluttering the desktop with separate application windows.

Easy comparison: Allows side-by-side comparison of documents for easier referencing and editing.

Organization: Helps keep related documents grouped together within a single application.

Examples of MDI applications:

* Older versions of Microsoft Office applications (Word, Excel, PowerPoint)
* Adobe Photoshop
* Many text editors and code editors

Modern trends:

While MDI was common in older Windows applications, newer applications often use alternative approaches like:

* Tabbed interfaces (similar to web browsers)
* Single-document interfaces (SDI) where each document opens in its own separate window

These approaches offer different user experiences and trade-offs in terms of window management and organization.

MDI in Historical Context:

Early Complexity: While MDI was introduced in Windows 2.0, its implementation was challenging for developers due to intricate programming requirements.

Greater Support: Windows 3.0 and subsequent versions significantly streamlined MDI development by providing built-in support and enhancements.

Key Elements of MDI Applications:

Main Application Window:

Similar to standard application windows, featuring a title bar, menu, sizing border, system menu icon, minimize/maximize/close buttons.

Unique client area, often called the "workspace," designed specifically to house child windows instead of direct program output.

Child Windows (Document Windows):

Resemble small application windows within the main window's workspace.

Possess title bars, sizing borders, system menu icons, minimize/maximize/close buttons, and potentially scroll bars.

Notably lack their own menus, relying on the main application window's menu for actions.

Active Document:

Only one child window can be active at a time, indicated by a highlighted title bar.

Active window appears in front of other child windows, ensuring focus.

Document Containment:

Child windows are confined to the workspace area within the main application window, preventing them from extending beyond its boundaries.

Essential Programming Considerations:

* Window Management: MDI support involves specific functions and data structures for managing the relationship between the main application window and its child windows.
* Message Handling: MDI applications must process window messages differently to coordinate interactions between the main window and child windows.
* User Experience: Programmers must carefully design the MDI interface to ensure intuitive navigation, window management, and document interaction for users.

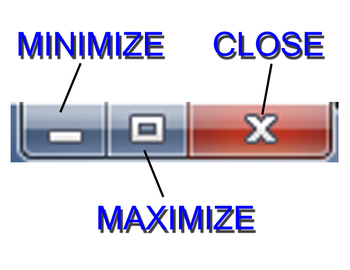
BEYOND INITIAL IMPRESSIONS:

While Multiple-Document Interface (MDI) might seem straightforward at first glance, it quickly reveals a nuanced set of complexities that demand careful attention from developers. Let's delve deeper into the key elements and considerations involved in crafting effective MDI applications.

Unveiling the Nuances:

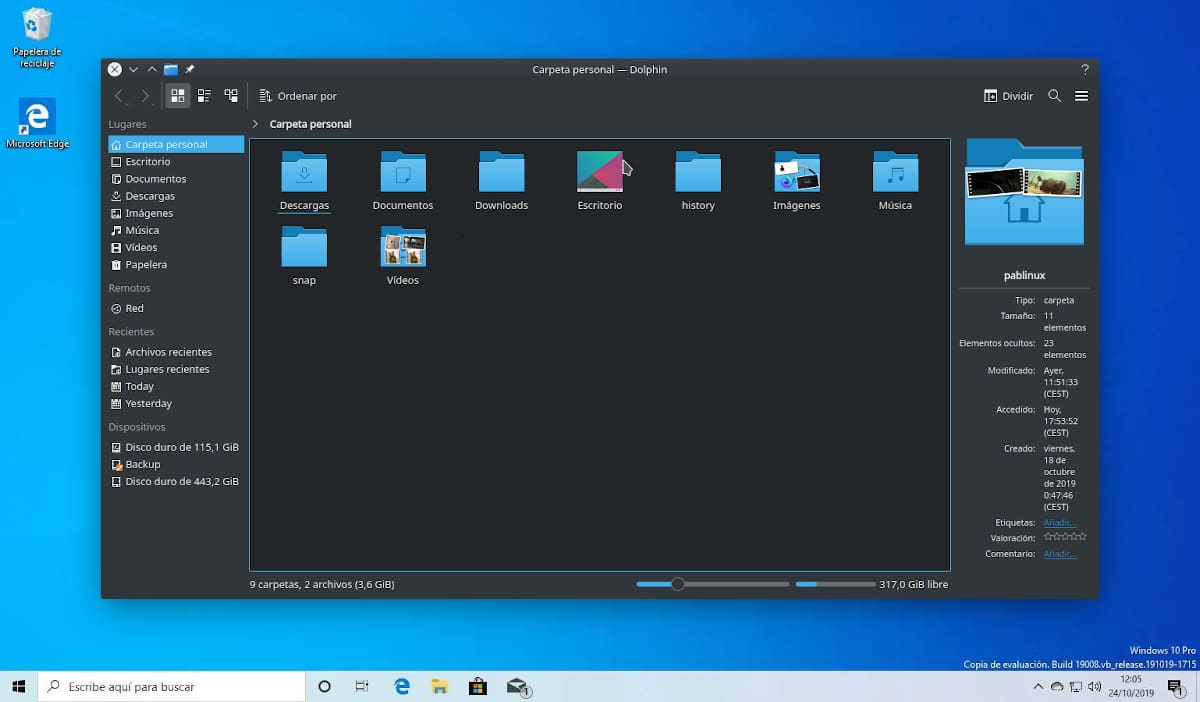
Minimized Windows:

* These windows gracefully tuck themselves away as compact title bars with icons, residing at the bottom of the workspace.
* To aid visual clarity, they typically adopt distinct icons from the main application window, ensuring easy differentiation.



