M-FILES CORPORATION

# DISTRIBUTING VAULT-SPECIFIC REGISTRY SETTINGS VIA VAULTS ON M-FILES SERVER

VERSION 1.5

LAST UPDATED 21 MARCH 2024

# Contents

# 1. Introduction

M-Files behavior can be adjusted to better match your requirements. In addition to server-level settings, you can have an effect on the vault functionality with various vault-specific settings and other data entries. Instead of separately applying client-side settings to each client computer, use the **Custom Vault Data** section in M-Files Admin to distribute these settings. The settings are applied to the client computers upon user login. These settings are stored in the named value storage of the vault on M-Files Server.

The settings can be saved to the vault as JSON objects, and when a user logs in, M-Files reads the settings from the vault on the server and writes them to the Windows registry of the client computer. This offers system administrators an efficient way of distributing client-side settings to all the necessary client computers.

This document gives system administrators instructions on using **Custom Vault Data** to distribute vault-specific registry settings via vaults on M-Files Server.

## 1.1 Prerequisites

Please make sure that your M-Files software meets these minimum requirements:

| M-FILES PRODUCT | VERSION |
|---|---|
| M-Files Desktop | M-Files Online June '19 Update or M-Files 2018 Update 1906 |
| M-Files Server | M-Files Online June '19 Update or M-Files 2018 Update 1906 |

# 2. Named Value Storage

*Named value storage* is a server-side mechanism for storing name-value pairs (settings) for modifying the behavior of custom applications. As the storage is shared between all the applications on the server, the name of the named values is based on a namespace-name pair for better isolation.

To add JSON objects to the named value storage, use the **Custom Vault Data** section in M-Files Admin.

# 3. Specifying Settings In M-Files Admin

This section first gives a brief introduction to the JSON syntax that is used for storing named values. Then, it tells you how to add, edit, and remove named values in M-Files Admin. JSON stands for JavaScript Object Notation. It is an open-standard, lightweight, and easily human-readable data-interchange format.

## 3.1 JSON Syntax

The named values are stored in JSON format with this syntax:

```
{
    "Subobject": {
```

```
        "Setting name": {
            "Value name": Value data
        }
    }
}
```

The syntax has four components:

| COMPONENT | DESCRIPTION | EXAMPLE |
|---|---|---|
| **Subobject** | This node refers to one of these two registry keys that include the client-specific settings:<br>• *Common*<br>• *MFShell*<br><br>**Note:** The vault-specific subkey (such as *Sample Vault*) is not visible in the JSON object. The *M-Files Client* service adds it automatically. | `MFShell` |
| **Setting name** | This node is the name of the setting. It is shown in the Windows registry as a key under the target vault. For example, *SearchBar* under *Sample Vault*. | `SearchBar` |
| **Value name** | This node is the name of the value. It is shown in the Windows registry as a key under the setting key. | `DefaultSearchWithinThisFolder` |
| **Value data** | This node is the data of the value.<br><br>The data type of the JSON value defines the type of the resulting registry value as described in the table below. Make sure that the registry value is compatible with the current registry settings format.<br><br>JSON DATA TYPE / JSON EXAMPLE / REGISTRY VALUE TYPE:<br>String / `"Value"` / REG_SZ<br>Array / `["ValueA", "ValueB"]` / REG_MULTISZ<br>Boolean / `true` / REG_DWORD<br>Number (use integers only) / `6` / REG_DWORD<br><br>If you want to remove a value, enter *null*:<br>`"Value name": null`<br><br>To modify the *(Default)* value of the registry key, enter `""` as the JSON name, a colon, and your value. For example:<br>`"": "Value"` | `1` |

**Table 1:** JSON syntax components.

These components have their counterparts in the Windows registry editor. The red arrows in Figure 1 show the components used as examples in Table 1.

