Wallboard Desktop Broadcast App

Settings and startup parameters

Installation

The MSI package installs the application with `perMachine` scope and requires **elevated privileges**, the installation location is `C:\Program Files\Desktop Broadcaster` folder.

The installer uses custom actions to start the application after the installation or to close running applications during the installation. These actions requires **user context** to run and can be skipped with command line arguments.

The following parameters can be set by command line arguments

Parameter	Type	Default	Description
install.url	string		Wallboard server URL
install.key	string		Installation key that can be generated on the target server
install.force	bool	false	Force the application to move to the given server URL
install.start	bool	true	Start the application after the installation - requires user context
install.close	bool	true	Close the running application during the installation - requires user context
install.autostart	string	all	Start the application automatically after Windows logon *all* - for all users *user* - for the current user only *none* - for no one
install.appusage	bool	false	Install Application Usage Reporting module
window.alwaysOnTop	bool	false	Whether the application appears in the topmost Z-order
window.canMinimize	bool	true	Whether the application can be minimized

Parameter	Type	Default	Description
window.canMove	bool	true	Whether the application can be moved
window.canExit	bool	true	Whether the user can exit from the application
window.showInTaskbar	bool	true	Whether the application is shown in the taskbar
window.position	enum	fix	Base position of the window *fix* - fix location *docked* - dock the window
window.size.width	int	0	Width of the Window Default - width of the working area / 2
window.size.height	int	0	Height of the Window Default - height of the working area / 2
window.alignment.horizontal	enum	center	Horizontal alignment of the window *left*, *center*, *right*, *stretch*
window.alignment.vertical	enum	center	Vertical alignment of the window *top*, *center*, *bottom*, *stretch*
window.alignment.offset.x	int	0	Offset - x axis
window.alignment.offset.y	int	0	Offset - y axis

Settings defined by command line arguments are saved to `C:\Program Files\Desktop Broadcaster\setup.json` encrypted JSON file.

Example of passing command line args to the installer

```
msiexec /i DesktopBroadcaster.Installer.msi install.url=[SERVER_URL] install.key=[INSTALL_KEY] install.appusage=true
```

Settings store

Settings are stored in the `C:\Users\[username]\AppData\Roaming\DesktopBroadcaster\settings.json` encrypted JSON file.

Force settings by command line arguments

Application settings can be forced by passing command line arguments directly to the `DesktopBroadcaster.exe`. It can be useful for testing specific settings.

> **WARNING** - Settings that are defined by command line arguments cannot be changed during runtime

Parameters are the same as `Window` installation command line arguments

Parameter	**Type**	**Default**	**Description**
window.alwaysOnTop	bool	false	Whether the application appears in the topmost Z-order
window.canMinimize	bool	true	Whether the application can be minimized
window.canMove	bool	true	Whether the application can be moved
window.canExit	bool	true	Whether the user can exit from the application
window.showInTaskbar	bool	true	Whether the application is shown in the taskbar
window.position	enum	fix	Base position of the window *fix* - fix location *docked* - dock the window
window.size.width	int	0	Width of the Window Default - width of the working area / 2
window.size.height	int	0	Height of the Window Default - height of the working area / 2
window.alignment.horizontal	enum	center	Horizontal alignment of the window *left*, *center*, *right*, *stretch*
window.alignment.vertical	enum	center	Vertical alignment of the window *top*, *center*, *bottom*, *stretch*
window.alignment.offset.x	int	0	Offset - x axis
window.alignment.offset.y	int	0	Offset - y axis

Example of starting application with command line arguments

```
DesktopBroadcaster.exe window.alignment.horizontal=right window.alignment.vertical=bottom
window.alignment.offset.x=-10 window.alignment.offset.y=-10 window.canMove=false
```

Javascript API

Desktop Broadcaster API is available on the `winClient` window object.

Apply application settings from content script

```
winClient.setWindowSettings({
    "alwaysOnTop": false,
    "canMinimize": true,
    "canMove": true,
    "canExit": true,
    "showInTaskbar": true,
    "alignment": {
        "horizontal": "center",
        "vertical": "center",
        "offset": {
            "x": 0,
            "y": 0
        }
    },
    "position": "fix",
    "size": {
        "width": 0,
        "height": 0
    }
});
```

Show/Hide application settings from content script

```
winClient.show();
winClient.hide();
```

Async Javascript Binding (experimental)

Async Desktop Broadcaster API is available on the `DesktopBroadcasterApi` window object, all functions return a `Promise` that can be awaited.

```
(async () => {
    await CefSharp.BindObjectAsync("DesktopBroadcasterApi");

    await DesktopBroadcasterApi.show();
    await DesktopBroadcasterApi.hide();

    await DesktopBroadcasterApi.setWindowSettings({
        "alwaysOnTop": false,
        "canMinimize": true,
        "canMove": true,
        "canExit": true,
        "showInTaskbar": true,
        "alignment": {
            "horizontal": "center",
            "vertical": "center",
            "offset": {
                "x": 0,
                "y": 0
            }
        },
        "position": "fix",
        "size": {
            "width": 0,
            "height": 0
        }
    });
})();
```

Apply application settings from the Web UI

Use the `Send user command` on the WebUI

```
{
    "type": "SetSettings",
    "data": {
        "window": {
            "alwaysOnTop": false,
            "canMinimize": true,
            "canMove": true,
            "canExit": true,
            "showInTaskbar": true,
            "alignment": {
                "horizontal": "center",
                "vertical": "center",
                "offset": {
                    "x": 0,
                    "y": 0
                }
            },
            "position": "fix",
            "size": {
                "width": 0,
                "height": 0
            }
        }
    }
}
```

Start external application by app URI

To start an external application from a content, you have to set the interactive widget click type to `Start application` and set the URI at the `Package name` parameter.

mailto

```
mailto:<user email>
```

sip

```
sip:<user email>
```

msteams

For more information, please visit Create deep links in Microsoft Teams.

```
msteams:/l/chat/0/0?users=<user email>
```

browser

```
https://google.com
```

Application Usage Reporting

Foreground applications can be scanned and reported periodically to the server. This reporting module is not installed by default, it can be enabled by passing `install.appusage=true` command line arguments to the installer.

If `install.appusage=true` is not defined during the installation, the required DLL files won't be installed.

The following information is uploaded

- process filename
- file description property
- product name property
- foreground time

Device information reporting

This module reports application environment information to the server, that can be viewed on the `Web UI / Screen Info` menu.

- Application version
- Target architecture (x86/x64)
- Release / Debug version
- Computer name
- Last application start time

Additionally, it collects more system information from `WMI` and `HKLM registry`, that can be disabled by the `report.systemInfo` setting or via command line arguments.

- OS version
- Firmware / BIOS Version
- Computer model information
- Native resolution
- CPU information
- RAM
- Storage information
- Network information
- .NET Framework version

CEF CrashDumps

CEF CrashDumps are located under `C:\Users\[username]\AppData\Local\CrashDumps` by default.

For more information, please visit CrashReporting.