**Introduction**

DuplicateSearcher was created by Unisoft to be an advanced utility for finding duplicate files and folders. One goal of DuplicateSearcher was to generate results that could be used by other applications. The process of identifying duplicates can be complex and time-consuming. There are many attributes of files and directories that can be compared, and the criteria for identifying a duplicate often depends on the file's type and contents. DuplicateSearcher enables the user to specify fairly extensive criteria for identifying duplicate files and directories.

DuplicateSearcher has two operating modes: console and Graphical User Interface (GUI). The console's executable file is dsc.exe, and the GUI's executable file is dsgui.exe. Both have the same search capabilities; however, only the GUI supports performing operations on duplicates, such as deleting, moving, or recycling. The GUI makes the criteria easier to specify, and the search results are easier to view. The console makes it easier to integrate the functionality of DuplicateSearcher with other applications. For example, the output from the console could be redirected to a file that is processed by a separate application.

**System Requirements**

DuplicateSearcher is a .NET 2.0 application. The .NET 2.0 framework must be available for DuplicateSearcher to function. The .NET 2.0 framework can be downloaded here: <http://msdn.microsoft.com/en-us/netframework/aa731542.aspx>. The minimum hardware requirements typically depend on which operating system is being used, but 512MB - 1GB of system memory should suffice.

**General Search Criteria**

The search criteria that can be specified in the GUI have direct correlation to the command line options available to the DuplicateSearcher console. All search criteria must specify at least one 'search location', which can any drive, network path, or directory that is accessible to the user. If DuplicateSearcher cannot access a file, directory, or subdirectory, it will ignore it and continue searching. Both the console and GUI can specify 'built-in' search criteria for finding duplicate video, audio, etc, files.

**Technical Details**

DuplicateSearcher uses a combination of multithreading, advanced data structures, and various search algorithms to minimize the time it takes to find duplicates and present them to the user. Internally, DuplicateSearcher creates a 'tag' for each file and directory. Depending on the specified search criteria, this 'tag' can be a combination of a file or directory's attributes, including but not limited to its name (exact or SOUNDEX), size, header, footer, or hash of the contents. The 'tag' is used to organize the files and directories using a self-balancing AVL binary search tree (BST). As the file system is searched, references to files and directories are added to the BST using the generated 'tag'.

DuplicateSearcher employs multiple threads to maximize search speed. By default, DuplicateSearcher will assign one thread to each search location. As a general rule, peripheral I/O (such as access to hard drives and network controllers) is slower than the CPU and system memory. Instead of making DuplicateSearcher wait on various peripherals, it is almost always faster to attempt concurrent access to multiple peripheral devices using multiple threads. Nevertheless, there are some cases where concurrent access to peripheral devices might make the search process slower, such as if one hard drive contains multiple partitions. DuplicateSearcher allows the user to specify a maximum thread count.