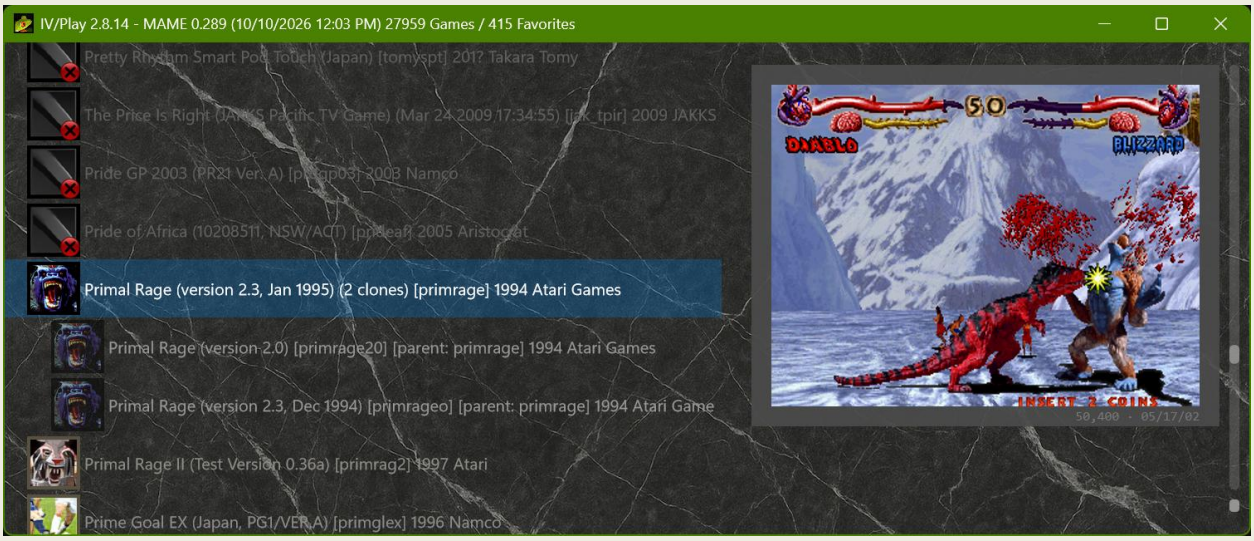
# IV/PLAY USER GUIDE

**John L. Hardy IV**

v.2.8.14 - Saturday, October 10, 2026



# TABLE OF CONTENTS


Table of Contents ..... 2

Overview ..... 8

IV/Play vs. MAMEUI ..... 8

Requirements ..... 8

Installation ..... 9

Resources ..... 9

Major Application Functions ..... 9

- Import XML Data from MAME.exe ..... 9
- Configuration Screen (F1) ..... 9
- Display XML Data in UI ..... 10
  - Title Bar ..... 10
  - Game List ..... 10
  - Favorites ..... 11
  - Icons ..... 11
  - Art Area ..... 11
  - Text Filter Dialogue ..... 12
  - Background Selection and Randomization ..... 12
  - Properties View ..... 13
- Launch Game ..... 13
- Accelerator / Navigation Keys ..... 13

External Files ..... 15

- IV-Play.cfg ..... 16
- IV-Play.dB ..... 16

Updates ..... 16

- IV/Play 1.5 Features ..... 16
  - Unlimited Art Types ..... 16

History.XML / MAMEInfo.DAT Support ..... 17
Favorites Toggle ..... 17
Non-Working Game Art/History Support ..... 17
Command Line Override ..... 17
IV/Play 1.6 Features ..... 17
IV/Play 1.7 Features ..... 17
Clone and Non-Working Filters ..... 17
IV/Play 1.8 Features ..... 17
IV/Play 2.0 Features ..... 18
Move from .NET 4.5 to .NET 8.0 & Exit from WPFWrapper to Pure WinForm ..... 18
GPU Render Pipeline ..... 18
LiteDB to SQLite ..... 18
Support for High DPI Monitors / Large Icon Mode / Scaling Improvements ..... 18
Support for Software List Devices ..... 19
Custom Game List Support ..... 19
Grid View ..... 19
Background Brightness Detection Mode ..... 20
Image Atlas for Icons and Binary Versions of History.XML and MAMEINFO.DAT ..... 20
Additional Features ..... 20
IV/Play 2.1 Features ..... 20
Move from .NET 8 to .NET 9 ..... 20
Additional Features ..... 20
IV/Play 2.2 Features ..... 21
Vertical Scrollbar for Game List ..... 21
Filter Improvements and Additions ..... 21
Additional Features ..... 21
IV/Play 2.3 Features ..... 22

Reintroduced Legacy Error Detection ..... 22
BigBench Benchmark Script ..... 22
Dynamic Color Themes ..... 22
Expanded Key Customization ..... 23
Favorites.ini Audit ..... 23
Additional Features ..... 23
IV/Play 2.4 Features ..... 24
Software List Filtering & Favorites ..... 24
F1 Customizable Keys Section ..... 24
The Arcade Gamelist ..... 24
Support for Zip and .7z Art Packs ..... 24
Multiple MAME Launches and Tiling ..... 24
Move From .NET 9 to .NET 10 ..... 24
Additional Features ..... 25
IV/Play 2.6 Features ..... 25
Flat View & Global Sort ..... 25
Deeper DAT Integration for Filter Searches ..... 25
Available ROMs Gamelist ..... 26
Platform-Based Softlist Filter Grouping ..... 26
Native AOT Transition / Performance ..... 26
Machine Status Aware Icon Borders ..... 26
Hidden.ini ..... 27
Custom Game List Generation from Filter Results ..... 27
Number of Clones / Parent ..... 27
Command Line Override Autocomplete ..... 27
Additional Features ..... 27
IV/Play 2.8 Features ..... 28

DPI & Layout Modernization....28
Internal HiScore.ini....28
MAME Source .cpp Added to DAT Peek....28
Unattended Benchmarking (-bench)....29
Additional Features....29
IV/Play 2.8.12 Features....29
IV/Play 2.8.14 Features....30
Appendix 1 - Filter & Search Guide....31
Simple Text Search....31
System Name Routing....31
Date & Decade Filtering....41
Field-Specific Searches....42
Advanced Operators....46
Putting It All Together....47
Appendix 2 - User Reference....48
Architecture....48
Appendix 3 - Deep Technical Reference....50
Unified Architecture....50
Rendering Engine....51
GameList Partial Class Structure....51
Data Layer....52
Raw ADO.NET (ADO-Only)....52
MachinesSummary Table....52
Cache Result Provenance....52
Atomic Writes....52
String Interning....53
Commands Table....53

Multi-Layered Caching ..... 53
Filter Engine ..... 54
Pipeline ..... 54
DAT Integration and MetadataHydrator ..... 54
Autocomplete ..... 54
Available Gamelist Pipeline ..... 54
Flat View and Global Sort ..... 55
Machine Status-Aware Icon Borders ..... 55
Settings Management and Snapshot System ..... 55
Diagnostics ..... 55
Runtime Diagnostic Identity ..... 56
Benchmark Engine ..... 57
Reading Benchmark Reports and Histories ..... 57
Concurrency Model ..... 58
Live ROM Monitoring (FileWatcherService / ChangeQueue / BatchProcessor) ..... 58
Extracted Service Classes ..... 59
Native AOT Runtime Model ..... 59
Recorded Performance Snapshot — May 2026 ..... 60
Appendix 4 - Development History ..... 60
Foundation Stabilization & Early Architecture (June–July 2025) ..... 60
Rendering, DPI, and Softlist Depth (August–September 2025) ..... 60
Dynamic Themes, Scrollbar, and Input Infrastructure (October 2025) ..... 61
Softlists Become First-Class Citizens (November 2025) ..... 61
Mixed-View Filter Hardening (December 2025) ..... 61
Family Grouping, Properties Overhaul, and the Schema Bug Fix (January–February 2026) ..... 61
Filter Polish, Autocomplete, DAT Integration (Late February–Early March 2026) ..... 61
Mixed-View Unification and Native AOT (March 2026) ..... 62

Forensic Audit & Structural Hardening (April 2026) ..... 62
Structural Re-org and Service Extraction (May 2026) ..... 63
Startup Accuracy, Update Identity, and DAT Correctness (July 2026) ..... 63
High-DPI Headroom and the Icon Master (late July 2026) ..... 64
The Benchmark Engine and Unattended Runs (late July–August 2026) ..... 64
Concurrency Formalization and Contract Closure (August 2026) ..... 64
Search Semantics, Software Identity and Refresh Repair (September 2026) ..... 65
Benchmark Provenance and Guard Verification (September 2026) ..... 65
F1, HBMAME and Icon Ownership (late September–October 2, 2026) ..... 65
List Accuracy and Refresh Selection (October 2026) ..... 66
Key Bug Archive ..... 66
Appendix 5 - Performance Delta Timeline ..... 67
Architectural Evolution Summary ..... 67
Entry 1: v2.4.14 — February 16, 2026 ..... 68
Entry 2: v2.6.2 — March 17, 2026 ..... 69
Entry 3: v2.8.4 — May 10, 2026 ..... 69
User Guide Version History ..... 70
Credits ..... 70

## OVERVIEW

IV/Play (pronounced 'Four Play') is a desktop/keyboard-oriented GUI front-end for MAME™. It was designed and its coding commissioned in 2011 by original MAMEUI team member John IV, from his original 2006 concepts. It has a particular feature-set, is keyboard driven, and utilises many of the navigation and keyboard shortcuts of MAMEUI. It targets native Microsoft Windows, with Windows 11 preferred. IV/Play leaves emulator configuration to MAME's own ini files and passes launch arguments to the linked executable. HBMAME can also be linked; its name appears in the title bar and F1 About. IV/Play is self-contained and portable for easy transport and mobility.

In 2025 a modernization was undertaken of the 14-year-old codebase to update from its .NET 4.5 WPFWrapper origins to its current state, rebuilt entirely using WinForms, with WPF fully deprecated. It provides a high performance, RAM based experience on i7-12700K, 64GB RAM, GPU level machines. The goal is near instant launch, game list population, and zero slowdown displaying any art assets (icons, snaps, flyers, bkgrounds, etc.) utilizing as much RAM and graphic (GPU) hardware as needed; employing modern design techniques and technologies.

**Naming conventions**: The application name is 'IV/Play' but due to WinOS file-naming limitations, 'IV-Play' is used where necessary.

In the 1080p 96DPI era, the initial IV/Play dimensions were 1015x432; a 2.35:1 aspect ratio in honor of the 1960's TohoScope tech used in the filming of Godzilla movies. Today, on 4K monitors, the same ratio is used but at a default of 1522x648 (144dpi and 1.5X scaling).

## IV/PLAY VS. MAMEUI

MAMEUI is an integrated UI to MAME and is compiled using the base source. It was the first 'official' MAME port to the Windows platform and the most popular version of MAME for decades. It leveraged the underlying MAME code and depended on it to produce its UI and functionality. This was its strength and weakness. As MAMEUI matured (b. 1997), the original development team waned over time. It often took long stretches to catch up with the sweeping core changes that periodically took place in the underlying MAME engine. While the number of supported arcade games, gambling 'fruit machines', computers, chess games, calculators, consoles, and miscellaneous other ROM based electronics neared 40,000 sets, the UI slowed down when all the icons and screenshots were processed by the near 3-decade old controls. This introduced memory leaks over time and the user experience degraded. Changes to the core at the time also resulted in huge memory footprints and 5x longer load times.

IV/Play was born out of the desire to continue a similar 'classic' desktop interface to MAMEUI without the attendant issues of being coupled to the core.

## REQUIREMENTS

Windows 11 is the preferred platform. Faster modern processors and solid-state drives (SSDs / NVMes) help considerably with image/resource loading speed. IV/Play is DPI aware and scales its UI and elements for high-resolution monitors. It ships as a self-contained Native AOT (Ahead-of-Time) .NET 10 executable, so no separate .NET runtime download is required. Its GPU-based rendering and caches are designed for modern systems with ample RAM and a capable GPU.

## INSTALLATION

The most straightforward installation option is to place the IV/Play files into the top level of an existing MAME directory structure alongside the respective .exe. IV/Play will utilise the structure's icons, art types, and bkground folders of an old MAMEUI installation. IV/Play does not rely on registry settings or an install so it is easily portable on external media like a thumb drive with MAME and attendant files. IV/Play will create its owned named directory under its location for its various assets, but also maintain a same level favorites.ini file, custom game list and its *.cfg file, which allow them to survive the asset directory removal during testing or refresh. The attendant e_sqlite3.dll must also be present with the .exe.

## RESOURCES

The current build, its recommended icons, and snapshots for IV/Play are found at the IV/Play homepage.

## MAJOR APPLICATION FUNCTIONS

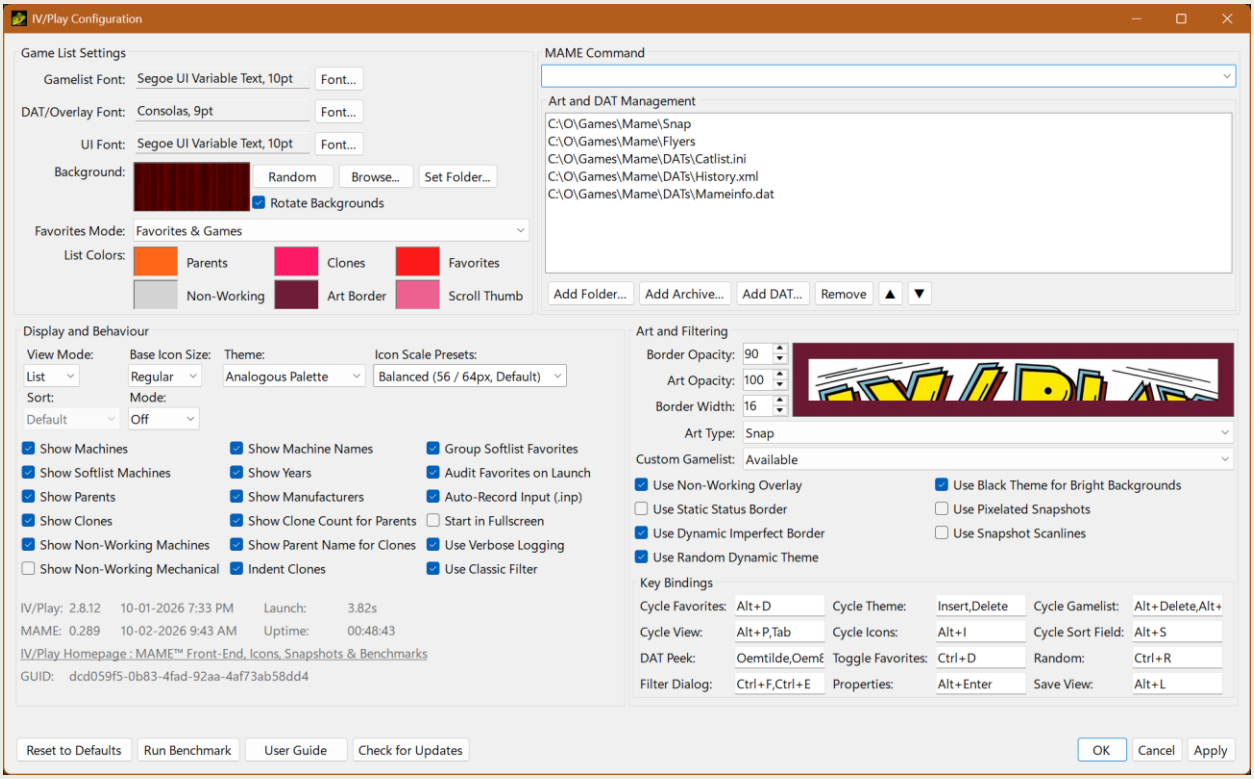
IV/Play is, at its core, a game launcher. It imports XML from the linked MAME or HBMAME executable to populate its lists and properties, then passes back the selected game's launch command. Configure the emulator in its own ini files: mame.ini for MAME, hbmame.ini for HBMAME. The F1 command-line override can supersede those settings for ordinary game launches.

### IMPORT XML DATA FROM MAME.EXE

IV/Play resides in the same directory as the MAME executable or in its own directory. Upon initial launch, it searches the local directory for a MAME version. Failing to find one a dialogue box appears asking to locate the MAME.exe to 'link'. It then grabs the output of the (currently) 296MB MAME XML file (MAME.exe –listxml). **F5** refreshes the XML using the same named MAME.exe, (when a new version of MAME is released for example) while **F4** re-opens the dialogue box to select a different MAMExx.exe.

### CONFIGURATION SCREEN (F1)

F1 brings up the configuration screen. The UI settings are stored in the user-editable IV-Play.cfg file, which may be viewed in-session with F3. On smaller desktops, settings scroll horizontally or vertically as needed while the footer buttons remain visible. F1 keeps the game highlighted when it opened, provided that game remains in the resulting list. If icon work is already running, F1 waits for it before opening.




## DISPLAY XML DATA IN UI

### TITLE BAR

The title bar displays the version of MAME linked along with the number of games supported in that version and favorites in use. The number of favorites displayed will change if the favorites and filter are used.

### GAME LIST

The game list displays all the games from the MAME XML output. Clones are automatically offset from their parents (but can have the indent removed in the **F1** screen). The font is selectable as is the color of the parents, clones, non-working, and favorites. Navigation through the list is via regular direction keys like **up**, **down**, **page up page down, home and end**.

Display of the year, manufacturer, and machine (internal MAME) name is configurable through **F1**.

Free typing ‘donkey kong’ in the game list goes to the first instance depending where focus is when typing; if there are no instances downward it will go to the first instance of ‘donkey kong’ which may wrap around to the top in the favorites.

Alt-P or TAB cycles through Regular List, Large List, Flat View, Grid with art, and Grid without art. Shift-TAB cycles backwards. If the cycle starts with art set to None, it skips the grid-with-art step. Flat View supports global sort; see the 2.6 feature section.

## FAVORITES

IV/Play supports the use of a .\favorites.ini file. This is a text file with one game per line, for example:

10yard
1941
dkong
frogger

Favorites are toggled by Alt-D and can be set in F1. They appear above the MAME game list in alphabetical order, without clone icon indentation, and can use separate colors. With at least one favorite, Alt-D cycles through Games Only, Favorites & Games, and Favorites Only. With no favorites, Games Only is the sole available mode; **Ctrl-D** adds the selected game. Removing the last favorite, including by editing favorites.ini externally, returns the view to Games Only. A filter with no matching favorites can still produce an empty Favorites Only result when the collection itself is non-empty. Non-working and clone favorites survive the visibility filters applied to the lower game list.

Favorites are added by **Ctrl-D** from the game list and removed by **Ctrl-D** from the favorites area.

**Up**/**Down**/**Left**/**Right**/**Page Up**/**Page Down** behave the same way they do in the game list. A single press of **down arrow** on the last favorite goes to the first regular game list item. **Home** and **End** buttons on the keyboard provide a two-stop navigation system to move between the favorites and main game lists efficiently.

- **End Key**: When the focus is in the favorites list, the first press of **End** jumps to the last favorite. A second press jumps to the very end of the main game list. If the focus is already in the main list, it jumps directly to the end.
- **Home Key**: When the focus is in the main game list, the first press of **Home** jumps to the top of the main list (directly below favorites). A second press jumps to the top of the favorites list. If the focus is already in the favorites, it jumps directly to the top.

## ICONS

Each game supported by MAME can have an associated, freestanding *.ico in the .\icons directory. IV/Play also supports having all the icons in \icons\icons.zip or \icons\icons.7z.