Maximized Windows:

* In this state, they seamlessly blend their title bars with the main window's, effectively appending document filenames for a cohesive display.
* System menu icons and close buttons relocate to the main window's menu bar, maintaining a unified interface while preserving access to essential controls.



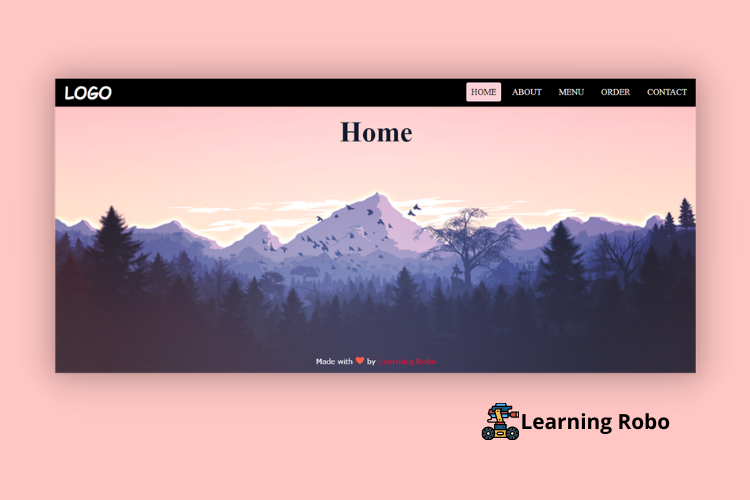
Keyboard Shortcuts for Efficiency:

* Ctrl+F4 offers a swift way to close document windows, while Alt+F4 retains its traditional role of closing the main application window.
* Ctrl+F6 enables effortless switching between child windows, promoting fluid navigation.
* Alt+Spacebar remains dedicated to invoking the main window's system menu, and Alt+- (minus) unlocks the active child window's system menu for granular control.



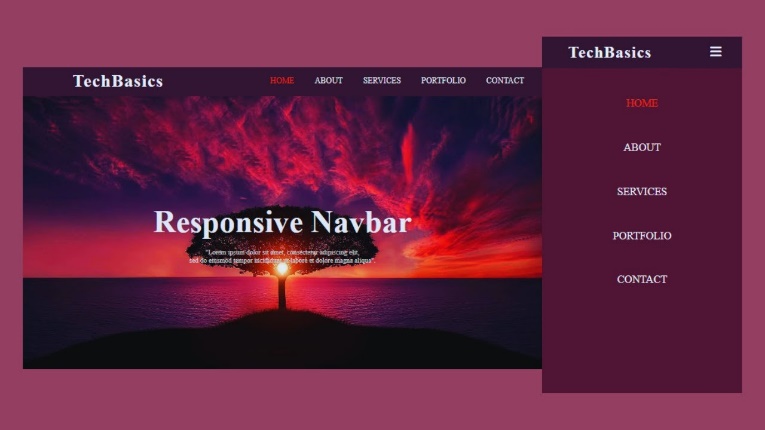
Menu Navigation: A Unified Journey:

* Cursor keys gracefully guide users through MDI menus, initiating their journey at the application system menu, then gracefully transitioning to the active document system menu, before finally arriving at the first item on the main menu bar. This intuitive flow promotes a cohesive user experience.



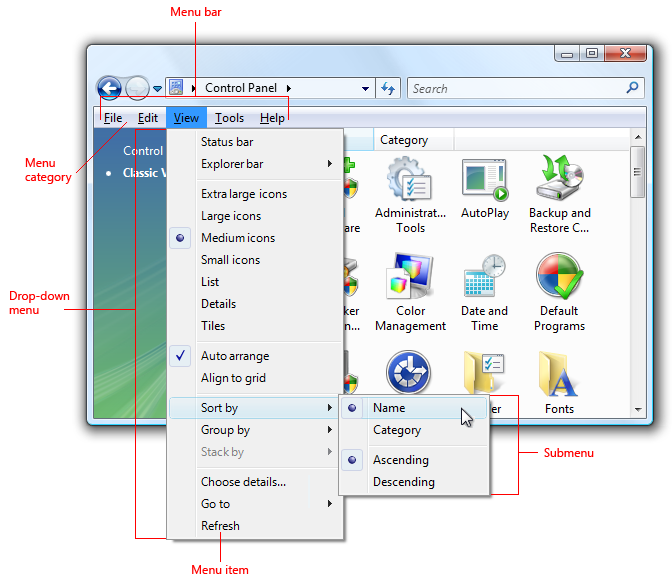
Context-Aware Menus:

* Intelligent applications dynamically adapt their menus based on the active document's type, ensuring that only relevant actions are presented.
* This responsiveness extends to the absence of document windows, where menus gracefully streamline to showcase options for opening or creating new documents, minimizing clutter and guiding users towards primary tasks.



The Window Menu: A Command Center for Document Management:

* Nestled strategically within the top-level menu bar, typically preceding the Help menu, this dedicated menu empowers users to effortlessly arrange and access document windows.
* It offers convenient options for cascading windows in an overlapping fashion or tiling them for full visibility, catering to different organizational preferences.
* A comprehensive list of all open document windows resides within this menu, enabling users to swiftly select and activate desired documents with a single click.



Windows 98: A Helping Hand:

* Windows 98 extended a valuable hand to developers by incorporating built-in support for these MDI features, significantly reducing the programming effort required for their implementation.
* While this support does introduce some overhead, it's a small price to pay compared to the manual implementation of such a comprehensive set of functionalities.