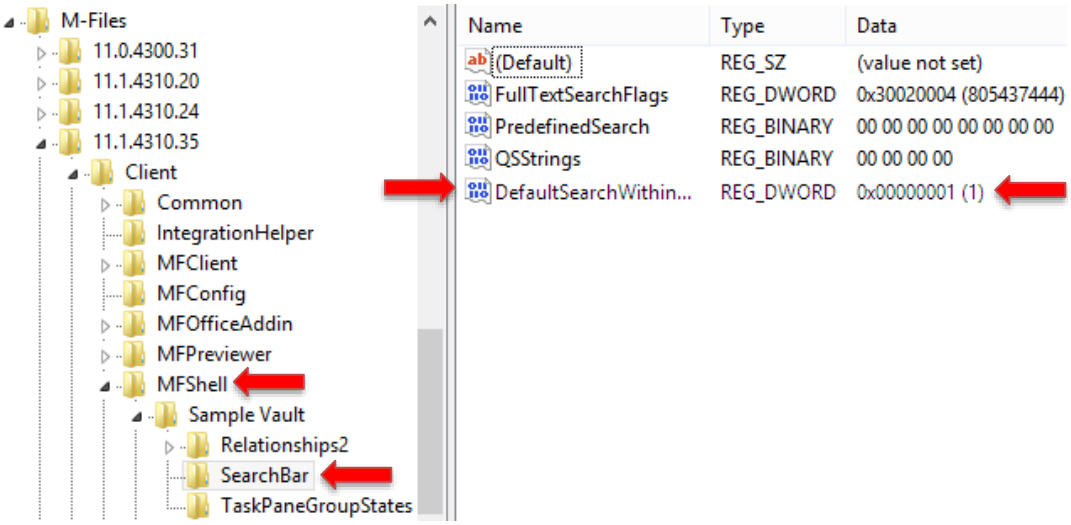


**Figure 1:** The counterparts of the JSON syntax components in the Windows registry.

To make sure that no previous settings exist on the client computer, use the suffix `\\DELETE_KEY_RECURSIVELY`. It tells M-Files to delete a key from the registry. It is recommended to place the deletions in a separate value in the named value storage with a name that precedes other values in the sorting order. This way, the key deletions are applied first.

Example of a named value that deletes all pre-existing preferences:

```
{
  "Common": {
    "AutoFillingOfProperties\\DELETE_KEY_RECURSIVELY": { }
  }
}
```

## 3.2 Adding, Editing, and Removing Settings

The **Custom Vault Data** section in M-Files Admin lets you add, edit, and remove name-value pairs of a vault's named value storage.

To open *Custom Vault Data*:

1. Open M-Files Admin.
2. In the left-side tree view, expand a connection to M-Files Server.
3. In the left-side tree view, expand **Document Vaults**, and then expand a vault.
4. Still in the left-side tree view, select **Configurations**, and then in the gray navigation area, expand **Custom Vault Data**.

For instructions on registering a namespace and adding named values, see section 3.2.1. To remove a named value, click the minus icon (-) on the right side of the named value row and click **Save**.

### 3.2.1 Adding a Name-Value Pair

To add a name-value pair:

1. In M-Files Admin, register a namespace. For instructions, refer to Configuring Custom Vault Data in the M-Files user guide. Use these values:
    - **Storage Type:** *MFConfigurationValue*
    - **Namespace:** *M-Files.Core.Client.Settings*
        - The values in the namespace are processed in a case-sensitive alphabetical order. This establishes a deterministic order for how potentially overlapping or otherwise conflicting settings are applied on the client side.
2. In M-Files Admin, add a named value under the registered namespace. Use the syntax given in section 3.1 for the value. For more instructions, refer to Configuring Custom Vault Data in the M-Files user guide.

**Result**: When a user logs in to the vault, the classic M-Files Desktop reads the settings from the vault on the server and writes the corresponding entries to the vault-specific locations in the client computer's registry under these keys:

| | |
|---|---|
| **Key** | `HKEY_CURRENT_USER\Software\Motive\M-Files\<version>\Client\Common\<vault>` |

| | |
|---|---|
| **Key** | `HKEY_CURRENT_USER\Software\Motive\M-Files\<version>\Client\MFShell\<vault>` |

The name-value pairs are processed in case-sensitive alphabetical order. Make sure that the keys on the list are ordered correctly.

# 4. Change History

The table below describes the essential changes by document version.

| VERSION | DATE | ESSENTIAL CHANGES |
|---|---|---|
| 1.0 | 2015-08-20 | Initial published version. |
| 1.1 | 2015-10-13 | MFNamedValueManager updated to version 1.0.2 |
| 1.2 | 2016-10-28 | Links to external documents and applications updated. |
| 1.3 | 2017-01-23 | Added information about which JSON data types generate which types of registry values (ch. 3.1). |
| 1.4 | 2018-08-22 | Fixed the broken JSON reference. |
| 1.5 | 2021-01-28 | Updated the instruction to point to Custom Vault Data instead of Named Value Manager. Changes throughout the document. |
| 1.6 | 2024-03-21 | Unnecessary reference removed. |

# 5. Reference Documents

- W3Schools: JSON - Introduction
- M-Files Named Value Manager