Base icon size and the icon scale presets can be chosen with the drop box in F1. Adding, changing or removing an Icons folder/archive in F1 takes effect on OK; F5 is not required to adopt the new icon source.

If ‘Draw non-working overlay’ is toggled on in the configuration menu, a small red circle ‘x’ will be overlaid on any non-working game in the lower right-hand corner of its icon.

## ART AREA

The upper right of the game list is the art area. The art area displays the current art type for the selected game. These are freestanding *.png files that reside in their respective directories and take the name of each game, e.g. dkong.png. IV/Play also support archived art assets so \snap\snap.zip or \snap\snap.7z are viable options.

Alt-1 through Alt-0 selects an art type; None is Alt-0. Add picture-art folders or archives in F1. If no picture art was available, adding the first one selects it automatically (Snap preferred); a None deliberately selected while picture art is available is kept. Sorting or cleaning the art list retains the selected folder. Removing the selected art falls back to Snap, then another picture-art entry, then None.

Opacity for the art types and art border is set in **F1** configuration dialogue.

The border width and color for the art can be set with **F1**.

Left clicking on the art area with the cursor toggles the art type.

If a game is displayed for which there is no parent snapshot (for example after a new version of MAME is released and linked, prior to a snapshot pack being created), then nothing displays in the art area, the bkground.png will show through.

IV/Play supports History.XML and MAMEInfo.DAT support files available at their respective sites. They are searched for in the MAME.exe directory structure on initial IV/Play launch and can be added at any time through the **F1** configuration menu. These files' information is accessible with the 'Peek' option using the tilde key (~) above the tab on most keyboards. Left and right in DAT peek mode sizes the peek overlay horizontally, the **TAB** key cycles through the DAT types while the peek overlay is open.

Snapshots and other art assets are placed in the upper right of the window offset from the edge, and are scaled to the height of the window. If art is wider than 40% of the horizontal width of the window, it will be scaled to fit. Border width, color, and its opacity are set in the **F1** screen.

## TEXT FILTER DIALOGUE

The Text Filter dialogue box is accessible with either **Ctrl-F** or **Ctrl-E**, and is extremely flexible. When typing in this field and hitting enter it will search and display resulting hits, like a filter. For example, typing 'donk' would bring up Donkey Kong and all other items with 'donk' in its description including something like 'Crazy Donkey'.

The filter can also take manufacturer, year, driver source for the game, i.e. 'dkong.cpp', and other key elements. Filtered game list is maintained until the filter is cleared. The filter includes favorites if toggled on. The dialogue can be dismissed with **ESC**. Filtered result counts are displayed on the title bar. A filtered view may be backed out of with **ESC**. The filter can also take powerful advanced searches and gives some examples in its drop down. It is able to form these advanced searches based on elements imported from the MAME XML. See **Appendix 1** for additional filter syntax and information on natural language 'aliases' for some common search topics. **Note**: The filter dialogue opens a tiered Quick Start dropdown automatically when the field is empty — 12 curated filters appear immediately, with the full library accessible via Alt-Down or the hint line. Down arrow or clicking the blank filter area also opens it. The most recent user filter appears above the Quick Start entries. To disable the two-step discovery type filter dialogue, chose the 'Use Classic Filter' in **F1**.

## BACKGROUND SELECTION AND RANDOMIZATION

The bottom art layer of the main list area takes a user defined background *.png image from the .\bkground\ directory. The **F1** configuration dialogue toggles the background image rotation, so IV/Play will use a random *.png from the .\bkground directory on each launch. **Note**: If the bkground image is 'larger' than current game

window it will scale to fit the inner window. If the bkground image is smaller than current game window and less than 903px wide and 800 tall, it will tile from upper left. These bounds can be altered in the *.cfg.

## PROPERTIES VIEW

**Alt-ENTER** displays the properties dialogue for the selected game. It can be dismissed with the **ESC** key. The text in each field is highlight-able and can be copied.

## LAUNCH GAME

Launching a game via Enter or double-clicking sends the simple argument 'MAME.exe dkong' to the linked MAME.exe. The configuration for MAME is done using its own *.ini files. Holding down the **Ctrl** key and hitting enter will launch the game and run its last recorded *.inp file if found. A regular launch records an *.inp of the game session only when Auto-Record Input (.inp) is turned on in F1; it is off by default. Holding down **SHIFT** and hitting enter on a software list machine will launch that machine as if it had no cartridge inserted, like an Atari 2600 or Intellivision.

## ACCELERATOR / NAVIGATION KEYS

The following are the keyboard shortcuts in IV/Play:

| | |
|---|---|
| **~ (Tilde)** | Toggles the DAT peek feature. Use with **Ctrl**+left and right arrows to size from 40 to 75 and 100 percent UI width. **Ctrl** plus Up, down, Page Up/Down, Home, and End can be used to scroll, as is common in the other text areas of the app. The **TAB** key will move to the next DAT file and then loop. |
| **Alt-~ (Alt-Tilde)** | Toggle persistent DAT peek. Regular game-list navigation changes the selection; Ctrl plus navigation keys scrolls the peek text. |
| **Alt-1 through Alt-0** | Go directly to art type. (left click on art area) Note: 'None' is always 0. |
| **Alt-D** | Cycle Games Only, Favorites & Games, and Favorites Only when favorites exist. With none, remain on Games Only. |
| **Alt-Enter** | Display properties for the game selected. |
| **Alt-I** | Cycle through the icon display sizing presets. |
| **Alt-Insert / Delete** | Cycles through Custom Game Lists (e.g. Full, Arcade, User Lists) |
| **Alt-L** | Save the current visible list as a new Custom Gamelist entry. |
| **Alt-Left Arrow** | Exits a DAT overlay, software list view, or clears an active filter. |
| **Alt-O** | Select a new random bkground. |
| **Alt-P / TAB** | Toggle through the different game list views. (right click on UI) |

| | |
|---|---|
| **Alt-S** | Cycles through the main 5 sort options while in flat view. |
| **Arrow Left / Right** | Navigate one letter's worth of games, from A to B to C, etc. through favorites and down through the main game list eventually looping to the top again. |
| **Arrow Up / Down** | Navigate one game at a time. |
| **Ctrl-B** | Benchmark the selected machine or software item with its actual host/media launch configuration; default is 90 emulated seconds. |
| **Ctrl-D** | Add or remove a favorite from favorites.ini. |
| **Ctrl-Enter** | Launch the inp of a game that has been previously recorded. (games launched normally are recorded only when Auto-Record Input (.inp) is turned on in F1; it is off by default) |
| **Ctrl-F / Ctrl-E** | Open text filter dialogue. |
| **Ctrl-R** | Select a random game. |
| **Ctrl-Shift-0** | Re-center and reset the dimensions to default without touching any other settings. |
| **Ctrl-Shift-C** | Copies the contents of the filter results list to the clip board in tab separated, Excel friendly format. |
| **Ctrl-Shift-L** | Cycle command line history in F1. |
| **F1** | Display the configuration dialogue. |
| **F10** | Internal tool for Build Verification Test. |
| **F11** | Toggles full screen mode, removes title bar and window frames. |
| **F12** | Build the Available list from top-level ROM archive names on the linked MAME/HBMAME rompath. |
| **F2** | Launches the IV-Play.log overlay. |
| **F3** | Launches the IV-Play.cfg overlay. |
| **F4** | Select the MAME.exe. |
| **F5** | Refresh the XML import from MAME.exe and rebuild icon / art caches. |
| **F6** | Internal tool for art asset auditing. |
| **F7** | Toggle the onscreen diagnostic dashboard for FPS, memory usage for CPU and GPU, and garbage collection. |

| | |
|---|---|
| **F8** | Factory reset removes generated assets and resets cfg settings after confirmation. Favorites.ini, IV-Play.Custom.ini, Hidden.ini, HiScore.ini and benchmark histories remain beside the executable. Use F12 if Available prompts for a new audit. |
| **F9** | Run the configured BigBench suite (default 36 games, 90 emulated seconds each), show its report in the benchmark overlay and save a text report to the desktop. |
| **Home / End** | Move to the beginning / end of the game list, or as a stop at the beginning and end of the favorites (press home/end a second time to navigate out of favorites). |
| **Ins / Del** | Cycle through the dynamic themes while in the game list. |
| **Page Up / Page Down** | Navigate one window height worth of games. |
| **Shift-1-9-Enter** | Launches the selected game in N concurrent tiled windowed instances. |
| **Shift-Alt-S** | Cycle sort direction in Flat View. |
| **Shift-Enter** | Launch a softlist machine without entering its softlist. (effectively like powering on a console with no cartridge in it) |
| **Shift-F8** | Display a grid of all the IV/Play splash screens. |

## EXTERNAL FILES

IV/Play.exe can reside in its own folder or along with MAME.exe. It also now creates a sub-folder for its asset files for easy rebuilds. It's recommended to keep the *.inis at the same level as the .exe so they survive directory purges.

C:\games\IV-Play\IV-Play.exe (main application)
C:\games\IV-Play\IV-Play User Guide.pdf (this document)
C:\games\IV-Play\e_sqlite3.dll (required SQLite native library)
C:\games\IV-Play\Favorites.ini (user favorites list)
C:\games\IV-Play\IV-Play.Custom.ini (custom game list definitions)
C:\games\IV-Play\Hidden.ini (persistent machine/category/driver exclusions)
C:\games\IV-Play\IV-Play.cfg (application configuration)
C:\games\IV-Play\HiScore.ini (optional high score data for the art area overlay)
C:\games\IV-Play\IV-Play.Bench.History.txt (benchmark run history, one row per run)
C:\games\IV-Play\IV-Play.Bench.Machines.txt (benchmark history, one row per machine per run)

C:\games\IV-Play\IV-Play\IV-Play.available.bin (current results of F12 available ROMs audit)
C:\games\IV-Play\IV-Play\IV-Play.background.state.json (background rotation tracking)
C:\games\IV-Play\IV-Play\IV-Play.db (SQLite game database)
C:\games\IV-Play\IV-Play\IV-Play.db.meta.json (database schema metadata)
C:\games\IV-Play\IV-Play\IV-Play.history.cache (parsed History.XML binary cache)

C:\games\IV-Play\IV-Play\IV-Play.icon.atlas.bin (raw binary icon texture atlas)
C:\games\IV-Play\IV-Play\IV-Play.icon.map.json (icon atlas coordinate index)
C:\games\IV-Play\IV-Play\IV-Play.icon.atlas.hash (atlas validity checksum)
C:\games\IV-Play\IV-Play\IV-Play.log.txt (diagnostic log)
C:\games\IV-Play\IV-Play\IV-Play.mameinfo.cache (parsed MAMEInfo.DAT binary cache)
C:\games\IV-Play\IV-Play\IV-Play.SoftListFavorites.cache (softlist favorites binary cache)
C:\games\IV-Play\IV-Play\IV-Play.SoftlistIndex.cache (140K-item softlist binary cache)
C:\games\IV-Play\IV-Play\Bench\cfg\ (benchmark-only MAME cfg state)

Once linked to a MAME.exe, IV/Play will use its structure for its art, for example:
C:\games\MAME\MAME.exe
C:\games\MAME\History.XML
C:\games\MAME\MAMEInfo.DAT
C:\games\MAME\INI\MAME.ini
C:\games\MAME\bkground*bkground*.png
C:\games\MAME\icons*gamename*.ico
C:\games\MAME\snap*gamename*.png
C:\games\MAME\flyers*gamename*.png
C:\games\MAME\etc\etc.png

### IV-PLAY.CFG

This file is created on first launch of IV/Play if it is not present. It contains the configuration settings for the application. Delete this file (and the IV-Play sub-directory) to start fresh. Note that the *.cfg file can point to a custom asset directory (asset_directory), which is helpful if a RAM drive or similar technology is being employed for storage.

### IV-PLAY.DB

The SQLite database is created from the linked emulator's XML on first run, or rebuilt when the executable or required schema changes. IV-Play.db and its metadata file live in the configured asset directory (by default the IV-Play subdirectory beside the application). F5 refreshes the linked executable's XML and caches; F4 links a different executable. Both retain the Full or Available list and restore the selected game when it still exists. Required availability repair completes before the Available list is displayed. The matching e_sqlite3.dll must remain beside IV-Play.exe.

## UPDATES

### IV/PLAY 1.5 FEATURES

#### UNLIMITED ART TYPES

IV/Play supports multiple art types. In F1, add directories or archives containing images named gamename.png. On initial launch it probes for snap, flyers, cabinets, PCB, marquees, cpanel, titles, bosses, ends, gameover, logo, scores, select, versus, howto, warning, and artpreview, including their respective .zip or non-solid .7z packs. F1

allows the list to be reordered; startup and F5 normalize it with Snap first. The selected art folder is retained through that normalization. Art shortcuts depend on the current order (Alt-1 through Alt-9, with Alt-0 for None).

## HISTORY.XML / MAMEINFO.DAT SUPPORT

IV/Play supports History.XML and MAMEInfo.DAT. They are searched for in the MAME.exe directory on initial IV/Play launch and can be added at any time through the **F1** configuration menu. Activation of the view is via the tilde ~ key which opens the DAT 'peek' view. The peek overlay is expanded left or right to multiple sizes and navigation is via the regular arrow keys, **page up** and **down**, and **Ctrl-End** and **Ctrl-Home**. A separate font with size and color is selectable for the text area on the **F1** configuration menu.

## FAVORITES TOGGLE

With favorites present, Alt-D cycles through Games Only, Favorites & Games, and Favorites Only, including software-list views and filter results. With zero total favorites, only Games Only is available; add a favorite with **Ctrl-D** to enable the other modes.

## NON-WORKING GAME ART/HISTORY SUPPORT

IV/Play will display icons and any art type for non-working games if they are present. This also allows viewing of History.XML and MAMEInfo.DAT entries for non-working games.

## COMMAND LINE OVERRIDE

The command line override in the **F1** configuration dialogue allows the use of various switches to be added to launched games, e.g. –window to play games windowed without having to drop to the mame.ini for editing. Note that command line arguments take precedence over MAME's *.ini processing hierarchy. The command line is now a drop down and has a well with the last 5 commands remembered. The command lines can also be cycled through with **Ctrl-Shift-L** and will complain if the command does not begin with MAME's expected dash (-) prepending the arguments.

# IV/PLAY 1.6 FEATURES

Updated to work with new XML output of MAME .162.

# IV/PLAY 1.7 FEATURES

## CLONE AND NON-WORKING FILTERS

IV/Play now has the ability to filter out clones and/or non-working machines in the **F1** configuration screen.

# IV/PLAY 1.8 FEATURES

- New background and modern Windows Segoe UI font as default on first run.
- ESC will exit IV/Play.

- It is now possible to un-indent the clones in **F1**.
- Relative paths are now used in the IV-Play.cfg file, this allows for easy portability/transport via thumb drive or removable storage.
- With 1.8.4.0 added support for the new format of the History file, now an *.XML at [www.arcade-history.com](http://www.arcade-history.com).
- With 1.8.5.0 added the ability to change the color of the non-working games/devices in **F1**.

## IV/PLAY 2.0 FEATURES

After 9 years in maintenance mode, IV/Play sees a significant modernization effort, rebuild, major feature additions, and a full version bump in the summer/fall of 2025.

## MOVE FROM .NET 4.5 TO .NET 8.0 & EXIT FROM WPFWRAPPER TO PURE WINFORM

A migration was undertaken successfully to get IV/Play from .NET 4.5 all the way to 8.0 and also to extricate it from the WPFWrapper which had ultimately caused bothersome flickering of the handoff on launch.

## GPU RENDER PIPELINE

IV/Play has been converted from a GDI app to utilising a full GPU accelerated pipeline. This leads to considerably smoother scrolling and navigation in-app with art assets appearing instantly from its various caches. Art, icons and backgrounds scale via DirectX High Quality Bicubic instead of straight bilinear filtering, resulting in sharper images on higher resolution monitors.

This architectural change was a strategic decision to resolve a persistent and severe performance bug known as "scrolling judder". The previous GDI+ rendering pipeline was CPU-bound, meaning all drawing operations were handled by the main processor. During fast navigation, this would overwhelm the UI thread, causing the application to stutter or freeze.

The new pipeline offloads all rendering to the system's graphics card (GPU) using DirectX 11 / Direct2D. This frees the CPU from heavy drawing tasks, resulting in a perfectly smooth, zero-latency user experience, even when scrolling rapidly through thousands of games with high-resolution artwork. This one-time processing of assets on startup is an intentional trade-off that enables the application's highly responsive and fluid feel during use.

## LITEDB TO SQLITE

IV/Play's data layer was migrated from LiteDB to SQLite during the modernization. The initial migration used Entity Framework Core; that layer has since been retired, and current database reads and writes use raw ADO.NET.

This was a critical architectural change driven by a series of persistent and difficult-to-diagnose crashes originating from the LiteDB query engine. The migration to a mature, industry-standard backend like SQLite completely resolved these stability issues.

## SUPPORT FOR HIGH DPI MONITORS / LARGE ICON MODE / SCALING IMPROVEMENTS

- IV/Play is now high DPI aware; it won't be necessary to use Windows 11 compatibility hacks to enable a non-miniscule screen. The UI is now at 1523x648 which is proportionally what it was at 1015x432 on 1080p monitors. Running at 144 DPI and 4K now looks very nice; the fonts are crisp and each art asset looks the way it should instead of being simply stretched after a Windows snapshot. Everything scales properly based on the DPI setting including row height.
- Large icon view is now supported, it can be toggled in the **F1** dialogue and via **Alt-P** or **TAB** as one of the views in the cycle.
- Backgrounds will now scale themselves to the inner window size, a set size floor for that functionality can be altered in the *.cfg so anything smaller than the setting will be tiled. This works particularly well for aspect ratio friendly bkgrounds that may be slightly bigger or smaller in their dimensions than the current inner window.
- Long time bug fixed for maintaining a maximized state on re-launch.
- IV/Play no longer locks its assets with file handles, so the snaps viewed in that session can be replaced on disk underneath the app.

## SUPPORT FOR SOFTWARE LIST DEVICES

IV/Play can now click/enter on a console like Atari 2600 and it will read its softlist and display as the new game list. The navigation is the same as the full game list. Hit **Backspace** or **ESC** or **Alt-Left-Arrow** to return to the level above with highlight on the system entered. The favorites work the same way in the softlist view, adding or removing them with **Ctrl-D**, or toggling their views with **Alt-D**. Use **Shift-Enter** to launch a softlist machine directly with no software/cart attached, effectively like powering on an Atari 2600 with no cartridge in the slot.

## CUSTOM GAME LIST SUPPORT

Similar to the favorites.ini, IV/Play now allows the use of the IV-Play.Custom.ini file, selectable from the **F1** dialogue. If chosen, this file takes over the game list and displays entries contained within. This is a single file, but it can be nested based on [] or <> heading delimiters that will produce the drop-down choice.

#Custom GameLists
<Golden Age>
amidar
armora
astdelux
asteroid

<Fighters>
sf2
sfa3

## GRID VIEW

Grid View displays games with descriptive labels under their icons. Navigation follows the game-list model. Alt-P or TAB cycles through Regular List, Large List, Flat View, Grid with art, and Grid without art; Shift-TAB reverses the cycle. If art is already None, the grid-with-art step is skipped.

## BACKGROUND BRIGHTNESS DETECTION MODE

IV-Play can now automatically determine if the bkground image is of a very light/bright variety and adjust its fonts to black accordingly (which can still be overridden in the **F1** config).

