Application Mode

Application Mode is a feature that transforms the way users interact with remote Windows applications. This feature provides a native web application-like experience while streamlining and enhancing the user interface.

Application Mode has been carefully designed to create a user experience akin to that of a native web application, deviating from the conventional Windows (or Linux) environment. When Application Mode is enabled, the session excludes common Windows/Linux UI elements, such as the taskbar and Start Menu, focusing the user's attention purely on the application. This maximizes the use of the screen's real estate, as the remote application will be started in a fullscreen/maximized mode.

> It is still possible to use multiple applications within Application Mode. The behavior is still more refined and less flexible than multitasking in Desktop Mode.

To be able to use Application mode, you must first create and configure an Application Launchpad. As Frame offers two different Launchpad types, so do we also offer two different types of session modes (Application and Desktop mode). Desktop mode is, as suggested, a mode where a full OS Desktop experience is desired (Along with all of the multitasking features and interfaces).

Launchpad and Session modes might seem similar at first, but there are important distinctions between the two.

Application Mode vs. Launchpads

While both Application Mode and Frame's Launchpads focus on delivering a curated and enjoyable experience, they are different in terms of their operational scope.

Launchpads are client-side interfaces that have no impact on the remote environment. They focus on providing a tailored user experience at the browser or FrameApp level. Launchpads help you navigate the Frame feature set as an end-user with a user experience based on the desired workflow and Session Mode (Application or Desktop).

In contrast, Application Mode alters the remote Windows and Linux environments to mimic the experience of using a native web application. The objective of Application Mode is to blur the

distinction between a remote legacy application and a native web application, resulting in a seamless, immersive experience for the user.

Through Frame's Application Mode, we aim to make remote application use simpler, more intuitive, and more efficient. As we continue to enhance this mode, our goal is to make your workflow smoother and more productive.

If you wish to dig further into what Launchpads can do (and more about what they are), refer to the Launchpad overview page in our docs.

Enhanced Security

In addition to its core features, Application Mode bolsters the security of your remote environment. This includes disabling many default Windows file explorer behaviors, such as right-click context menus and non-client area behaviors, ensuring that user interaction is confined to the application's parameters. By doing so, Frame deters activities outside the intended application experience. These behaviors are not present by default in Desktop Mode (Exclusive to Application Mode).

Custom Taskbar and Start Menu

As mentioned earlier, Application Mode removes the traditional Windows taskbar and Start Menu. The latest version of Application Mode (App Mode 2.0) adds a few features which also include a custom, Frame-branded taskbar and Start Menu. These are designed to simplify task management and provide an uncluttered, user-friendly interface. Their configuration is also flexible, meaning you can adjust and personalize your workspace.

Taskbar Override Configuration

To take things further with the new taskbar and start menu, you can customize the taskbar programmatically through a JSON configuration file. This JSON file provides the settings for overriding the default behavior of the custom taskbar and start menu. The flexibility of the customization is currently set on specifying which apps should appear in the menu, the order of those applications, and also a folder/subfolder hierarchy for all applications.

This override process outlines and controls the structure and content of the taskbar. With this file, you can organize your taskbar to align with your work style, improving your overall efficiency.

You can find this JSON file in the `C:\ProgramData\Nutanix\Frame\Config` directory. The filename is `server_launchpad_override.json`.

> The Config directory does not exist by default, so make sure you create the directory before attempting to create the server_launchpad_override.json if you are attempting this override with a script.

An example of what this file could look like is the following:

```
{
  "name": "root",
  "order": 0,
  "applications": [
    {
      "name": "Frame Explorer",
      "path": "C:\\Program Files\\Frame\\FrameExplorer\\FrameExplorer.exe",
      "icon_url": "https://next-cpanel-
dev.s3.amazonaws.com/images/icons/frame_explorer.png",
      "order": 0
    },
    {
      "name": "Notepad",
      "path": "C:\\Windows\\system32\\notepad.exe",
      "icon_url": "https://next-cpanel-dev.s3.amazonaws.com/images/icons/notepad.png",
      "order": 2
    }
  ],
  "folders": [
    {
      "name": "Folder between two applications",
      "order": 1,
      "applications": [
        {
          "name": "Command prompt",
          "order": 0,
          "path": "C:\\Windows\\System32\\cmd.exe",
          "working_directory": "C:\\"
        },
        {
          "name": "Browser",
          "order": 2,
          "path": "C:\\Program Files (x86)\\Google\\Chrome\\Application\\chrome.exe"
        },
        {
          "name": "Dizzion Website",
          "order": 3,
          "path": "C:\\Program Files (x86)\\Google\\Chrome\\Application\\chrome.exe",
          "arguments": "https://www.dizzion.com/",
          "icon_url": "https://staging-cpa-6ae2ad7f7ca734e5abe8.s3.us-east-
1.amazonaws.com/images/icons/ce69f714bad94c83ac40f5fb9648308e.png"
        }
      ]
    }
```

```
  ]
}
```

> The "name": "root property, and "order": 0, property at the root level of the JSON file are both required. You must have these in the file for this to work properly. This is basically specifying the “root folder” of the entire hierarchy (Top-most level of the hierarchy).

> Every Application and Folder item that is specified in the corresponding arrays absolutely require both the `order` and name fields.

As you can see in the JSON file above, there are two properties in the JSON file that are relevant to the proper customization of the taskbar and start menu: “applications” and “folders”. Both of these properties are arrays that allow you to add as many items as you would like (Folders allows for as many sub-folders and sub-applications as you would like as well).

For an application object, here is a list of fields that can (or must) be specified:

Required fields

path (string) - This is an absolute path to the application's executable file (.exe, .sh, etc).

Optional fields

arguments (string) - If provided, these arguments will be used as the application’s starting arguments. This can be left empty..

working_directory (string) - If provided it will be used as working directory when starting the application.

icon_url (string) - This is a URL from which to load the application’s icon. If not provided, Frame's service will attempt to use the corresponding Operating System API to fetch the icon associated with the provided executable's path.

> Unlike applications, all folders use the default OS folder icon.

Revision #3
Created 1 October 2025 04:53:54
Updated 13 January 2026 15:19:59 by Dominik Conrad