## IMAGE ATLAS FOR ICONS AND BINARY VERSIONS OF HISTORY.XML AND MAMEINFO.DAT

To maximize performance, IV/Play combines all individual game icons into a single large image file called an "icon atlas". This atlas is loaded into GPU memory on startup, allowing for extremely fast icon display and smooth scrolling by eliminating the need to load thousands of individual files from the disk.

To accelerate startup, IV/Play creates optimized cache files for History.XML and MAMEInfo.DAT the first time they are used. On all subsequent launches, the application reads from these pre-parsed files, dramatically reducing load times by avoiding the slow process of reading the large, original text files.

## ADDITIONAL FEATURES

- IV/Play.cfg is now alphabetized and sectionalized.
- Via the auto-record **F1** toggle, machines launched will have their session *.inp recorded. To play it back, launch the same game a second time with the **Ctrl** key held down.
- Nearest neighbor scaling is an option now in **F1** (unsmoothed snapshots); it allows a more pixelated scaling algorithm that lends a retro ‘air’ to the snaps.
- A simulated scanline effect can now be applied to snaps, both over the pixelated and the smooth versions.
- Clone icons and the focus/highlight are now transparent and controlled via settings in the *.cfg file.
- On a fresh 96-DPI desktop, IV/Play uses its historical 1015x432 dimensions and adjusts icon and row sizes. Current builds also rescale saved geometry when an installation moves between monitor DPIs; deleting the cfg is not required just to move from a 4K system to a 96-DPI display. **Ctrl-Shift-0** re-centers the window and restores default dimensions without changing other settings.

# IV/PLAY 2.1 FEATURES

## MOVE FROM .NET 8 TO .NET 9

Upgraded the project from .NET 8 to .NET 9 to benefit from the latest runtime performance enhancements, contributing to a more responsive UI and faster icon cache processing.

## ADDITIONAL FEATURES

- Tapping the **Tilde (~)** key will open the DAT ‘peek’ view where the DAT files are accessible in a temp window. **Ctrl** plus Left and right size the window to 40%, 75%, and full width of the UI. **TAB** cycles through the DATs.
- Copy out of Filter, create a filter search and then **Ctrl-Shift-C** to copy the results to the clipboard in a tabbed, Excel friendly output.
- **F2** (log) and **F3** (config) now activate their ‘peek’ overlays with navigation the same as the DAT peek.

- **F8** will do a 'factory reset' and remove the contents of the \IV-Play directory including the *.db and art caches. It also removes the *.cfg file, but leaves the favorites.ini and custom game list ini.
- Aspect ratio constrained sizing: If **Ctrl** is held down during a sizing event the UI will scale up or down in 2.35:1 TohoScope.
- **F7** displays the performance dashboard/overlay in the upper right corner giving information on memory usage for the system, GPU, FPS, and garbage collection data.
- Distributable is now packaged with AOT (Ahead-of-Time) compilation during publication (vs. JIT) which speeds the initial bring-up of the app.
- **F10** runs a short BVT (Build Verification Test) used in development to exercise basic functionality for breakage.
- Added a 'Theme' drop down to **F1** which allows a single switch for multiple color combos on the various game list entries.
- Added ability to launch a softlist machine on its own, e.g. 'mame.exe a2600', instead of going into its softlist view. Hold the **Shift** key down while hitting enter or double clicking on the machine. This is akin to starting a console with no cart in it.
- Reintroduced an **F1** toggle to hide non-working mechanical machines, removing ~15K entries from the game list; this is now on by default.
- Added a drop down in **F1** for icon scale presets based on monitor resolutions and DPI combos. This controls the scaled size of the regular and large icon display on said monitors.
- **Alt-I** in the game list will cycle through the various icon scale presets.
- **F11** will now toggle into and out of full screen mode, hiding the window borders and title bar; similar to **F11** in browsers. **ESC** will also exit full screen mode. A default setting for this is user configurable in the *.cfg file in the window section.

# IV/PLAY 2.2 FEATURES

## VERTICAL SCROLLBAR FOR GAME LIST

A custom vertical scrollbar control has been added to the right of the art area.

## FILTER IMPROVEMENTS AND ADDITIONS

The **Ctrl-F/E** filter has been overhauled with a clearer presentation and expanded functionality. Key features are now surfaced in the drop-down example list. Opening the filter on an empty field now shows a **Quick Start Filters** preview tier: 12 high-value curated filters shown immediately, with the full library accessible via Alt-Down or by clicking the hint line at the bottom of the list. Natural language terms such as *80s games*, *working only*, or *no clones* can be entered directly, while the full SQL-like advanced syntax remains available for precise control with field-specific searches (e.g., year:1994), negation (manufacturer:!Capcom), and complex Boolean logic. New hybrid queries now allow natural language keywords to be combined with advanced operators in a single expression, enabling flexible searches such as Year >= "1995" noclones for parent games from 1995 or later, manufacturer:Capcom working for working Capcom titles, or description LIKE "%puzzle%" nonworking parents for non-working parent puzzle games. This hybrid approach bridges simplicity and precision, giving both casual and advanced users greater control over filtering results.

## ADDITIONAL FEATURES

- IV/Play's version information is now surfaced on the title bar after its name before the linked MAME .exe.
- The DAT Peek feature's key is now configurable; especially handy for non-US keyboard layouts. The default will try to find the key in the upper left above the **TAB** key and should handle most European keyboards. This key can be set in the IV-Play.cfg file. The various key codes may be found on Microsoft's site by searching for System.Windows.Forms.Keys enumeration in their .NET Framework docs.
- The **F1** config check boxes have been re-ordered and the verb has standardized on 'Show' vs. a mix of show and hide. Attendant logic has been updated for the new verbiage.
- The Grid View + No Art combo has been added to the **TAB** / **Alt-P** cycle as the fourth option.
- The diagnostic **F10** Build Verification Test (BVT) has been extended with a new overlay and progress screen with grid view interactions.

## IV/PLAY 2.3 FEATURES

### REINTRODUCED LEGACY ERROR DETECTION

Reintroduced the legacy IV/Play (1.8.5) feature that detects and reports MAME errors when attempting to run a machine without the required ROMs. This is displayed via one of the overlays and allows a copy and paste out of.

### BIGBENCH BENCHMARK SCRIPT

Added John IV's *BigBench* benchmark script. Launchable via **F9** (and dedicated **F1** button), it runs 36 games in succession, measuring performance over 90 (emulated) second intervals. Monitoring overhead is negligible, and results are statistically equivalent to those obtained from a direct command-line run. Results are displayed in an overlay and also saved as a text file on the user's desktop. Both the game list and the -bench (time) value are configurable in the *.cfg file. See the benchmark page at the IV/Play site for historical data: https://john-iv.github.io/iv-play/bench.html. Note that benchmark results for each new MAME release are typically posted monthly on the MAMEWorld News forum. The geometric mean value is the current preferred measure of PC / MAME performance across disparate hardware.

### DYNAMIC COLOR THEMES

Added dynamic theme options in addition to the static presets accessible in the **F1** config. Dynamic themes calculate text and UI colors based on the dominant background image color and its brightness. This allows the interface to adapt automatically to different backgrounds while maintaining legibility and acts on the fonts, scrollbar thumb, and art border. **Insert** and **Delete** cycle through the dynamic themes while in the game list.

- **Brightness Ladder** Adjusts text lightness relative to the background. On darker backgrounds, text colors are brightened in stepped increments to ensure readability. On lighter backgrounds, text colors are darkened to maintain contrast.

- **Saturation Gradient** Maintains a fixed lightness level but varies saturation. Favorites are rendered with higher saturation, clones with reduced saturation, and non-working entries in grayscale. This produces a consistent brightness level while distinguishing categories by color intensity.

- **Analogous Palette** Selects colors close to the background's hue, shifted slightly on the color wheel. A minimum lightness threshold is enforced to prevent text from becoming too dark. This produces a cohesive palette aligned with the background color.

- **Dual Contrast** Applies both saturation and brightness adjustments to create stronger separation between categories. Favorites and parents are shifted lighter and more saturated, while clones are darker and less saturated. Non-working entries are rendered in grayscale with a minimum brightness floor to ensure visibility.

- **Miami Pastels** Selects a vibrant, high-saturation color scheme using complementary pastel hues, typically teal and pink. Colors are assigned based on the background's general warmth (warm background triggers cool pastel text, and vice versa). A fixed lightness level is targeted to maintain the signature pastel aesthetic, with slight adjustments made for very bright backgrounds. This theme overrides the hue of the background completely to apply a distinct, stylized color set.

## EXPANDED KEY CUSTOMIZATION

Users can now modify several keyboard shortcuts to better suit their preferences. These customizations are made directly within the IV-Play.cfg configuration file. Look for the Key Bindings section near the top of the file; it includes comments explaining the syntax and providing examples of valid key combinations. Formalized the Euro keyboard version of the 'Tilde Peek' key, which is Oem8 and made it a standard alias.

## FAVORITES.INI AUDIT

Toggled on by default in the *.cfg, this feature will audit the favorites.ini file on initial launch looking at each line. If it finds something it doesn't recognize it will prepend an asterisk to the entry; to surface it for manual editing and removal. It also displays an overlay with this information in-app. This feature helps keep track of occasional renames of machines that occur in baseline MAME .exe, and also clean up accidental misspelled manual entries to the favorites.ini.

## ADDITIONAL FEATURES

- The scrollbar thumb color can now be customized in the F1 configuration screen. The active scrollbar thumb color may also be set manually in the *.cfg file. If left at the default, IV/Play will automatically adjust the thumb color based on background brightness, selecting either a dark or light variant for optimal visibility. The scrollbar thumb also takes the dynamic themes colors if not user customized.
- The border opacity has been surfaced into a control on the **F1** config dialogue.
- Added a toggle for Random Dynamic Theme in **F1**. If checked, this will randomly cycle through the dynamic themes on launch, providing an attractive new look each session; especially when paired with randomizing the background.
- MAME and IV/Play versions now appear in the **F1** config form as well as a link to the home page.
- Benchmark button added to the **F1** config.
- The default short-cut keys will now display in the **F1** dialogue for easy reference.
- Added the ability to tweak which themes are available to the theme cycle and which are included in the randomization carousel in the *.cfg file.
- Added a button in **F1** to set the background images folder.

- **F6** launches an internal tool for icon and snap asset auditing.

## IV/PLAY 2.4 FEATURES

### SOFTWARE LIST FILTERING & FAVORITES

A large refactoring of the handling of software lists now allows for filtering/searching across the entire 100K plus collection of MAME software items. This means it's now possible to search for an item like name:Zaxxon and get the hits for Zaxxon across all of its ports. Software list items are also now eligible for inclusion to the favorites.ini file. Add or remove them with the same key **Ctrl-D**. They can be grouped at the end of the main game list favorites (sorted by machine) or integrated into the main list via descriptive name. The favorites audit will now extend to include these items, so a2600/combak vs. a2600/combat will be flagged.

### F1 CUSTOMIZABLE KEYS SECTION

The main alterable key combos have been surfaced into the **F1** dialogue. All the remaining keys are editable in the IV-Play.cfg file.

### THE ARCADE GAMELIST

IV/Play now has a curated, subjective, arcade-only game list; accessible via **Alt-INS** / **Alt-DEL** cycle keys or **F1** in the custom game list drop down. Excludes all the non-coin-op, mahjong, fruit, gambling, bar top, quiz, pinball, redemption, mechanical, and BIOS machines. With clones and non-working filtered out it's about 3400 games.

### SUPPORT FOR ZIP AND .7Z ART PACKS

Added the ability to use \snap\Snap.zip|.7z and \icons\Icons.zip|.7z and the other art types, cabinets, flyers, marquees, etc. **Note**: 7z solid archives are not supported due to performance degradation, the app will warn if it finds one in its asset search. The Snap.7z and Icons.7z from the IV/Play Homepage can now simply be dropped in to the \icons and \snap directories for direct use.

### MULTIPLE MAME LAUNCHES AND TILING

IV/Play now allows for using a single session of itself to launch multiple windowed instances of MAME; selecting either different games or the same game multiple times for multi-player gaming on a single screen. Use **Shift-1** through **Shift-9** while launching a game with **ENTER** or double-click to create that many windows. This allows for playing networking supported games in MAME like linked Cruisn' USA cabinets or similar. **Note**: Only the first instance gets the sound, to prevent the cacophony of all of them outputting at once. The first windowed game also takes the mouse vs. the others for easier navigation and windowed placement, as desired.

### MOVE FROM .NET 9 TO .NET 10

Upgraded the project from .NET 9 to .NET 10 to take advantage of long-term support and further runtime optimizations, delivering faster startup responsiveness, smoother UI interactions, and improved memory efficiency for a more stable user experience.

## ADDITIONAL FEATURES

- The art border size and the opacity settings now have a preview box next to them to visually balance out some of the other additions to **F1**.
- Verbose logging is now a toggleable option in **F1**.
- IV/Play's start-up time is now displayed on the **F1**, this includes time to an interactive UI and the final completion of background threads.
- Added a 5-layer history drop-down to the MAME command-line override combo box in **F1**, with the ability to assign a shortcut key to cycle, currently **Ctrl-Shift-L**, tweak in the *.cfg key mappings section.
- Added a 'User Guide' button to the F1, this will launch this document if it is in the same directory as the .exe.
- An individual machine can now be benchmarked; use **Ctrl-B** on a selected item to get the same result the full bench suite produces. The duration is the default 90 emulated seconds. This can be adjusted in the *.cfg file as desired.
- Allow a nonworking machine to display its own art asset if there is one.
- Prevent snap scanlines on vector games and non-MAME generated art (flyers, marquees, PCBs, etc.)
- Allow saving off the random bkground tracking json file before a full factory reset.
- Allow DAT peek with art type 'None'.
- Auto-populate paths for the current common art types if present in the MAME.exe directory structure.
- Added Record INPs and Launch in Fullscreen as toggles in **F1**.
- The app now hides itself when launching MAME.
- Reworked the status line on the game Properties view (Alt-Enter) to provide more detailed and clearer information presentation.
- The F4 executable picker defaults to all files, making differently named MAME/HBMAME builds easier to link. IV/Play's supported platform remains native Windows.
- Added a toggle in **F1** to show or hide Softlist machines themselves.
- Added ability to search by softlist media types like cartridge, cassette, CD-ROM, etc.
- Added autocomplete with 'ghost text' look-ahead to the **Ctrl-F** filter dialogue.

# IV/PLAY 2.6 FEATURES

## FLAT VIEW & GLOBAL SORT

Added a new flat view which displays all machines alphabetically and non-indented; clones decoupled from their parents, favorites not pinned, parents at full opacity. Accessible via the **TAB** cycle or the **F1** View dropdown. Flat View is also the only mode that supports global sort, which orders the list by default, description, machine name, manufacturer, or year. The selected sort field persists through restarts; direction resets to its natural default. Sort cycles via **Alt-S** or the **F1** Sort dropdown; especially useful when a filter result contains thousands of entries.

## DEEPER DAT INTEGRATION FOR FILTER SEARCHES

History.xml, MAMEInfo.dat and CatList.ini participate in **Ctrl-F** searches. history: / hist: searches History.xml text for machines and software; mameinfo: / mi: searches the full MAMEInfo.dat text for machines, while mameversion: / mv: selects the first-appearance version. genre: / gg: searches CatList.ini categories and subgenres. These sources also support natural-language searches such as imperfect shooters from 1995. workingstatus: and

supportstatus: distinguish machine driver status and software support level. See Appendix 1 for syntax and examples.

## AVAILABLE ROMS GAMELIST

The Available gamelist, selected in F1 or via Alt-Insert/Delete, audits top-level .zip/.7z names on the linked emulator's rompath. IV/Play reads mame.ini for MAME or hbmame.ini for HBMAME along the emulator's inipath, honoring the earliest value for each setting and an inipath redirect. Relative paths are resolved from the emulator's directory. Without a configured rompath it uses the emulator's default roms directory. Detection reads names only, never file contents. A game is listed when its archive is present. In a merged collection, where clones have no archives of their own, a clone is listed from its parent's archive and a game also needs its BIOS archive. IV/Play works this out from which clone archives are present; when it cannot tell, it lists the game rather than demanding a separate BIOS archive, so a game can occasionally appear that still lacks a file. A board whose ROMs all belong to its BIOS or parent, such as the Neo Geo MV-1, is listed from that archive. A game that needs a CHD is listed only when a folder named after it or its parent, holding a .chd file, is on the rompath. A ROM or disk that MAME marks as never dumped is not required, so laserdisc games whose disc was never dumped are listed from their ROMs. Machines that need no file at all, such as Pong, are omitted. F12 builds the initial audit, after which live monitoring keeps it current as archives come and go; press F12 again after adding or removing CHDs.

## PLATFORM-BASED SOFTLIST FILTER GROUPING

When a filter (**Ctrl-F**) returns software list results, IV/Play now collapses them into platform-level family nodes rather than producing one row per software item per hardware variant. A search for 'zaxxon' produces a single 'Zaxxon (Atari 2600)' entry, a single 'Zaxxon (ColecoVision)' entry, a single 'Zaxxon (MSX1)' entry, and so on; one row per platform, regardless of how many hardware machine variants or regional releases exist for that platform in MAME. Pressing Enter on a family node drills down into it, showing the parent release and any clones (regional variants, alternate revisions) for that platform. The parent entry shows its clone count, and each clone shows its parent name in brackets. Press **Alt-Left Arrow**, **Backspace**, or **ESC** to return to the platform-level filter results. Adding 'noclones' to the filter suppresses clone entries before grouping runs, so only parent releases appear and drill-down shows a single launchable entry.

## NATIVE AOT TRANSITION / PERFORMANCE

IV/Play ships as a fully self-contained Native AOT executable and does not require a separate .NET 10 runtime installation. Native code generation avoids runtime JIT compilation during ordinary use; the distributable includes the required runtime support. The SQLite native library remains a separate e_sqlite3.dll beside the executable.

## MACHINE STATUS AWARE ICON BORDERS

This configuration feature surfaces the status of MAME machines that appear in **Alt-Enter** properties as not working (red), imperfect (orange), and working (green) by using the black border of the existing icons and changing them to that color (or any user chosen in the *.cfg). Further, since the imperfect games are the most in need of visual differentiation, they are now defaulted to a bkground derived themed color to complement the auto-theme feature. This auto color for the imperfect status border can be turned off in **F1** to return to user chosen RGB settings. It also extends to softlist machine media items.

## HIDDEN.INI

IV/Play now supports a hidden.ini file at the executable level for permanently suppressing entries from the game list before any filter or display logic runs. Each line takes one of three prefix forms: type: targets a machine category, machine: targets a specific short name, and driver: targets all machines sharing a source driver. Excluded entries are removed from the dataset entirely — they do not appear in filter results, favorites, or any custom gamelist. The file follows the same precedence rules as Favorites.ini and survives factory resets. It is generated automatically if not present and contains the syntax/usage information. Its creation can be suppressed by a toggle in the *.cfg if desired.

## CUSTOM GAME LIST GENERATION FROM FILTER RESULTS

Using **Alt-L** from a filter result set will now automatically create an entry in the custom-list.ini file.

## NUMBER OF CLONES / PARENT

Added the ability to show the number of clones a machine has on the gamelist, as well as the counterpart `[parent: x]` on clone entries. Clone counts use singular/plural forms ("1 clone" / "5 clones") and appear left-anchored immediately after the machine description. This can reveal interesting historical patterns where widely bootlegged games may have dozens of clones. Accessible in **F1** as two independent toggles.

## COMMAND LINE OVERRIDE AUTOCOMPLETE

Returning from IV/Play 1.8, MAME's `-showusage` output is captured and used for the command line override text input.

## ADDITIONAL FEATURES

- DPI detection and display has been hardened; IV/Play now flows better when dragged across multiple monitors with different DPIs and scaling settings.
- Added toggles for 'Show Parents' and 'Show Machines' to create even more display permutations. These can also be accessed via natural language queries in **Ctrl-F** such as "Parents" or "Machines Only."
- Added new randomized splash screens which appear on full MAME XML export and .db rebuilds at launch.
- Added **Shift-F8** to show all the new internal splash screens in an overlay panel.
- Added two new 3-way filter operators to **Ctrl-F**: workingstatus:working / :imperfect / :notworking (alias ws:) distinguishes fully-working from imperfect-emulation machines, going beyond the boolean working only phrase. supportstatus:yes / :partial / :no (alias ss:) filters softlist items by their MAME support level. Combine with explicit system scope for best results: system:snes supportstatus:yes. Natural language aliases such as imperfect machines, broken games, and partially supported are also supported.
- Added a 'snapshot' of the entire user configuration taken on launch and available in the **F2** log. If changes are made through the session, they are flagged by a color change and delta icon (Δ) on the next **F2** to make for easier debugging/diagnostic work.
- Added Genre, MAME version, and machine type to **Alt-Enter** properties since we consume all of that and hydrate it now. Requires History.xml, MAMEINFO.dat, and CatList.ini.

- Added a subtle configuration/**F1** button at the bottom of the vertical scrollbar. This is the first time in the app's history that the configuration dialogue is accessible without a keypress.
- Added current game list in use to the titlebar.
- Add **Alt-O** to select a new random bkground. **Note** shortcut keys are all assignable in the *.cfg and some major ones in the **F1** dialogue.
- Rename favorites and custom gamelist in the distribution packages so they don't overwrite users' own files.
- Added the ability to return the app to center of screen and its default dimensions. Double click the titlebar or use **Ctrl-Shift-0**.
- Added an OS aware System / Dark / Off mode drop-down in **F1**.
- Added a browser-like **Ctrl-F** (Find) inside the overlays, including **F2** & **F3**, and the DAT peeks.
- Modernized the font and color pickers in **F1**.
- Removed hardcoded Segoe UI usage throughout the rest of the app, main UI font now set in the **F1** config and will propagate. New default per Windows 11 is Segoe UI Variable Text.
- Created a TTF font based on MAME's very old uismall.bdf for use in MAME's internal UI. Double-click it, choose 'Install' to bring it into Windows 11, then use '-uifont MAMEUISmallMod' in the MAME.ini to utilise it, in-game. It also looks nice as the UI font in IV/Play itself, set in **F1**. Added to the distribution package.
- Added a third font picker field in **F1**, this for the rest of the UI, including dialogues. This allows a separate gamelist font setting to be decoupled and individualized.
- No longer including favorites.ini and the custom gamelist.ini in the distribution package so there is no accidental overwriting on decompressing the archive.

## IV/PLAY 2.8 FEATURES

### DPI & LAYOUT MODERNIZATION

IV/Play's DPI system has been rebuilt to eliminate the mix of pixel-based and DIP-based sizing that earlier versions relied on. All layout calculations now flow through a unified DPI-aware geometry model, so the interface scales cleanly on high-resolution displays and behaves correctly when moved between monitors with different scale factors. Window metrics, icon sizes, and hit-testing now update deterministically when DPI changes, preventing the fractional drift and inconsistent sizing that could occur before. The result is a sharper, more stable presentation on modern 4K and mixed-DPI setups without requiring any configuration changes from the user.

### INTERNAL HISCORE.INI

HiScore.ini sits beside IV-Play.exe. Use one entry per line: entry key, TAB, score or text, and optionally TAB plus a date. Machine keys are plain shortnames (pacman); software keys use the software-list name and item (a2600/pacman or adam_cass/buckrog), matching Favorites.ini even when the launch host has a different name. A software item never borrows a same-named arcade score. The overlay's position and opacity are configurable in the cfg. The file is loaded and sorted at startup, survives F8, and is not hot-reloaded; restart after editing it.

### MAME SOURCE .CPP ADDED TO DAT PEEK

Added a link to the MAME source to the DAT Peek feature. The DAT peek cycle will include the machine in question's XX.cpp file so it is easily accessible for informational purposes. MAME source path is set in the *.cfg source_root line.

## UNATTENDED BENCHMARKING (-BENCH)

Launching IV-Play.exe with -bench runs the configured suite without opening the UI, writes a report and exits. Games and duration come from IV-Play.cfg, as for F9. Runs are single-instance, isolate benchmark tuning from the emulator's ini settings while retaining necessary paths, and pin to performance cores on hybrid CPUs. Eligible runs append IV-Play.Bench.History.txt and IV-Play.Bench.Machines.txt beside the executable; these survive F8 and open in Excel. The report distinguishes history-write success, partial persistence and failure. Regression checks and the All-Time Overview compare matching host, launch path, duration and CPU model/logical-processor count; the overview also matches game count. See Appendix 3 for report interpretation and exit codes.

## ADDITIONAL FEATURES

- Added TohoScope Internal / External to the *.cfg. This is an amusing little feature that will alter the dimensions of the main UI to conform with TohoScope's 2.35:1 aspect ratio in two different ways. The current default method and the one that has been used since IV/Play's inception is 2.35:1 for the *entire* app including the Windows chrome. That always meant the internal window area was not TohoScope as a result. The new setting 'Internal' will now make the inner window area 2.35:1, increasing the total height of the app by the titlebar size.
- Live ROM availability monitoring via FileWatcherService. When a ROM is added or removed from a configured ROM directory, the Available gamelist updates automatically approximately one second after the file settles, without requiring an F12 manual scan.
- Auto-refresh of favorites sync on external favorites.ini changes, also via the FileWatcher pipeline.
- Added a check for update button on F1, this will ping the website and compare the local .exe version's GUID against the release on the web site and provide a link to the releases page to download the newer one.
- Introduced a 'persistent' DAT view. Toggle with the alt-tilde key and control the interior navigation of the DAT overlay with **Ctrl** key combinations, e.g. **Ctrl**-D**own**, **Ctrl**-**Page Up**, **Ctrl**-**Home**, etc. Regular navigation keys will move through the gamelist, selecting a new machine. Note: Regular tilde key launches the regular DAT peek which allows full control
- Fixed window-restore under multiple simultaneous MAME launches. IV/Play now stays hidden until the last running instance exits, rather than reappearing as soon as the first one closes.
- Refined CHD filters. chd games now matches any arcade machine that requires a CHD image, and a new chd software filter isolates CD-ROM and hard-disk software list titles (see Appendix 1).
- Screen-count filters are now exact. 2 screens, 3 screens, and 4 screens match machines with precisely that many displays so results line up with MAME's own totals; multi-screen still means two or more.
- History: / hist: searches now include software list items, not just arcade machines — the DAT text for software was always present and is now reachable from the filter bar.

# IV/PLAY 2.8.12 FEATURES

- HBMAME support improvements. IV/Play reads hbmame.ini for HBMAME, identifies it in the title bar and F1 About, and lets F4 link differently named emulator executables.
- Driver-name and Boolean search improvements. Plain words such as model2 also match the exact driver filename, while quoted phrases stay text-only. Boolean searches preserve typed operator order; words alongside an OR group restrict the whole group. See Appendix 1 for examples.
- Filter comparisons now ignore capitalization everywhere. One comparison type still matched case, so a query that differed only in capitalization could return nothing.
- More accurate platform labels and family grouping for software lists. Host-machine resolution was reworked so a platform's canonical parent is chosen correctly, fixing cases where a clone could become the family node or a peripheral's software was mislabeled.
- Fixed custom playlist export so machine entries are written in a form MAME can load. Earlier builds could emit an unusable composite line for machines that own multiple software lists.
- F4 and F5 keep the Full or Available list in use and return to the selected game if it is still listed. When F4, F5 or a cache rebuild leaves the Available list out of date, it is rebuilt before it is shown. F8 also resets the cfg, so press F12 again if Available asks for it afterwards. Software favorites, artwork, HiScore entries and **Ctrl-B** benchmarks are keyed by software list and item (a2600/pacman), even when the item launches on a differently named machine.
- F1 now fits smaller work areas with scrollable settings and a fixed footer, applies newly selected icon sources on OK, and keeps the highlighted game when it remains available. Art selection follows its folder through sorting and removal instead of drifting with a list position.
- Steadier DAT peek rendering. Navigating to a new game while the peek is open no longer briefly blanks the text, and the media-type column now reads consistently. Removed filter-term highlighting inside the DAT peek. It added little once the peek gained its own in-text find (**Ctrl-F**), and the two were unrelated.
- During an icon cache rebuild the gamelist now keeps the icons already on screen and lets the new ones appear as they land, instead of dropping the whole list to generic placeholders for the length of the build. A machine the previous cache had never seen shows the correct generic icon and then pops to its own when the rebuild completes.
- Window width is now checked the same way as height: an implausible width is neither saved at exit nor applied at startup, so one bad value can no longer keep reproducing itself.
- Live monitoring changes now appear on screen at once. Removing a ROM or editing favorites.ini outside IV/Play had updated the lists in the background but left the old view until the next refresh. A favorite whose ROM is missing now stays visible whatever the F1 hide settings, and after F4 links a different executable, F1's command-line suggestions come from the new one.
- The mouse Back button closes overlays, such as the DAT peek, the F2 log, the F3 cfg and the benchmark window, as ESC does.
- SQLite updated to 3.53.4. When upgrading, replace e_sqlite3.dll together with IV-Play.exe; the F2 log names the SQLite version actually loaded.

## IV/PLAY 2.8.14 FEATURES

- Refined Arcade Only list. Restored video games that were unintentionally excluded and improved filtering of gambling, medal, quiz and photo-booth machines. Inclusion and exclusion exceptions now handle related clones more consistently.
- More consistent Arcade results at startup. Category information from catlist.ini is loaded before the Arcade list is filtered, and loading is retried if the file was not yet available.

- More accurate Available list. Games that need a CHD are listed only when their CHD folder is present, and in a merged collection a game is listed only when its BIOS archive is present too. Boards whose ROMs all come from their BIOS, such as the Neo Geo MV-1, are listed from the BIOS archive, and in a split collection a clone needs its own archive. A ROM or disc that MAME marks as never dumped is not required. Detection still reads names only, never file contents. When IV/Play cannot tell how a collection is organized, it lists the game rather than demanding a separate BIOS archive.
- Keep browsing during F5 refresh. When a rebuild finishes, IV/Play keeps the game you most recently selected, provided it remains available, rather than returning to the selection from when the refresh began.
- Internal maintenance and regression checks. Continued work on correctness and automated checks supporting these improvements.

## APPENDIX 1 - FILTER & SEARCH GUIDE

The filter bar (**Ctrl**-F or **Ctrl**-E) accepts plain English phrases, keyword modifiers, and a full SQL-like advanced syntax. All three modes can be freely mixed in a single query.

### SIMPLE TEXT SEARCH

Type any word or phrase. Multiple terms side by side are combined with AND. Quoted phrases match anywhere in description, name or manufacturer, ignoring case: "king of fighters" includes The King of Fighters 2002, and "donkey kong" includes its sequels. Words inside quotes remain literal text; explicit description = "Donkey Kong (US, Set 1)" instead requires the whole stored field.

| | |
|---|---|
| pacman midway | Matches 'pacman' and 'midway' in description, name or manufacturer; an unquoted word can also match an exact machine driver filename stem |
| “metal slug” | Phrase-anywhere match in description, name or manufacturer; quotes keep the words literal and omit driver matching |
| capcom fighting | All Capcom fighting games (both words must appear) |
| system:snes Zelda | Explicit SNES scope + keyword — finds Zelda titles for SNES |

### SYSTEM NAME ROUTING

Ordinary unquoted words search description, name and manufacturer, and also match the exact driver filename before .cpp, ignoring case. model2 finds machines in model2.cpp; galaxian includes other games in galaxian.cpp, and neogeo includes machines in neogeo.cpp. Folder names and partial filenames do not match: nes does not match snes.cpp. Quotes omit driver matching: "galaxian" searches text only. Software never inherits its host driver. Existing year/date handling and visibility filters still apply. Bare system shortnames never activate platform scope;

use system:<shortname> to select a machine together with its software, or add a routing qualifier to select a specific tier.

| | |
|---|---|
| system:nes | Both the NES hardware driver AND all NES softlist media (explicit system scope) |
| nes machine / nes hardware / nes system | Only the NES hardware driver entry (Tier 1). Qualifiers: machine, hardware, system, platform, driver, host |
| nes games / nes titles / nes software | Only NES softlist media — no hardware driver (Tier 3). Qualifiers: games, titles, media, software, items, library, catalog, collection |
| nes cartridges / nes floppies / nes cdroms | NES softlist media filtered by media type. Any media-type word (cartridges, floppies, cdroms, cassettes, hdds, memcards) triggers this rule |
| softlist systems / softlist devices | All hardware drivers that expose a software list, across all platforms (Tier 2) |
| software only / media only / softlist | All softlist media across every platform, no hardware drivers (Tier 3) |
| machines only / no softlist | All hardware drivers only, no softlist media (Tier 1) |
| arcade media | ROM-based arcade machines (MediaType = rom). Distinct from arcade only, which uses the IsArcade flag |

**Note**: qualified forms remain concise: nes machine selects hardware only, nes games selects softlist media only, and nes cartridges adds a media-type restriction. For arbitrary combinations, use explicit scope — for example, system:snes zelda or system:snes supportstatus:yes.

## KEYWORD MODIFIERS

Add these keywords to any search — or use them alone. Keywords can be combined freely.

## WORKING / CLONE / PARENT STATE

| | |
|---|---|
| working only | Show entries not marked non-working / unsupported (broad IsWorking filter) |

| | |
|---|---|
| nonworking / nonworking only | Show only non-working games |
| imperfect | Show games with 'imperfect' driver status |
| parents only | Hide all clone games |
| no clones / noclones | Hide all clone games (aliases) |
| clones only | Show only clone games (overrides global hide setting) |
| not parents / no parents / nonparents | Show only clone entries (inverse of parents only) |
| working parents | Working non-clone games only |
| nonworking parents | Non-working non-clone games only |
| working clones | Working clone games only |
| nonworking clones | Non-working clone games only |
| working clones of nonworking parents | Playable clones whose parent set is broken |
| nonworking parents with working clones | Broken parents that have at least one working clone |

## MACHINE WORKING STATUS (3-WAY)

Use workingstatus: (alias: ws:) to distinguish the three MAME machine driver statuses precisely. working only and nonworking use the broader IsWorking boolean: good and imperfect machines are both working; for softlist items, support yes and partial are both working while no is not. Use workingstatus: when you need to isolate imperfect from fully good machines.

| | |
|---|---|
| workingstatus:working / ws:working | Fully working only — DriverStatus = “good”. Also: fully working games, fully working machines |
| workingstatus:imperfect / ws:imperfect | Imperfect emulation only — DriverStatus = “imperfect”. Also: imperfect machines, imperfect games |
| workingstatus:notworking / ws:notworking | Not working / preliminary — DriverStatus = “preliminary”. Also: broken machines, broken games, not working machines, preliminary |

## CONTENT TYPE

| | |
|---|---|
| softlist / software / software only | Show only software list items (console/computer games) |
| machines only / no softlist | Show only arcade/computer machines, no software items |
| softlist devices / softlist systems | Machines that have a software list attached |
| arcade only / coin-op / coinop / coin op / arcade games | Coin-op, non-mechanical, non-redemption video games only |
| non-arcade | Everything except coin-op arcade machines |
| bios only | Show only BIOS entries |
| no bios | Hide all BIOS entries |
| mechanical | Show mechanical games (pachinko, pinball, etc.) |
| working mechanical | Working mechanical games only |
| not mechanical | Exclude mechanical games |
| devices / device only | Show only MAME device entries (sub-boards, BIOS helpers — not launchable) |

| | |
|---|---|
| no devices / exclude devices | Hide all MAME device entries from results |
| console / home consoles | Home console hardware (machinetype:console) |
| computer / home computers | Home computer hardware (machinetype:computer) |

## FAVORITES

| | |
|---|---|
| favorites only | Show only games in the favorites list |
| exclude favorites | Hide all favorites from results |
| not favorites | Alias for exclude favorites |
| nofavoriteslist / no favoriteslist / noflist | Suppress the pinned Favorites group at the top of the list without removing favorites from results — creates a single flat list. Useful with broad filters: softlist noflist, capcom noflist |

## HARDWARE & DISPLAY

| | |
|---|---|
| vector | Vector graphics games (Asteroids, Tempest, etc.) |
| raster | Raster (pixel-based) graphics games |
| vertical / portrait | Vertically oriented monitor games |
| horizontal / landscape | Horizontally oriented monitor games |
| chd games | Arcade/PCB machines that require a CHD image — any machine with a disk in its manifest (CD-ROM, laserdisc, or hard disk) |
| chd software | Software list titles on CD-ROM or hard-disk media — the software-side counterpart to chd games |

| | |
|---|---|
| laserdisc games | Machines that use laserdisc media |
| cd-based | Arcade/PCB machines with CD-ROM hardware ⚠ Hardware filter — NOT the same as 'cdroms' which filters softlist CD media |
| samples / no samples | Show or hide games requiring external sound sample files |
| screenless | Machines with no display output |
| 1 screen / one screen | Machines with exactly one display |
| 2 screens / 3 screens / 4 screens | Machines with exactly that many displays. Word forms such as “two screens” and “N monitors” also work |
| multi screen / multiscreen | Machines with two or more displays |
| no protection | Hide games with emulated protection schemes |
| not bootleg / no bootlegs | Hide bootleg versions |
| cocktail / cocktail games | Machines supporting cocktail / flip-screen mode (HasCocktailDip = true) |
| no cocktail / upright only | Machines that do NOT support cocktail mode (HasCocktailDip = false) |
| cpu:z80 | Machines using a specific CPU chip. Substring match on the chip list — cpu:z80, cpu:68000, cpu:arm. Multi-word names work: cpu:zilog z80. Machines only — not available for softlist items. |
| sound:ym2151 | Machines using a specific sound chip. Substring match on the audio chip list — sound:ym2151, sound:oki, sound:pokey. Machines only — not available for softlist items. |

## CONTROLS

| | |
|---|---|
| lightgun / lightgun games / light gun | Show lightgun games |
| no lightgun / no light gun | Hide lightgun games |
| trackball / dial / paddle | Games using these specific analog controls |
| dual stick / dual joystick | Games using a dual joystick setup |
| joystick | Games using any joystick input |
| 2-way / 4-way / 8-way | Games with specific joystick directions |
| spinner / spinner games | Games using a spinner (rotary) control |
| mouse / mouse games | Games using a mouse or trackpad control |
| mahjong / mahjong games | Mahjong games (mahjong-keyboard input) |
| keyboard games | Games using a keyboard as primary input |
| keypad | Games using a numeric keypad |
| gambling games | Games using gambling-style input hardware |
| no buttons / 0 buttons | Games with no buttons |
| 1 button / one button | Games with exactly 1 button |
| 2 buttons ... 6 buttons | Games with 2 to 6 buttons (word forms also work: 'six buttons') |
| 1 player ... 4 player | Filter by player count |

| | |
|---|---|
| multiplayer / co-op / 2p | Games supporting 2 or more players (also: 3p, 4p) |
| 1p / singleplayer | Single-player games only |

## MEDIA TYPE (SOFTWARE LIST ITEMS)

These filter software list items by physical media type. For arcade CD hardware, use cd-based or chd games instead.

| | |
|---|---|
| cartridges / cart games / modules | Cartridge-based software (media:cartridge) |
| cassettes / tape games | Cassette tape software (media:cassette) |
| floppies / floppy disks / diskettes | Floppy disk software (media:floppy) |
| cdroms / discs / disc games / cd-rom games / optical discs | CD-ROM software (media:cdrom) |
| hard disks / hard disk / hdds / hdd / hdd games | Hard disk software (media:harddisk) |
| memory cards / memory card / memcards / memcard | Memory card software (media:memcard) |

## MACHINE TYPE

Natural language phrases for machine type categories (requires history.xml in Art Paths). The machinetype: operator (aliases: mtype:, mt:) performs a substring match against the machine-type descriptor extracted from history.xml. Some NLP keywords route via genre: (catlist.ini) instead where no history.xml descriptor exists — noted below.

| | |
|---|---|
| fruit machines | Fruit machines / AWPs (machinetype:fruit machine) |
| slot machines / slots | Video and mechanical slot machines (machinetype:slot machine) |

| | |
|---|---|
| pinball | Pinball machines (machinetype:pinball) |
| electro-mechanical / em games | Electro-mechanical games — routes via genre:Electromechanical (catlist.ini, not history.xml) |
| medal games | Medal / prize games (machinetype:medal) |
| home computers / home consoles | Home computers or consoles (machinetype:computer / machinetype:console) |
| handheld consoles / handheld games | Handheld gaming systems (machinetype:handheld) |
| calculators | Electronic calculators (machinetype:calculator) |
| chess games / chess computers | Chess computers and boards (machinetype:chess board) |
| musical instruments | Electronic musical instruments (machinetype:musical instrument) |
| tabletop games / tabletop | Tabletop game units (machinetype:tabletop) |
| jukeboxes / jukebox | Jukebox machines (machinetype:jukebox) |
| photo booths / photo booth | Photo booth machines (machinetype:photo booth) |
| redemption games | Redemption / prize games (machinetype:redemption) |
| dart games / dart game | Electronic dart games (machinetype:dart game) |
| bingo games | Bingo machines (machinetype:bingo) |
| kiddie rides | Kiddie ride machines (machinetype:kiddie ride) |

| | |
|---|---|
| bowlers / bowler | Bowling alley machines (machinetype:bowler) |
| gun games / gun game | Video gun games (machinetype:gun game) |
| quiz games / quiz machines | Quiz machines — routed via genre:Quiz (catlist.ini, not history.xml) |

## SOFTLIST SUPPORT STATUS (3-WAY)

Filter software list items by their MAME support level using supportstatus: (alias: ss:). Returns no results for arcade machines — combine with explicit system scope for best results, e.g. system:snes supportstatus:yes.

| | |
|---|---|
| supportstatus:yes / ss:yes | Fully supported softlist items (also: supported games, fully supported) |
| supportstatus:partial / ss:partial | Partially supported items (also: partially supported, partial support) |
| supportstatus:no / ss:no | Unsupported items (also: unsupported games, not supported) |

## COMPOUND NO-X AND RUN-TOGETHER FORMS

Most two-word “no X” and “non-X” phrases also work without the space or hyphen. These are fully equivalent to their spaced forms.

| | |
|---|---|
| noclones | Same as no clones |
| nobootleg / nobootlegs | Same as no bootlegs |
| nomechanical / nonmechanical | Same as not mechanical |
| noprotection | Same as no protection |
| noworking | Same as nonworking |
| noarcade | Same as non-arcade (IsArcade = false) |

| | |
|---|---|
| nosamples | Same as no samples |
| nolightgun | Same as no lightgun |
| nodevices | Same as no devices |
| nosoftlist / nosoftware / nomedia | Same as machines only |
| nobios / nonbios | Same as no bios |
| coinop / coin op | Same as arcade only |
| coop | Same as co-op (multiplayer) |
| vertical only / horizontal only | Same as vertical / horizontal |
| noX (general pattern) | Most 'no X' and 'non-X' phrases support a compact no-space form. If 'no clones' works, 'noclones' works too. |

## DATE & DECADE FILTERING

Natural language date expressions are parsed automatically.

| | |
|---|---|
| 1982 | Games from exactly 1982 |
| 79-82 / 1979-1983 | Year range (2-digit or 4-digit, or mixed: 1999-02) |
| after 1990 | Games released after 1990 |
| before 1985 / pre-1985 | Games released before 1985 |
| made in 1982 | Games from exactly 1982 (natural sentence form) |

| | |
|---|---|
| from 1980 to 1985 | Inclusive year range |
| between 1991 and 1995 | Inclusive year range (advanced syntax form) |
| 80s games / in the 90s | Full decade (1980–1989, 1990–1999, etc.) |
| early 90s | First half of decade (1990–1993) |
| late 80s | Last third of decade (1987–1989) |
| 00s games / in the 2010s | Works for any decade including 2000s and 2010s |
| mid 80s | Middle of a decade (1984–1986); works for any decade: mid 70s, mid 90s, etc. |
| since 1990 | Open-ended: 1990 and later (Year >= 1990). Also: from 1990 |
| through 1990 | Open-ended: 1990 and earlier (Year <= 1990) |

## FIELD-SPECIFIC SEARCHES

Target a specific data field using field:value syntax. On string fields, colon (:) performs a case-insensitive contains match; explicit equals (=) performs exact equality. This distinction lets manufacturer:sega match manufacturer strings containing Sega while manufacturer = "Sega" requires the whole value to be exactly Sega.

| | |
|---|---|
| year:1994 | Games from 1994 |
| manufacturer:capcom | Capcom games (contains match) |
| name:dkong | Games whose internal name contains 'dkong' |
| description:puzzle | Games with 'puzzle' in the title/description |
| cloneof:pacman | Clones of Pac-Man |

| | |
|---|---|
| sourcefile:cps2.cpp | Games on the CPS-2 hardware driver (aliases: hardware:cps2 or driver:cps2) |
| media:cartridge | Cartridge software (softlist items) (values: cartridge, cassette, floppy, cdrom, harddisk, memcard, rom) |
| DriverStatus:imperfect | Games with imperfect driver status |
| workingstatus:working / :imperfect / :notworking | 3-way machine driver status filter (alias: ws:). Machines only. Finer-grained than IsWorking: which treats good and imperfect as equivalent |
| supportstatus:yes / :partial / :no | 3-way softlist support level filter (alias: ss:). Softlist items only. Combine with explicit system scope: system:snes ss:yes |
| machinetype:fruit machine | Substring match against machine-type descriptor from history.xml (aliases: mtype:, mt:). Examples: mtype:computer, mt:pinball |
| mameversion:0.285 / mv:0.285 | Exact match on MAME first-appearance version from MAMEInfo.dat. Also: version 285 (natural form — integer only) |
| mameinfo:scramble / mi:scramble | Search full MAMEInfo.dat descriptive text for machines; distinct from mameversion: first-appearance filtering |
| history:scramble / hist:scramble | Search History.xml descriptive text for the given term. Matches both arcade machines and software list items |
| genre:Shooter / gg:Fighter | Substring match against CatList.ini category. Catches all subtypes automatically |
| HasLightgun:true | Direct boolean field access |
| IsArcade:false | Non-arcade machines |
| IsWorking:true | Broad working-state field: good/imperfect machines and yes/partial softlist items |

| | |
|---|---|
| IsFavorite:true | Favorite games (direct field form) |
| UsesSamples:true | Games requiring sample files |
| CloneOf:"" | Parent games (empty CloneOf = not a clone) |

Bare-word DAT operators: the colon is optional for DAT prefix fields. genre fighting, history pacman, workingstatus imperfect, machinetype pinball, and supportstatus partial all work identically to their colon forms. Multi-word values are captured correctly: machinetype:quiz machine finds quiz-type machines without 'machine' leaking into text search.

## GENRE ALIASES

Genre keywords translate to genre: queries against catlist.ini (requires CatList.ini configured in Art Paths). Substring matching is used, so genre:Shooter catches all subtypes — Flying Vertical, Flying Horizontal, Run and Gun, etc. These aliases combine freely with other filters: shmups vertical working, brawlers noclones since 1990.

| | |
|---|---|
| shmups / shoot em up / shooters | Shoot-'em-ups and all shooter subtypes (genre:Shooter via catlist.ini) |
| fighters / fighting games / vs fighters | All fighting game subtypes (genre:Fighter) |
| platformers / platform games / run and jump | Platform / jump-and-run games (genre:Platform) |
| brawlers / beat em ups / belt scrollers | Beat-'em-up / brawler games (genre:Beat) |
| racing / racing games / driving games | Racing and driving games (genre:Racing) |
| puzzle / puzzle games / puzzlers | Puzzle games (genre:Puzzle) |
| maze / maze games | Maze games (genre:Maze) |
| rpg / role playing | Role-playing games (genre:RPG) |

| | |
|---|---|
| sports / sports games | Sports games (genre:Sports) |
| adventure / adventure games | Adventure games (genre:Adventure) |
| casino / casino games | Casino games (genre:Casino) |
| baseball / basketball / soccer / boxing | Sport-specific catlist subgenres (e.g. genre:Baseball, genre:Boxing) |

## DIAGNOSTIC & QUALITY FILTERS

These filters target ROM dump quality and diagnostic cross-queries. Bad dump and no dump data are populated from the MAME XML at build time. Prototype detection uses a description substring match.

| | |
|---|---|
| bad dump / bad dumps / bad roms | Machines with at least one known bad ROM dump (HasBadDumps = true) |
| no dump / nodump / missing dump | Machines with at least one ROM dump not yet found (HasNoDumps = true) |
| clean roms / good dumps / verified roms | No bad or missing dumps (HasBadDumps = false AND HasNoDumps = false) |
| prototype / prototypes / proto | Prototype and pre-release versions (description contains 'proto') |
| no prototypes / no proto | Exclude all prototype / pre-release entries |
| working clones of nonworking parents | Working clones whose parent set is not working (diagnostic cross-query) |
| nonworking parents with working clones | Non-working parents that have at least one working clone |
| working bios / nonworking bios | BIOS entries filtered by working status |
| working mechanical clones | Working clone entries that are also mechanical machines |

| | |
|---|---|
| nonworking mechanical parents | Non-working parent entries that are mechanical machines |

## FIELD NEGATION (!)

Prefix the value with ! to exclude matches.

| | |
|---|---|
| manufacturer:!capcom | Every manufacturer EXCEPT Capcom |
| year:!1994 | Every year EXCEPT 1994 |

## ADVANCED OPERATORS

For precise control, use the SQL-like operators below, mixed with words, quoted phrases and keyword modifiers. Operands keep the order typed. Words side by side apply to the whole explicit OR group: model2 OR daytona nonworking means (model2 OR daytona) AND nonworking. An explicitly typed AND binds more tightly than OR, so model2 OR daytona AND nonworking restricts only the second alternative. Use parentheses to make the intended grouping explicit. A trailing operator with no operand, such as model2 OR, is ignored.

| | | |
|---|---|---|
| = | Equality | manufacturer = "Capcom" |
| != | Inequality | manufacturer != "Sega" |
| > >= | Greater than / or equal | year > 1990 |
| < <= | Less than / or equal | year <= 1985 |
| LIKE | Substring match (% is wildcard) | description LIKE "%Fighter%" |
| NOT LIKE | Substring exclusion | description NOT LIKE "%bootleg%" |
| BETWEEN | Inclusive numeric range | year BETWEEN 1990 AND 1992 |
| IN | Matches any value in a list | manufacturer IN ("Taito", "Irem") |

| | |
|---|---|
| AND | Both conditions must be true working AND year > 1990 |
| OR | Either condition can be true manufacturer = "Sega" OR manufacturer = "Namco" |
| NOT | Negates a condition NOT (manufacturer = "Capcom") |
| ( ) | Groups expressions (manufacturer = "Sega" OR manufacturer = "Namco") |

## PUTTING IT ALL TOGETHER

Any combination of the above modes works in a single query. Keywords are translated to advanced syntax automatically before the filter runs.

**Working, parent-only Midway games:**

```
midway working only parents only
```

**Capcom games from the late 90s, hiding Street Fighter:**

```
capcom late 90s description NOT LIKE "%Street Fighter%"
```

**All Sega or Namco games after 1990:**

```
year > 1990 AND (manufacturer = "Sega" OR manufacturer = "Namco")
```

**Parent puzzle games from 1995 or later:**

```
Year >= "1995" noclones description LIKE "%puzzle%"
```

**NES cartridge games (software list):**

```
nes cartridges
```

**Working 6-button arcade games, no clones, after 1991:**

| |
|---|
| 6 buttons working only noclones year > 1991 arcade only |
| **Diagnostic: find playable versions of broken sets:** |
| working clones of nonworking parents |
| **All imperfect Sega games from the 90s:** |
| sega workingstatus:imperfect 90s |
| **Supported SNES cartridges only:** |
| snes cartridges supportstatus:yes |
| **Fruit machines added in MAME 0.196 or later:** |
| fruit machines mameversion >= 0.196 |

# APPENDIX 2 - USER REFERENCE

## ARCHITECTURE

**Application Architecture Summary**

IV/Play is a Windows MAME/HBMAME frontend optimized for startup, in-memory filtering and GPU rendering. Its live database summary, software index and validated asset caches have distinct responsibilities: the lists are established before filtering, while some asset and metadata work can finish after the initial UI is ready.

**Core Application Flow**

The application's lifecycle begins in Program.cs, which ensures only a single instance is running from the same directory and handles the initial detection of the MAME executable if it's not configured.

The `MainForm` class serves as the central orchestrator. It features two distinct startup paths to maximize performance:

- **Cold Start (First Run / Database Rebuild):** A splash screen accompanies XML processing and SQLite rebuilding when the linked executable, schema or required caches demand it. This is the longest startup path. F4 and F5 use the same data contracts for re-link and refresh.
- **Warm Start ("Perfect Paint" Path):** Machine data is loaded through raw ADO.NET from MachinesSummary while MainForm is being constructed. The software index is loaded, or rebuilt after a failed cache load, before the initial lists and favorites audit use it. The first list is filtered against current data rather than a saved binary game-list snapshot. Archive initialization and icon-atlas work can run in

the background; generic icons may appear until the initial atlas is ready. During later rebuilds, the existing icons remain until the replacement atlas and coordinate map can be published together. Startup timings depend on the linked data, assets and hardware; the session chart is the authority for that launch.

**Data Management and Persistence**

IV/Play's data strategy is robust, designed to transform MAME's raw XML output into a queryable format.

- **Data Source:** The primary source of game information is the XML output from the mame.exe -listxml command.
- **Parsing:** The XmlParser class launches a MAME process and deserializes the resulting XML stream directly into a list of Machine objects. This process populates complex, nested properties for each machine, such as CPU, sound, and display information.
- **Persistence:** A DatabaseManager using raw ADO.NET (SqliteConnection / SqliteCommand) and SQLite stores the processed Machine data in a local database file (IV-Play.db). Entity Framework Core has been fully retired. The database is completely rebuilt if the MAME executable has been updated or if the application's database schema has changed, ensuring data consistency.
- **In-Memory Processing:** MachinesSummary supplies Machine objects in DatabaseManager.AllMachines. Software-list metadata is held in the separate SoftlistIndex. Filters run against these in-memory representations, avoiding database round-trips during ordinary searches.

**Rendering and UI**

The user interface is built for performance, moving beyond standard Windows Forms controls for its core component.

- **DirectX Engine:** The GameList control is a custom UserControl that bypasses standard GDI+ rendering. It uses the Vortice library to interface directly with DirectX (Direct3D11 and Direct2D), enabling smooth scrolling and high-quality image scaling, transparency, and effects like scanlines.
- **Modular Control:** The GameList logic is separated into partial classes for drawing (GameList.Drawing.cs), user input (GameList.Input.cs), and navigation/data handling (GameList.Navigation.cs), keeping the code organized.
- **UI Management:** The MainForm manages the GameList control and auxiliary windows, such as the ConfigForm for settings and the FilterDialog for search queries.

**Multi-Layered Caching Strategy**

To achieve its high performance and rapid startup, the application employs several layers of caching.

- **Icon Atlas Cache:** AssetCache packs icons into a single GPU texture with a disk-backed raw binary atlas, coordinate map and validity seal. Source assets and emulator identity participate in validation; display/settings changes can also require a rebuild. Initial warm loading can be deferred. Later rebuilds retain the published icons until a complete replacement is ready.
- Game List State: The retired binary GameListCache is not a current startup layer. IV/Play builds its visible lists from the live in-memory machine and software data, current favorites, custom-list domain and filter settings.

- **Artwork Cache:** ArtCache implements a memory-based Least Recently Used (LRU) cache for game artwork like snapshots and flyers. This reduces disk reads for frequently viewed art. It uses a background ArtLoader to fetch images from disk asynchronously without blocking the UI.
- **DAT File Cache:** InfoParser uses binary caches for parsed History.xml and MAMEInfo.dat, validated against their source files. Separate metadata hydration makes the configured DAT content available to filter predicates.

**Filtering and Search**

The application features a sophisticated filtering system that allows for both simple and complex queries.

- Natural Language Parser: NaturalLanguageParser translates phrases and modifiers into advanced predicates, preserving explicitly typed Boolean order. Ordinary words match description, name and manufacturer and can add exact machine-driver filename-stem matches. Quoted phrases stay text-only; explicit system scope and routing qualifiers select hardware or software tiers.
- Advanced Filtering: This advanced syntax supports a wide range of operators (=, >, LIKE, NOT, OR, AND, BETWEEN), parentheses for grouping, and specific keywords for fields like year, manufacturer, and driver.
- **In-Memory Execution**: The filter query is converted into a LINQ expression and executed directly against the in-memory list of all games (DatabaseManager.AllMachines), resulting in near-instantaneous filtering.

**Configuration and Diagnostics**

IV/Play includes tools for configuration and troubleshooting.

- **Settings Management:** A central, static SettingsManager class manages all user-configurable options. Settings are persisted to a text-based IV-Play.cfg file. The manager also handles the auto-detection of art folders and DAT files based on the MAME executable's location.
- **Diagnostics:** The application integrates a buffered Logger to record detailed events with minimal performance overhead by flushing writes to disk periodically rather than instantly. A StartupProfiler is also used to time key operations during the launch sequence, helping to identify and diagnose performance issues. It now reports both the primary "time-to-interactive" metric and a separate set of "Deferred Tasks (Post-UI)" timings in the log file.

## APPENDIX 3 - DEEP TECHNICAL REFERENCE

This appendix is the implementation-level complement to Appendix 2. Where Appendix 2 describes the shape of IV/Play—its startup model, rendering pipeline, data flow, caching layers, filter engine, and diagnostics—this appendix describes how those systems are built and why they work the way they do. It is an evergreen reference: updated when the implementation changes, never allowed to accumulate history.

### UNIFIED ARCHITECTURE

IV/Play is comprised of three core components: the Frontend, the Backend, and the SQLite Database. Each has a strict responsibility boundary that must be maintained to preserve performance and correctness.

- **Frontend (UI):** Pure WinForms shell. The core GameList control bypasses standard GDI+ rendering entirely and uses a custom engine built on DirectX (Direct3D11 and Direct2D) via the Vortice library. No GDI or GDI+ anywhere in the rendering path.

- **Backend (Data Layer):** Responsible for XML parsing, database management, cache orchestration and filter execution. Database access uses raw ADO.NET throughout; Entity Framework Core has been retired. OpenRuntimeConnection() centralizes the runtime connection policy for callers that use it, while the descriptor-driven summary loader opens its own read connection.
- **SQLite Database (IV-Play.db):** Central repository for all game metadata. Two tables drive the application: Machines (full XML-derived records) and MachinesSummary (flattened, scroll-optimized, defined by MachineSummarySchema.Columns). For maximum filtering speed, the entire MachinesSummary table is loaded into the static in-memory list DatabaseManager.AllMachines at startup. All filter execution runs against this list—no database round-trips occur during user interaction.

## RENDERING ENGINE

All drawing operations are performed by the GPU via DirectX 11 / Direct2D. The pipeline is built on the Vortice libraries and enforces strict no-allocation rules in the hot render path.

- **Pipeline components:** ID3D11Device and swap chain for surface management, ID2D1DeviceContext for drawing, a sprite batch for atlas icons, and DirectWrite for text. DPI-aware geometry and scaling govern layout and rendering; the frame scheduler coordinates presentation.
- **Layout geometry:** Fixed 60/40 split—the game list occupies the left 60% of the control width, the art panel the right 40%. Calculated in Gamelist.Drawing.cs as artZoneWidthInDips = widthInDips * 0.40f. Art panel bounds are stored in _imageArea (SysRectangleF). In Grid mode the grid occupies exactly the space the list would. In No-Art mode _imageArea is set to SysRectangleF.Empty and the full width is used.
- **DPI scaling:** GetDpiScale() returns the current scale factor (1.0 at 96 DPI, 1.5 at 144 DPI, 2.0 at 192 DPI). All layout calculations are performed in device-independent pixels (DIPs) and converted to physical pixels for DirectX calls. Multi-monitor DPI transitions are guarded by _atlasDpiScale, _dpiChanging, _shownComplete, and _renderSuspended flags in MainForm and GameList respectively.
- **Device-lost recovery:** RenderState gates active rendering, resizing and recovery, and a stale draw batch triggers recovery before another frame. MainForm retains the device-independent WIC icon and artwork caches, recreates the GPU resources and re-uploads the surviving atlas; current artwork is restored from its cache when available. Disk atlas reload is an exceptional fallback, not the normal recovery path. If automatic recovery is suspended, F5 provides the existing manual retry.
- **Art scaling:** DirectX High Quality Bicubic interpolation by default; nearest-neighbor available via art_scaling_nearest setting for a pixelated retro look. Scanline overlay (scanlines_enabled) is suppressed automatically for vector games and non-MAME-generated art types such as flyers, marquees, and PCBs.

## GAMELIST PARTIAL CLASS STRUCTURE

The GameList control's logic is divided across partial class files with distinct responsibilities; its retired history-highlight partial remains empty. The key divisions are:

- **Gamelist.cs** — Core state. Owns _mainGameList, _favoriteGameList, _activeGameList, _selectedGame, _filter, _dataLoaded, and _currentSystemProvider.
- Gamelist.Drawing.cs — DirectX rendering of the main game list, artwork, borders and overlays.
- **Gamelist.Input.cs** — Keyboard and mouse handling.
- **Gamelist.Navigation.cs** — Scroll position, focus management, back/forward navigation stack.
- **Gamelist.Sorting.cs** — Sort logic for all view modes including Flat View global sort.

- **Gamelist.Predicates.cs** — Static filter predicate helpers such as IsArcadeVideoGame() and MachineHasAtLeastScreens(), used for conditions that cannot be expressed in pure SQL-like field comparison syntax.
- **Gamelist.DatOverlay.cs** — Tilde peek overlay rendering and sizing.
- **Gamelist.HistoryHighlight.cs** — Retired. Previously highlighted filter terms within the DAT peek; that behavior was removed once the peek gained its own in-text find (**Ctrl-F**). The file is retained as an empty partial.
- **Gamelist.BVT.cs** — Build Verification Test harness (F10). Must remain synchronized with any changes to GameList interaction, navigation, or rendering logic.

# DATA LAYER

## RAW ADO.NET (ADO-ONLY)

Current database access is raw ADO.NET. OpenRuntimeConnection() applies the shared WAL/synchronous/foreign-key policy where used, while the warm summary loader has a dedicated read connection. The former EF contexts, generated compiled models and EF packages have been removed. The first runtime connection logs the SQLite engine version, source identity and loaded e_sqlite3.dll path, allowing a mismatched native library to be diagnosed.

## MACHINESSUMMARY TABLE

MachinesSummary is the flattened, scroll-optimized projection used by the warm loader. MachineSummarySchema.Columns drives its table declaration, projection, select list and reader mapping together. A new field still needs matching production data in Machines and any specialized direct readers; the descriptor does not make those separate boundaries automatic. On-disk summary schema comparison can bring added columns forward without a separate handwritten migration. The schema and hydration scanners guard the remaining producer/reader couplings.

LoadAllMachinesIntoMemoryAsync() returns CacheResult<List<Machine>>. A successful summary load is Fresh. When the fast path fails, the raw full-hydration fallback populates the list and returns Stale with a diagnostic; the failure and fallback are logged rather than silently treated as a normal summary load.

## CACHE RESULT PROVENANCE

Cache-style operations across the application return CacheResult<T>—a zero-heap-allocation record struct carrying the data and a CacheQuality enum (Fresh / Stale / Partial / Missing / Invalid / Failed). Callers are required to inspect quality and log when the result is not Fresh; silent fallbacks are prohibited by convention. Affected subsystems: SoftlistParser.ParseAll(), AssetCache.LoadFromCacheAsync(), and LoadAllMachinesIntoMemoryAsync(). DatInfoParser intentionally does not use CacheResult—its single call site has adequate inline discipline and the type would over-engineer it.

## ATOMIC WRITES

All persistent state writes follow the write-to-temp-then-rename pattern: data is written to a .tmp sibling on the same NTFS volume, then promoted atomically via File.Move(overwrite: true). A crash anywhere in the write window leaves either the previous file intact or the new file complete—never a partial write. This pattern is enforced in three subsystems:

- **SettingsManager:** IV-Play.cfg is written to a .tmp file and renamed.
- **AssetCache:** The three-file atlas group (.bin / .map.json / .hash) is staged as three .tmp files, renamed in order, with the hash file written last as a seal—a missing or mismatched hash on load forces a full atlas rebuild.
- **DatabaseManager:** The database is rebuilt to IV-Play.db.new, then moved to the live path; the metadata seal is written to a .tmp and moved last. A crash anywhere in this window forces a rebuild on next launch rather than silently loading a corrupt file.

## STRING INTERNING

During loading, repeated strings are deduplicated through InternedStringPool and the CLR's process-lifetime intern table. MachineSummarySchema.StringPool delegates to that helper; it is not a separate local dictionary, and its logged size of zero does not mean interning is disabled.

## COMMANDS TABLE

The Commands table is populated during cold start from mame.exe -showusage. At warm start, LoadMameCommandsAsync() reads this table into DatabaseManager.MameCommands (a Dictionary<string, string>), which drives the command-line override autocomplete in ConfigForm via the ComboGhostPainter NativeWindow subclass. This step is explicitly non-fatal—if the table is absent in an older database, loading proceeds normally and autocomplete is simply unavailable until the next cold start.

## MULTI-LAYERED CACHING

- **Icon Atlas Cache (AssetCache):** Icons are packed into a texture and cached as IV-Play.icon.atlas.bin with IV-Play.icon.map.json and IV-Play.icon.atlas.hash. Source assets and the linked executable participate in validation; relevant display/settings changes can trigger rebuilding independently. Initial atlas loading may run in the background. Published icons survive subsequent rebuilds until the new atlas and map are swapped together. Loose .ico files, .zip packs and non-solid .7z packs are supported; an installation with no icon source can retain its valid generic atlas.
- **Softlist Index Cache:** Software metadata is stored in the dictionary-compressed binary IV-Play.SoftlistIndex.cache. It is loaded, or rebuilt on cache failure, before the initial consumers run. Unloaded, ready (including legitimately empty) and unavailable are distinct states. If the index cannot be established, favorites audit reports unjudged software entries and preserves them rather than marking them invalid without an authority.
- **Artwork Cache (ArtCache):** Memory-based LRU cache for snapshots, flyers, and other art types. A background ArtLoader fetches images asynchronously so the UI thread is never blocked. Supports both loose files and .zip / non-solid .7z archives. Solid .7z archives are detected and warned against on startup due to their incompatibility with random-access performance.
- **DAT File Cache:** History.xml and MAMEInfo.dat are parsed into binary caches validated against their sources. InfoParser owns the cached peek data; MetadataHydrator exposes the configured content to

filters. Historical parsing measurements are recorded in Appendix 5 rather than treated as current release guarantees.

# FILTER ENGINE

## PIPELINE

Filter execution follows a fixed pipeline: Tokenize → Normalize → NLP translation → Build predicate → Apply to DatabaseManager.AllMachines (in-memory LINQ). NaturalLanguageParser.cs translates human-readable queries into SQL-like advanced filter syntax. FilterExpressionParser.cs tokenizes the resulting string, builds a recursive-descent expression tree (ParseOrExpression → ParseAndExpression → ParseComparison), and compiles it into a LINQ lambda. The compiled predicate is applied to the in-memory list; no database round-trips occur.

## DAT INTEGRATION AND METADATAHYDRATOR

History.xml, MAMEInfo.dat and CatList.ini feed runtime filter metadata through MetadataHydrator. Hydration can occur when a filter needs a source or through startup work, with separate completion state for each source and for machine versus software history. Replacing the machine data or changing configured DAT inputs requires the corresponding reset/re-hydration.

DAT operators: history: / hist: searches History.xml text for machines and software; mameinfo: / mi: searches full machine MAMEInfo.dat text; mameversion: / mv: searches first-appearance versions; genre: / gg: searches CatList.ini categories; machinetype: / mtype: / mt: matches History.xml machine-type descriptors. workingstatus: / ws: distinguishes machine good / imperfect / preliminary status, while supportstatus: / ss: distinguishes software yes / partial / no support.

## AUTOCOMPLETE

The filter bar (**Ctrl-F**) uses GhostTextBox, a TextBox subclass that renders VSCode-style inline ghost text via WndProc/WM_PAINT interception. The ConfigForm command-line override field uses ComboGhostPainter, a NativeWindow subclass that attaches to the internal edit control of a DropDown ComboBox and renders ghost completion text against DatabaseManager.MameCommands. WinForms TextBox controls do not participate in the standard paint pipeline, making WndProc interception the only reliable approach for inline ghost text in this framework.

# AVAILABLE GAMELIST PIPELINE

AvailableListProvider performs a name-only, non-recursive audit of top-level ROM archives and CHD folders on the linked emulator's rompath and saves IV-Play.available.bin. The shared ini reader follows the MAME/HBMAME config name and inipath, and relative paths are resolved from the emulator's directory. F12 builds the audit and selects Available. Machines that need no file, with no dumped ROM or disk of their own, are intentionally omitted. Each audit estimates the collection layout from whether clones beside a present parent carry archives of their own. With no such evidence it logs unknown (permissive), allows parent-archive fallback and does not require a separate BIOS archive. The estimate can change as archives are added or removed; a mixed or incomplete collection can therefore change which games are listed. In a merged collection a clone resolves to its parent's archive and a game also requires its BIOS archive; in a split or non-merged collection each game needs its own

archive and no BIOS archive is required, because a non-merged archive carries its BIOS ROMs and names alone cannot tell the two apart. A machine whose dumped ROM file names all appear in archives present up its BIOS or parent chain resolves to those archives, and a machine with a dumped disk needs a CHD folder named after it or its parent; ROMs and disks with no dump are never required. The log records the layout and how many machines were held back for a missing BIOS archive or CHD folder. Live watchers maintain the results as archives change; CHD folders are read when the audit runs. F4/F5 repair required missing or stale availability after database/index work and before filtering Available, retain the selected list, and restore the selected row if it remains. A required repair failure retains the displayed list and is reported rather than claiming refresh success. Before the first audit, selecting Available prompts for F12; later missing-cache self-healing follows the retained list and availability-history policy. F8 can clear that history when it resets settings, so use F12 again when prompted.

## FLAT VIEW AND GLOBAL SORT

Flat View is a non-indented display mode in which clones are decoupled from their parents and favorites are not pinned at the top. It is the only view mode that supports global sort. The sort field (Description, Machine Name, Manufacturer, Year) is stored in the sort_flat_alphabetical setting and persists across restarts when Flat View is active; sort direction resets to its natural default on launch. Sort field cycles via Alt-S; sort direction via Shift-Alt-S. Gamelist.Sorting.cs owns the sort logic and must be updated if new sort fields are added.

## MACHINE STATUS-AWARE ICON BORDERS

Icon borders are colored to reflect MAME driver status: working (green), imperfect (orange), not working (red). Colors are configurable per-status in the settings; the imperfect color defaults to a background-derived themed color to complement the dynamic theme system, and can be overridden to a fixed RGB value in F1. The border coloring extends to softlist media items. The implementation draws the status-colored border in Gamelist.Drawing.cs after the icon sprite is rendered, using Direct2D rectangle drawing with the status color applied as a stroke.

## SETTINGS MANAGEMENT AND SNAPSHOT SYSTEM

SettingsManager marks persisted settings with [Persisted]; [Category] groups user-facing sections. Shared codecs provide declaration-driven load/write behavior, and reset defaults are captured before user settings are loaded. Explicit residual keys and computed/preserved defaults remain intentional exceptions. The settings scanner checks declarations, wire names and the remaining explicit sites so a setting cannot silently drift across save, reload and reset.

DumpSettingsSnapshot(label, SnapshotRuntimeContext, emitToLog) captures the full in-memory settings state at three points per session: application startup (“Startup” label), immediately before F1 opens (“Pre-F1”), and immediately after F1 closes (“Post-F1”). A delta marker (Δ) is appended to any field whose value changed from the previous snapshot. The most recent Post-F1 block is pink-highlighted in the F2 log overlay. The _lastSnapshotLines field stores the previous snapshot body for comparison; the Startup snapshot is never annotated as it establishes the baseline.

## DIAGNOSTICS

- **Logger:** Structured producer/consumer logger. All call sites use a typed API — Logger.Info(LogCategory, message), Logger.Warn, Logger.Debug, Logger.Error — with an explicit subsystem category on every entry. The line format is: hh:mm:ss.fff tt [T:nn] [SEV] [CAT] message, where T:nn is the managed thread ID, SEV is one of VRB/DBG/INF/WRN/ERR, and CAT is one of 21 subsystem tags: PRG (startup), CFG (settings), DB (database), DAT (XML/DAT parsing), GUI (main form), ARC (archive provider), AST (icon atlas), BKG (background manager), AVL (available list), MAM (MAME process), FAV (favorites), GEO (geometry), DPI (DPI scaling), MON (monitor enumeration), WIN (window state), SPL (splash), ART (art loading), AUD (auditor), MEM (memory snapshots), GEN (general/unclassified), UX (operator actions). Entries are queued into a bounded BlockingCollection and drained by a dedicated consumer thread running at BelowNormal priority, keeping the hot path allocation-free. If the queue is full, the caller falls back to a direct synchronous write rather than blocking or dropping. Error-level entries always pass regardless of MinimumLevel or CategoryFilter. Debug logging is toggled via F1 (debug_mode); a per-category filter (debug_category_filter in the cfg, comma-separated category names) is available for targeted subsystem isolation. The log file rotates automatically and keeps prior generations beside it. FlushNow() provides a synchronous sentinel-based drain used by Stop() and crash handlers. In the F2 log overlay, each category is rendered in a distinct canonical color defined in Logger.CategoryColors, making subsystem activity immediately scannable. F2 state-dump sections (written when F2 opens) are stripped from the log file on clean exit via RemoveF2DumpsFromFile(). Legacy call sites (WriteToLog, LogDebug) are retained as [Obsolete] and route to the typed API with LogCategory.GEN. Three further mechanisms were added in the July 2026 revision, each closing a case where a reader of the log had to guess. Logger.Operation(category, name) returns a disposable scope that emits paired OP[n] START and OP[n] COMPLETE lines carrying an elapsed time, or an OP[n] ABANDONED line with a required reason, so multi-threaded work that spans many lines can be grouped and timed rather than inferred. LogCategory.UX and Logger.UserAction record what the operator did, as distinct from what the application did — window moves and resizes are bracketed as single gestures rather than logged per frame — so cause can be tied to effect without reading it back out of side effects. The session header now declares the verbosity and category filter actually in force along with the assembly's module version id, which makes the absence of a line evidence rather than ambiguity and settles whether a given log came from the build it is claimed to. A closing summary written before the final drain re-lists severity and category tallies, the slowest completed operations, and every warning and error in one place; it adds no new data, and exists because arrangement is the difference between present and noticed.
- StartupProfiler records internal readiness, first visible reveal and launch-to-first-appearance as distinct milestones. F1 displays True Launch, matching the chart's named row rather than the internal-ready timestamp. First-run time spent choosing an emulator is excluded. Cold/warm labeling follows the recorded startup path, and post-UI work appears separately in the session chart.
- **Developer overlays:** F2 displays the diagnostic log and settings/state snapshots, F3 the cfg, F7 the performance dashboard, F6 the asset auditor, and F10 the in-process BVT. F10 covers navigation, filtering, sorting, export and software identity and restores the settings it exercises.

## RUNTIME DIAGNOSTIC IDENTITY

When sharing a log, identify its session timestamp and BUILD mvid from the second line. The startup banner and F1 About show the same executable-file build timestamp. The log names the linked MAME/HBMAME, the Direct3D adapter, and the SQLite engine version/source and loaded DLL path. The current SQLite package is 3.53.4; the runtime line proves which library actually loaded. Normal absence of optional art on a fresh setup is not a stale-path failure. Clean exit writes one final session summary.

## BENCHMARK ENGINE

The benchmark exists once and is driven from two front ends. BenchmarkRunner holds the entire implementation — MAME invocation, result parsing, the three means, report assembly, sleep prevention, history writing and the regression check — and has no compile-time edge to WinForms at all, reporting progress through IProgress<string>. BenchmarkOverlayForm is presentation only: a window, a results view, a find bar and a cancellation source. HeadlessBenchmark is the second front end and owns nothing but command-line policy — argument detection, console attach, single-instance acquisition, settings bootstrap and exit codes.

The shape is the point. Before the extraction the engine lived inside the overlay form and was reachable only by opening a window, so adding a command-line mode would have meant a second copy of the parse and report logic, which is the drift class this project has recorded repeatedly. There is now one implementation and no way for the two paths to disagree about what a run means.

Run isolation is composed once per run and folded into every machine invocation, so a figure does not depend on the operator's ini hierarchy. Locations are carried across and tuning is not — the distinction that matters, since a suppressed ROM path produces a run that measures nothing and does not say so. Benchmark runs also opt out of efficiency-class scheduling and pin to performance cores, and the resulting placement is read back and logged rather than assumed. The two history files are written by the engine, not by either front end, and the regression check reads its own per-machine file back to build each baseline.

Exit codes are fixed rather than improvised, since they are the entire interface between an unattended run and whatever scheduled it. A run that produced no report is distinguishable from a run that was refused, and both from a run that hung:

0 — Report written; at least one machine produced a figure.
1 — Another IV/Play instance is running; nothing was started.
2 — MAME is not configured and was not found beside IV/Play.
3 — The benchmark_games list resolved empty.
4 — The suite ran but no machine produced a parseable figure.
5 — Figures were obtained but the report could not be written anywhere.
6 — Unhandled error.
7 — The watchdog fired; the suite exceeded its wall-clock budget and MAME was closed.

A refusal on code 1 reaches the console and the exit code but never IV-Play.log — the log belongs to the instance already running, and leaving it untouched is the point of refusing.

## READING BENCHMARK REPORTS AND HISTORIES

Speed (%) is MAME's emulation throughput. Seconds is emulated duration; Wall (s) is elapsed real time. GHz avg/max is effective operating frequency from Windows telemetry, not a measured speedup. Freq n counts valid frequency samples; short real-time games can still have sparse telemetry. Reports disclose the current 250 ms sampling cadence. Sample counts and shorter-window maxima should not be compared directly with older 1 Hz reports.

Other CPU (%) estimates eligible CPU capacity used by work other than MAME: performance-core capacity when pinned, whole-CPU capacity otherwise. Core-activity attribution is system-wide and carries a coverage/reliability indication; unrelated background work can dominate a short run. CPU accounting ends when MAME exits.

The AVERAGE telemetry row gives each game with valid telemetry equal weight. Its GHz max value is the mean of per-game maxima; Whole-run CPU peak separately reports the highest observed frequency. The arithmetic, geometric and harmonic speed means summarize emulation throughput and are distinct from this telemetry row.

Regression Check uses per-machine historical baselines matched by host, launch path, emulated duration and CPU model/logical-processor count. All-Time Overview applies those same keys plus game count and includes the current run. A processor replacement on the same Windows host starts a new CPU cohort; nominal MHz is not part of that identity. Interactive, CommandLine and Unattended label the invocation path; Unattended does not prove that Windows had no logged-on interactive user.

Both history files are tab-delimited. IV-Play.Bench.History.txt columns: Host Machine, Date, Launch Path, Duration, Game Count, OS, CPU, GPU, RAM, MAME Build, Arithmetic Mean, Geometric Mean, Harmonic Mean. IV-Play.Bench.Machines.txt columns: Date, Host Machine, Launch Path, Duration, CPU, MAME Build, Machine, Speed. Machine/software report identities use the plain machine shortname or software-list/item key respectively; **Ctrl-B** sends the selected item's full launch configuration separately.

Older per-machine rows can acquire CPU identity only from a matching aggregate row for the same run. Unknown CPU or missing scope fields exclude a row from the comparison pool; they are not guessed. A diagnostic-only run is kept as a report without becoming a baseline. A report written successfully does not prove that either history file was updated: the completion output reports both writes separately.

## CONCURRENCY MODEL

A four-stage formalization of the concurrency model, run in August 2026, produced a small number of ownership rules that are now explicit rather than emergent.

The MAME process handle has a single owner. It had accumulated one writer and several independent readers, each testing liveness on its own terms, which is the shape that produces a race nobody can reproduce. Shared collections that the UI reads are replaced by swap rather than mutated in place, so a reader always sees one coherent state and never a half-applied one; the favourites key set was the last holdout and is now under the same rule as everything else. The icon cache is constructed privately and published in a single step once it holds a real atlas, so nothing outside the rebuild can observe a coordinate table and a texture that came from different builds.

Two guarantees documented in earlier revisions were found to be verified by checks that could not fail, and one gate meant to suppress redundant work was positioned so that it dropped real events instead. Both classes are closed. Where a residual risk was accepted rather than engineered away — a wedged batch must not be able to block a rebuild forever, so a promotion proceeds after its wait times out — the acceptance is recorded at the call site rather than left as a silence for a later reader to mistake for an oversight.

Icon and archive changes, startup, F1, refresh and shutdown never overlap, so the icon atlas is never built from partly changed sources.

## LIVE ROM MONITORING (FILEWATCHERSERVICE / CHANGEQUEUE / BATCHPROCESSOR)

Three classes cooperate to keep the Available gamelist current without user intervention: ChangeQueue is a thread-safe, HashSet-backed dedup queue. Duplicate paths received before a drain are collapsed to one entry. A “full rescan” sentinel (null path) can be raised and is sticky until the next drain, ensuring no change event is lost even during a buffer overflow. FileWatcherService wraps one FileSystemWatcher per ROM directory (watching *.zip and *.7z, top-level only) and a separate FSW for the directory containing favorites.ini (filtered to that filename). All events funnel into the shared ChangeQueue. FSW buffer overflow triggers EnqueueFullRescan so no change is silently dropped. BatchProcessor drains the queue on a short timer tick with a stability window: if any change was queued very recently the tick is skipped, preventing a burst of FSW events (e.g. a large copy) from triggering many sequential rebuilds. A single rebuild fires shortly after the last file settles. Batch routing: a null batch (full-rescan sentinel) or a ROM archive path triggers AvailableListProvider.BuildAndSave(); a favorites.ini path triggers DatabaseManager.SyncFavoritesAsync(). Start/stop is owned by MainForm; the processor does not auto-start. RomChangeProcessed and FavoritesChangeProcessed events notify callers on completion. The service is active during both warm and cold start paths; the log reports the actual watcher count for the configured ROM directories and favorites.ini.

## EXTRACTED SERVICE CLASSES

Several responsibilities previously embedded in larger classes were extracted into standalone service classes during the May 2026 re-org:

- **InstanceLock** (extracted from Program.cs): Produces stable, path-derived names for the single-instance mutex and any future named pipes, so that multiple side-by-side installations each get their own lock. Naming strategy: names are derived deterministically from the normalized startup path, so an installation always maps to the same lock and different installations never share one. GetMutexName(prefix) and GetPipeName(prefix) return the corresponding names. The derivation is computed once and cached.
- **InternedStringPool** (extracted from DatabaseManager inline logic): Canonical string-interning helper shared across all subsystems that deduplicate high-cardinality strings at load time (machine names, manufacturers, genres, source files). Delegates to string.Intern(), placing values in the CLR’s permanent intern table for process lifetime. Null-safe: returns null when passed null. The warm-start profiler now reports StringPool size: 0, confirming the CLR intern table rather than a local dictionary is the active path.
- **LayoutMetrics** (Phase 3 DPI system): Centralized DPI-aware layout constants authored against a single design DPI (the 4K/HiDPI baseline). When the runtime DPI matches the design DPI, scale = 1.0 (pixel-perfect); at other DPIs the design values scale proportionally. D2D properties return DIPs directly (each design unit converted once at the source). Window geometry properties return design-DPI pixels directly for callers that apply RuntimeDpi/DesignDpi themselves. Initialize(runtimeDpi) is called once after Application.SetHighDpiMode; Update(newDpi) responds to WM_DPICHANGED.
- HiscoreLoader: Loads HiScore.ini once at startup and rewrites it sorted by case-insensitive entry key, preserving score-less stubs. Keys are machine shortnames or software-list/item composites, matching Favorites.ini rather than the launch host name. Score text is free-form and the date optional. Lenient tab/space handling supports hand-edited files. No hot reload is performed; restart after an edit. F8 preserves the file.
- **Relocations**: GameListSearch was moved from Controls/ to Classes/ to align with its non-UI data-only role. SoftwareItem was moved from Classes/ to Data/Models/ to co-locate it with its peers. Neither relocation changes behavior.

## NATIVE AOT RUNTIME MODEL

Beginning with version 2.6, IV/Play ships as a Native AOT (Ahead-of-Time compiled) executable. The .NET 10 runtime is embedded in the binary and is not a separate installation requirement. The distributable is larger as a result, but startup behavior is more consistent across machines and warm launches are smoother. AOT compilation required resolution of WinForms UIA reflection constraints. EF Core has since been fully retired; all database access is raw ADO.NET, which is AOT-safe by default. The S.cs helper class provides AOT-safe string obfuscation using Base64/XOR encoding; the encode method is present only in DEBUG builds.

### RECORDED PERFORMANCE SNAPSHOT — MAY 2026

The following measurements are the recorded v2.8.4 reference snapshot from May 2026, not a performance claim for the current v2.8.14 build or newer hardware. Appendix 5 preserves the longitudinal measurements; consult the current session log for its actual startup timings.

**Test configuration:** i7-12700K, 64GB DDR4, NVMe SSD, 4K Primary (3840×2160, 144 DPI) + 1440p Secondary (2560×1440). IV/Play 2.8.4, MAME 0.287, 42,839 machines, 140,763 software list items, Native AOT (.NET 10).

**Warm start milestones:** [1] Initial I/O ~69ms → [2] UI Interactive 957ms → [3] Full Startup ~1.07s.

**Interactive-phase breakdown (TTI 957ms):** DB Load (EagerInit) 684ms • DB Load (main thread, overlapping) 502ms • Softlist Index Load 179ms • Settings Load 57ms • DirectX Init 42ms • Background Load 18ms. Note: FilterGamesAsync and Gamelist Filter (RAM) are no longer measured in the warm-start TTI critical path following the May 2026 re-org.

**Deferred tasks (post-UI, ~408ms background):** Atlas Valid+Load 246ms • Art Path/Archive Init 100ms • Softlist Index Load 179ms.

**Cold start (one-time per MAME version):** Atlas Build 3.65s • XML Parse & Process 2.52s • DB Write 1.11s • DAT Load History.xml 879ms • DAT Load MAMEInfo.dat 663ms • Internal Ready 10.79s • Launch-to-splash 193ms.

## APPENDIX 4 - DEVELOPMENT HISTORY

This appendix is the engineering archaeology of IV/Play—the record of how the application got to where it is. It documents eleven months of modernization from June 2025 through May 2026, structured around the nine architectural arcs that define the system today. Unlike Appendices 2 and 3, this appendix is append-only: the history is not rewritten, only extended. New entries are added when a significant architectural arc completes.

### FOUNDATION STABILIZATION & EARLY ARCHITECTURE (JUNE–JULY 2025)

The project enters source control on June 30, 2025, already mid-flight. The first weeks are chaotic: focus issues on launch, broken favorites navigation, icon regressions, scroll judder, properties panel failing to populate, softlist launch commandline malformed, and several sessions accidentally committing handoff summaries as commit messages. Despite the turbulence, the foundations land: SQLite replaces the old database format, MAME XML parsing is hardened, softlist scraping begins, the properties panel appears, and the startup pipeline gets its first performance pass. A binary DB cache is attempted and then abandoned due to ghost entries and GC spikes. String interning is introduced—the technique that survives to the present day.

### RENDERING, DPI, AND SOFTLIST DEPTH (AUGUST–SEPTEMBER 2025)

This period is dominated by DPI battles and softlist ordering nightmares. The icon atlas becomes DPI-aware and the high-DPI scaling matrix is stabilized. Softlist parent/clone sorting repeatedly breaks: NTSC/PAL ordering takes multiple attempts, and region-variant orphans require hand-tuned fixes. Properties view expands with CPU, sound, and softlist metadata. The non-working overlay is fixed. Aspect ratio heuristics for snaps are introduced. A custom color picker for art borders and scrollbar thumbs debuts. The rendering pipeline becomes robust enough to support all later expansions.

## DYNAMIC THEMES, SCROLLBAR, AND INPUT INFRASTRUCTURE (OCTOBER 2025)

October is dominated by the dynamic theme engine: five themes derived from background hue, ESC cancellation logic, brightness detection, theme randomizer, and conflicts between custom colors and theme overrides. The vertical scrollbar lands—a deceptively simple feature that triggers weeks of layout and DPI adjustments. ShortcutManager and FilterDialog autocomplete infrastructure appear, laying groundwork for the filter engine rewrite.

## SOFTLISTS BECOME FIRST-CLASS CITIZENS (NOVEMBER 2025)

The first major architectural inflection point. 140,000 softlist items become filterable. The unified filter pipeline (BuildGameLists) is born. Arcade machines are sorted above ports. Archive support for .zip and .7z art assets lands. Multi-instance MAME launch (Shift-1 through 9) is added. The SQLite MachinesSummary summary table is introduced. The performance dashboard is rewritten multiple times. IV/Play becomes a mixed-domain frontend for the first time.

## MIXED-VIEW FILTER HARDENING (DECEMBER 2025)

December is a war zone of filter bugs: dkong softlist produces inconsistent results, NTSC/PAL ordering breaks repeatedly, the Dendy ordering problem resurfaces, noclones interacts badly with softlist results, softlist favorites fail to appear after cold start. The app begins hiding itself on MAME launch. Raw ADO.NET bulk insert accelerates DB writes. The binary gamelist cache is retired in favor of a live ADO.NET query—the Perfect Paint architecture. A schema bug silently breaks the Fast Path by omitting the Features column from the MachinesSummary CREATE TABLE statement; the Fast Path falls back to full EF Core hydration on every warm start without surfacing a clear user-facing error. This bug persists until February 2026.

## FAMILY GROUPING, PROPERTIES OVERHAUL, AND THE SCHEMA BUG FIX (JANUARY–FEBRUARY 2026)

January introduces family grouping—collapsing hundreds of softlist variants into drillable nodes. Expand/collapse logic breaks navigation repeatedly; ESC fails to restore the originating entry; the back-stack logic is rewritten. Version 2.4.12 ships. February uncovers the Features column schema bug: MachinesSummary CREATE TABLE was missing the Features column while both INSERT and SELECT referenced it, silently adding ~157ms to every warm start since December. Fixed in 2.4.14 by adding the missing column definition to DatabaseManager.cs. F8 factory reset could not fix it because the bug was in the code itself, not the data. Properties view gets a major overhaul: cocktail mode detection, bad/no-dump flags, working status aligned with MAME's color coding. Performance dashboard switches to bar chart format only.

## FILTER POLISH, AUTOCOMPLETE, DAT INTEGRATION (LATE FEBRUARY–EARLY MARCH 2026)

GhostTextBox (filter autocomplete) and ComboGhostPainter (command-line autocomplete) land. The Commands table from mame.exe -showusage drives command-line completion. History.xml, CatList.ini, and MAMEInfo.dat become fully filterable via MetadataHydrator. HiddenIniParser is introduced. Flat View with Global Sort (Alt-S) ships. Show Machines / Show Parents toggles are added. AvailableListProvider implements the F12 ROM scan. The Settings Snapshot delta marker system (DumpSettingsSnapshot, SnapshotRuntimeContext) is added. DAT search term highlighting (Gamelist.HistoryHighlight.cs) is implemented. DPI multi-monitor hardening adds _atlasDpiScale, _dpiChanging, _shownComplete, and _renderSuspended flags. Version 2.6 ships.

## MIXED-VIEW UNIFICATION AND NATIVE AOT (MARCH 2026)

The second major architectural inflection point. Softlist predicate matching, softlist proxy construction, and unified filtering of machines plus softlists are completed. Correct bypass of softlist toggles when user intent is explicit. Correct merging into the unified list. Native AOT transition requires EF Core compiled model generation and resolution of a WinForms UIA reflection crash (DataGridViewRowHeaderCell parameterless constructor trimmed by the AOT linker). The FilterGamesAsync cost rises to ~331ms as a consequence of DAT data participating in the filter predicate build pass across 42,700+ machines—an intentional architectural trade-off. Version 2.6.2 ships as the first Native AOT build.

## FORENSIC AUDIT & STRUCTURAL HARDENING (APRIL 2026)

A multi-session forensic audit across five architectural arcs closed the primary classes of structural fragility that had accumulated since the 2.0 modernization.

**Arc 1 — Atomic Writes.** Three non-atomic write windows were closed: IV-Play.cfg (SettingsManager), the three-file icon atlas (AssetCache), and the database+metadata pair (DatabaseManager). All three now follow write-temp/atomic-rename on the same NTFS volume.

**Arc 2 — Device-Lost State Machine.** The rendering recovery path had three silent failure modes: _isDisposed was not reset on recovery, D2DERR_RECREATE_TARGET was not in the catch filter, and atlas/art/background were not re-uploaded after context recreation. All three are resolved. RenderState (Active/Resizing/Recovering) now gates Render entry, and a _drawBatchOpen flag detects stale BeginDraw before each frame.

**Arc 3 — MachinesSummary Schema Discipline.** The three-site manual synchronization contract (DDL / INSERT / SELECT reader) that had already caused one silent regression (the December 2025–February 2026 Features column omission) was replaced by MachineSummarySchema.Columns—a single descriptor array that drives all four sites. One edit adds a column correctly everywhere.

**Arc 4 — Cache Result Provenance.** CacheResult<T> (record struct, zero heap allocation) replaced null-checks and ad-hoc wasPartial tuples across SoftlistParser, AssetCache, and DatabaseManager. Callers are now required to log when quality is not Fresh.

**EF Core Retirement.** The EF Core / ADO.NET hybrid—the largest structural liability identified in the audit—was fully resolved: both Microsoft.EntityFrameworkCore.* packages removed, three EF context classes deleted, all four using(context) blocks in DatabaseManager converted to raw SqliteConnection + SqliteCommand, CompiledModels/ (twelve generated files) deleted from the repository. The OpenRuntimeConnection() helper now enforces WAL mode and foreign keys on every connection uniformly.

Three-layer crash net (AppDomain.UnhandledException, Application.ThreadException, MainForm constructor try/catch writing timestamped crash logs to %APPDATA%/IV-Play/crash_*.txt) was confirmed present in source during the audit.

The audit established five architectural invariants that apply to all future work: atomic writes for all persistent state; no silent fallbacks (CacheResult<T> required); schema invariants in code (MachineSummarySchema.Columns); no EF Core; device-lost recovery via RenderState state machine.

## STRUCTURAL RE-ORG AND SERVICE EXTRACTION (MAY 2026)

Following the April 2026 forensic audit, a structured architectural review identified several classes whose responsibilities had grown beyond their original scope. The May 2026 re-org extracted those responsibilities into purpose-built service classes and reorganized file placement to match logical ownership. The re-org produced a measurable performance improvement: FilterGamesAsync and Gamelist Filter (RAM) were moved out of the warm-start TTI critical path, reducing warm TTI from 972ms (v2.6.2) to 957ms (v2.8.4) despite 112 additional machines and a newer MAME version.

**New services.** The FileWatcherService / ChangeQueue / BatchProcessor triad implements live ROM availability monitoring without user intervention. FileWatcherService wraps one FSW per ROM directory and one for favorites.ini. ChangeQueue provides thread-safe dedup with a full-rescan sentinel. BatchProcessor drains the queue on a short timer with a stability window and routes each batch to either AvailableListProvider (ROM changes) or DatabaseManager.SyncFavoritesAsync (favorites changes). HiscoreLoader introduces a Hiscore.ini subsystem: load once at startup, sort alphabetically by ROM name on write, lenient parsing of tab/space-separated formats, stub-line preservation.

**Extracted helpers.** InstanceLock extracted the path-derived mutex naming logic from Program.cs into a dedicated class, enabling future named-pipe usage with the same path identity. InternedStringPool extracted the CLR string.Intern() delegation from DatabaseManager's load-time inline code into a shared, null-safe wrapper available to all subsystems. LayoutMetrics (Phase 3) centralized all DPI-aware layout constants at a single design DPI, providing DIPs directly to D2D callers and eliminating scattered conversion arithmetic.

**File relocations.** GameListSearch moved from Controls/ to Classes/ (data class with no UI dependency). SoftwareItem moved from Classes/ to Data/Models/ (peers with other model types). CacheResult and MachineSummarySchema were already in their correct locations from the April audit.

## STARTUP ACCURACY, UPDATE IDENTITY, AND DAT CORRECTNESS (JULY 2026)

A run of individually small corrections, grouped here because they share a cause: several long-standing figures and behaviours turned out to be measuring or reporting something adjacent to what they claimed.

The Available gamelist could stay filtered to favourites after a MAME recompile, because the self-heal rebuild refreshed the data without re-filtering the visible list; fixing it introduced a visible double repaint on cold start, which was then fixed in turn. Machine-type population was running twice per cold start for no benefit. The F1 launch figure measured internal bootstrap completion rather than double-click to first appearance, which is the number a person actually experiences. Check for Updates reported an update against the exact build just downloaded, on release day; identity was moved onto the compiled module id, where equality is exact and neither version strings nor timestamps can drift. The DAT peek showed a parent's history entry for clones that carry their

own — Donkey Kong Japan had been permanently wrong. MAME's softwarelist filter attribute had never been parsed at all, and a description-text heuristic had been covering for data that was simply never read; parsing it retired the heuristic. Machine status colour was reconciled between the gamelist icon border and the Alt-Enter properties view, which had disagreed on 547 machines; both now port MAME's own rule rather than approximating it separately.

## HIGH-DPI HEADROOM AND THE ICON MASTER (LATE JULY 2026)

An audit aimed past the current display generation — 192 DPI and beyond — rather than at the hardware on the desk. It found one live geometry defect, one hard ceiling in the icon atlas, and a set of degradation points that would arrive together on a 6K or 8K panel. All were executed as a single compound pass across seven files.

The durable outcome is a design point rather than a fix: icons are authored as 64-pixel masters and the atlas is built to render them unscaled at the display's own scale factor, so the authored master is never resampled. The first findings sourced from a live field log arrived in the same period, and one of them retired real work — a DPI change was rebuilding several thousand icons to no visible effect.

The build verification suite gained a state guard in the same window, after a run was found to be leaving the live session on the small icon pack. The suite must now leave the operator's settings exactly as it found them, and the guard's own first three failures were its own.

## THE BENCHMARK ENGINE AND UNATTENDED RUNS (LATE JULY–AUGUST 2026)

A request for a command-line switch turned into an extraction: the benchmark had no engine to invoke, only a form. The result is the one-engine, two-front-end structure described in Appendix 3, and it is what made everything after it possible.

The first scheduled runs came back at 30 to 75 percent of a command-line run's speed, and the investigation that followed is the most instructive work in this period because nothing in the codebase was at fault. Time of day, descheduling, wake-to-run and an overclock utility were each proposed and each eliminated by measurement rather than by argument. Two diagnostic fields shipped during the investigation were falsified by their own first run and replaced — one had been probing the wrong process, and one inferred core placement where it could have measured it. The answer was that Windows Task Scheduler was getting the long single-threaded work migrated onto efficiency cores. Opting out of efficiency classification recovered part of the deficit and, on its own, was not enough; pinning to performance cores closed it exactly.

The fix generalized from five machines to the full thirty-six, and scoping the next request exposed a blocker worth recording: the history file stored only the three aggregates, which are nearly blind to one machine of thirty-six halving — the precise regression the investigation had just spent a night chasing would have passed a ten percent check in silence. Per-machine history and a median-baseline regression check followed. Late in the period the runs were also found never to have been isolated from mame.ini; immunity to the display settings had been read as immunity to configuration generally, and it is not.

## CONCURRENCY FORMALIZATION AND CONTRACT CLOSURE (AUGUST 2026)

The fourth and last stage of a formalization ladder applied to the concurrency model, run as findings-only with no code emitted, then closed cluster by cluster. The outcomes are in Appendix 3; the method is the part worth

keeping. Of the findings the pass ranked highest, several did not survive being read against the source: one could not happen as described, one named a gate that was mispositioned in the opposite direction from the one filed, and in one case the defect actually sitting at the site was worse than the one reported. Each was diagnosed against current source before anything was built.

The same period closed the tracking gap that had allowed some of that drift. The bug ledger's own index had been claiming entries it did not cover; the fix separated the curated index of open work from the chronological record without reordering a single entry, since an append-only record that gets reflowed is worth less than the defect it was repaired for. Automated checks now guard the direction that failure ran, and a check proposed for a related class was measured before being written and declined on the numbers: it would have carried a five-in-six false positive rate, and a check that is usually wrong trains the reader to skip checks.

## SEARCH SEMANTICS, SOFTWARE IDENTITY AND REFRESH REPAIR (SEPTEMBER 2026)

The filter example library was audited against the displayed queries and their machine/software result domains. Quoted phrases became literal phrase-anywhere searches. Bare words gained exact machine-driver filename-stem matches while quoted words stayed text-only. Boolean translation was repaired to preserve typed order; the owner confirmed that juxtaposed terms restrict the entire explicit OR group. Software qualifier cleanup was then repaired to retain a parseable expression and preserve quoted search text.

Software-list identity was separated from its launch host across **Ctrl-B**, HiScore, favorites, icons and artwork. Namespaced storage/report keys prevent software from borrowing a same-named arcade item, while the launch path retains the selected host and media configuration. F4/F5 gained view-aware selection restoration and availability repair before filtering after a rebuild, preserving the selected list. Empty favorites collections now expose only Games Only.

## BENCHMARK PROVENANCE AND GUARD VERIFICATION (SEPTEMBER 2026)

CPU model and logical-processor count joined history scope after a CPU replacement under the same host name contaminated comparisons. Benchmark CPU accounting now ends when MAME exits. Frequency telemetry moved to the report-disclosed 250 ms cadence for short games; equal-game averages and the separate whole-run peak clarified its meaning.

## F1, HBMAME AND ICON OWNERSHIP (LATE SEPTEMBER–OCTOBER 2, 2026)

F1 was fitted to smaller working areas with scrollable settings and a fixed footer; its art preview was aligned with the main pane's border/background compositing. The shared ini reader was corrected to follow emulator identity and inipath rather than an arbitrary executable filename, and logs/title/About now distinguish HBMAME from MAME. First-run optional-art noise and startup timing labels were corrected. SQLite moved to 3.53.4 and the log gained the engine version, source identity and loaded DLL path.

F1 now adopts a changed Icons source on OK, carries the selected art folder through sorting/removal, and restores the highlighted game with a view-aware key. Serializing only the final atlas build was found insufficient: archive/source changes and warm-start work could still overlap it. Ownership was extended to those producers, the F1 dialog lifetime, exceptional reload/reset paths and shutdown. These entries record the source changes; the Bug Ledger separately retains outstanding owner build, Native AOT and live acceptance gates.

## LIST ACCURACY AND REFRESH SELECTION (OCTOBER 2026)

The Arcade Only list was checked against the full machine list, with and without catlist.ini. Manufacturer rules had also matched license credits and bootleg labels, and some short names matched inside longer ones, which hid ordinary video games; those rules now match the manufacturer itself and whole names only. Name-based inclusions and exclusions now carry to clones, so a clone follows its parent. Medal, quiz, photo-booth and fruit-machine families that catlist.ini does not classify are now recognized by driver family, title and history text. The Arcade views load catlist.ini before filtering instead of applying the no-catlist rules at startup, and if the file is not yet available the load is retried the next time an Arcade view is filtered.

F5 had restored the game highlighted when the refresh began, so moving through the list during a rebuild was undone when it finished. The selection is now taken immediately before the refreshed lists replace the displayed ones, so the game most recently selected is kept when it is still in the list; otherwise the existing fallback applies.

The Available audit had listed every clone of a present parent and every game whose BIOS was present, and ignored CHDs, so games that could not start were listed. It now decides whether the collection is merged, requires BIOS archives in a merged collection, resolves a board whose ROMs are all inherited to the archive that holds them, and requires a CHD folder for a machine with a disk. A ROM or disk that MAME marks as never dumped is not required, machines that need no file at all are left out, and when the archive names give no sign of how a collection is organized the audit lists a game rather than demand a separate BIOS archive. It remains a name-level audit that reads no file contents.

## KEY BUG ARCHIVE

The following table records significant bugs that shaped the architecture. It is a historical record, not a current issue tracker.

| **Category** | **Description** | **Resolution** |
|---|---|---|
| Stability | Application crashed on exit after DirectX migration. | DirectX resources now disposed synchronously on the UI thread in the correct order. |
| Stability | NullReferenceException on startup after the God Object refactor. | DirectX initialization now tied to the OnHandleCreated event. |
| Performance | Scroll judder during fast navigation. | Entire rendering pipeline migrated from GDI+ to DirectX. |
| Performance | Multi-second bottleneck during type-to-search. | Search re-architected to query a lightweight in-memory list rather than full Machine objects. |
| Rendering | 1-pixel transparent seam around art panel border on high-contrast backgrounds. Attacked across both rendering stacks from July 2025; five GDI+ techniques failed outright before the DirectX migration. | Aliased border draw calls, plus physical-pixel quantization of the art rect and the border stroke. Not temporary and not cosmetic: the border brush is translucent by default, which forecloses art-under-border overlap, and the off-screen composite route was abandoned — |

| Category | Description | Resolution |
|---|---|---|
| | | so pixel-grid adjacency is the only mechanism holding this closed. Quantization binds to the D2D context DPI, never DeviceDpi. An alignment tripwire logs any grid residual, because a 90% border pixel already carries background and the seam is invisible on low-contrast wallpaper: not seeing it is not evidence of alignment. See Docs/IV-Play Bug Ledger.txt before altering any of this. |
| Navigation | "Stuck on S" — left-arrow navigation stuck or jumping erratically. | Letter-jump navigation logic fixed; sort key handling corrected. |
| Data | Hide Non-Working produced incorrect game counts vs. MAME. | Data model and parser corrected to align with MAME's internal logic (attributes vs. feature tags). |
| Schema | Missing Features column in MachinesSummary CREATE TABLE silently forced full EF Core hydration fallback on every warm start from December 2025–February 2026, adding ~157ms. | Features TEXT column added to the CREATE TABLE statement in DatabaseManager.cs (v2.4.14). F8 reset could not fix it—the bug was in the code, not the data. |
| Software Lists | Entering a software list crashed due to duplicate names. | Software items de-duplicated by name before dictionary insertion. |
| Software Lists | Games within a software list failed to launch. | Argument construction logic corrected to generate the proper MAME command (e.g., a2600 -cart combat). |
| AOT | WinForms UIA reflection crash: DataGridViewRowHeaderCell parameterless constructor trimmed by the AOT linker. | Resolved via AOT-specific annotations. EF Core compiled model generation added for AOT-safe database access. |

## APPENDIX 5 - PERFORMANCE DELTA TIMELINE

This appendix is IV/Play's longitudinal performance record. Every major changed release adds a new entry. Nothing is removed or rewritten. The intent is to track how the startup model evolves across MAME versions, hardware generations, and architectural changes. All measurements use the internal StartupProfiler instrumentation on the reference development system unless otherwise noted.

**Reference hardware (all entries unless noted):** i7-12700K, 64GB DDR4, NVMe SSD, 4K Primary (3840×2160, 144 DPI) + 1440p Secondary (2560×1440).

### ARCHITECTURAL EVOLUTION SUMMARY

The Time-to-Interactive (TTI) on a warm start evolved as follows across the 2025–2026 modernization period:

| Date / Milestone | TTI (Warm Start) | Key Architectural Change |
|---|---|---|
| July 1, 2025 | ~2.9s | Baseline before major refactoring. |
| July 5, 2025 | ~5.0s | Regression after initial un-optimized SQLite migration. |
| July 29, 2025 | 497ms | “Pure Speed” architecture: binary gamelist cache. Fast but produced ghost entries and GC spikes. |
| Sept 28, 2025 | 195ms | Peak raw speed. Relied on potentially stale cache; ~250K duplicate string objects; GC stutter 5–10s post-launch. Discarded. |
| December 2025 | ~1,040ms* | Perfect Paint architecture. Binary cache retired; live ADO.NET query. Fast Path broken by missing Features column—full EF Core hydration fallback on every warm start. *Measured reality, not documented target. |
| February 2026 (v2.4.14) | 906ms | Fast Path restored (Features column fix). .NET 10. String interning active. 13% improvement over broken state. |
| March 2026 (v2.6.2) | 972ms | Native AOT. DAT data in filter scope (+263ms FilterGamesAsync, intentional trade-off). Still sub-1-second Perfect Paint. |
| May 2026 (v2.8.4) | 957ms | Structural re-org. FilterGamesAsync removed from TTI critical path. FileWatcherService/ChangeQueue/BatchProcessor triad. InstanceLock, InternedStringPool, LayoutMetrics Phase 3, HiscoreLoader extracted. File relocations (GameListSearch, SoftwareItem). |

## ENTRY 1: V2.4.14 — FEBRUARY 16, 2026

**Software:** IV/Play 2.4.14, MAME 0.285, 42,639 machines, 140,278 software list items, .NET 10.0 (JIT).

**Key change:** Fast Path ADO.NET loader restored. Missing Features column in MachinesSummary CREATE TABLE fixed. String interning active.

**Warm start milestones:** [1] Initial I/O 45ms → [2] UI Interactive 906ms → [3] Full Startup 1,340ms.

**Interactive-phase breakdown (TTI 906ms):** DB Load Summary Raw 341ms (39.6%) • FilterGamesAsync 68ms (7.9%) • Gamelist Filter RAM 50ms (5.8%) • DirectX Init 40ms (4.6%) • Show & Activate UI 12ms (1.4%) • Other/UI Load 350ms (40.7%).

**Deferred tasks (post-UI, 434ms background):** Atlas Valid+Load 244ms (56.2%) • Softlist Index Load 135ms (31.1%) • Art Path/Archive Init 63ms (14.5%) • Favorites Sync 7ms (1.6%).

**Notes:** DB load 71ms slower than the December 2025 design target of 270ms, attributable to data growth (MAME 0.285), string interning overhead, and real-world vs. controlled-test variance. The 906ms TTI represents a 13% improvement over the broken-Fast-Path state (1,040ms).

## ENTRY 2: V2.6.2 — MARCH 17, 2026

**Software:** IV/Play 2.6.2, MAME 0.286, 42,727 machines, 140,360 software list items, Native AOT (.NET 10).

**Key change:** Native AOT. History.xml, MAMEInfo.dat, CatList.ini promoted to first-class filter participants via MetadataHydrator. FilterGamesAsync cost rises from 68ms to 331ms (intentional trade-off for substantially richer search capability).

**Warm start milestones:** [1] Initial I/O ~56ms → [2] UI Interactive 972ms → [3] Full Startup 1.38s.

**Interactive-phase breakdown (TTI 972ms):** DB Load Summary Raw 371ms (26.9%) • FilterGamesAsync 331ms (24.0%) • Gamelist Filter RAM 329ms (23.9%) • DirectX Init 109ms (7.9%) • Show & Activate UI 13ms (0.9%).

**Deferred tasks (post-UI, ~408ms background):** Softlist Index Load 204ms • Atlas Valid+Load 192ms • Art Path/Archive Init 106ms • Favorites Sync 13ms.

**Cold start (one-time per MAME version):** Atlas Build 3.86s • XML Parse & Process 2.58s • DB Write 1.48s • DAT Load History.xml 998ms • DAT Load MAMEInfo.dat 784ms. Total ~12.1s.

**Delta vs. v2.4.14:** DB Load +30ms (data growth) • FilterGamesAsync +263ms (DAT scope) • TTI +66ms • Total Launch +40ms. Perfect Paint guarantee maintained at sub-1-second TTI.

**Notes:**

## ENTRY 3: V2.8.4 — MAY 10, 2026

**Software:** IV/Play 2.8.4, MAME 0.287, 42,839 machines, 140,763 software list items, Native AOT (.NET 10).

**Key change:** Structural re-org. FilterGamesAsync and Gamelist Filter (RAM) removed from warm-start TTI critical path. Live ROM monitoring triad (FileWatcherService / ChangeQueue / BatchProcessor) added. Extracted: InstanceLock, InternedStringPool, LayoutMetrics Phase 3, HiscoreLoader. File relocations: GameListSearch (Controls→Classes), SoftwareItem (Classes→Data/Models).

**Warm start milestones:** [1] Initial I/O ~69ms → [2] UI Interactive 957ms → [3] Full Startup ~1.07s.

**Interactive-phase breakdown (TTI 957ms):** DB Load (EagerInit, T:04) 684ms • DB Load (main thread, overlapping) 502ms • Softlist Index Load 179ms • Settings Load 57ms • DirectX Init 42ms • Background Load 18ms • GetMameInfo 2ms. FilterGamesAsync and Gamelist Filter (RAM) no longer appear in TTI scope.

**Deferred tasks (post-UI, background):** Atlas Valid+Load 246ms • Art Path/Archive Init 100ms • Softlist Index Load 179ms.

**Cold start (one-time per MAME version):** Atlas Build 3.65s • XML Parse & Process 2.52s • DB Write 1.11s • DAT Load History.xml 879ms • DAT Load MAMEInfo.dat 663ms • Internal Ready 10.79s • Launch-to-splash 193ms.

**Delta vs. v2.6.2:** DB Load −32ms (EagerInit overlap) • FilterGamesAsync 0ms in TTI (−331ms from critical path) • TTI −15ms • Cold Atlas Build −0.21s • Cold DB Write −0.37s • Cold DAT History −119ms • Cold DAT Mameinfo −121ms. Perfect Paint guarantee maintained. Machines +112. Software items +403.

**Notes:** The dual DB-load pattern (EagerInit on T:04 + main-thread load on T:01) is expected and reflects the overlapping parallel load architecture; the profiler records both independently. DB Load wall time is longer than v2.6.2 (684ms vs 371ms EagerInit) because both paths run to completion before the profiler deducts overlap; effective TTI impact is the longer of the two (~502ms main-thread path).

## USER GUIDE VERSION HISTORY

| Date | Version | Document |
|---|---|---|
| Sunday, December 17, 2006 | Rev. a | Concept Doc |
| Tuesday, June 26, 2007 | Rev. b | Design Doc |
| Saturday, April 09, 2011 | Rev. c | Design Doc |
| Sunday, May 26, 2011 | 1.0 | User Guide |
| Sunday, June 12, 2011 | 1.5 | User Guide |
| Sunday, October 16, 2011 | 1.5.3 | User Guide |
| Wednesday, November 02, 2011 | 1.5.5 | User Guide |
| Sunday, November 20, 2016 | 1.7 | User Guide |
| Tuesday, January 15, 2019 | 1.8.1.2 | User Guide |
| Saturday, April 17, 2021 | 1.8.4 | User Guide |
| Friday, May 5, 2023 | 1.8.5 | User Guide |
| Thursday, September 30, 2025 | 2.1.1 | User Guide |
| Wednesday, October 15, 2025 | 2.2.2 | User Guide |
| Thursday, November 27, 2025 | 2.4.7 | User Guide |
| Saturday, December 6, 2025 | 2.4.9 | User Guide |
| Friday, December 26, 2025 | 2.4.10 | User Guide |
| Monday, February 23, 2026 | 2.4.14 | User Guide |
| Wednesday, March 25, 2026 | 2.6.2 | User Guide |
| Sunday, May 24, 2026 | 2.8.4 | User Guide |
| Saturday, June 13, 2026 | 2.8.8 | User Guide |
| Sunday, August 2, 2026 | 2.8.10 | User Guide |
| Friday, October 2, 2026 | 2.8.12 | User Guide |
| Saturday, October 10, 2026 | 2.8.14 | User Guide |

## CREDITS

**Creator & Designer (2006-Present): John L. Hardy IV**
**Legacy v1 Codebase (2011–2016): Matan Bareket**