Tray

Class: Tray

> *Add icons and context menus to the system's notification area.*

Process: Main

`Tray` is an EventEmitter.

```
const { app, Menu, Tray } = require('electron')

let tray = null
app.whenReady().then(() => {
  tray = new Tray('/path/to/my/icon')
  const contextMenu = Menu.buildFromTemplate([
    { label: 'Item1', type: 'radio' },
    { label: 'Item2', type: 'radio' },
    { label: 'Item3', type: 'radio', checked: true },
    { label: 'Item4', type: 'radio' }
  ])
  tray.setToolTip('This is my application.')
  tray.setContextMenu(contextMenu)
})
```

Platform limitations:

- On Linux the app indicator will be used if it is supported, otherwise `GtkStatusIcon` will be used instead.
- On Linux distributions that only have app indicator support, you have to install `libappindicator1` to make the tray icon work.
- App indicator will only be shown when it has a context menu.
- When app indicator is used on Linux, the `click` event is ignored.
- On Linux in order for changes made to individual `MenuItem`s to take effect, you have to call `setContextMenu` again. For example:

```
const { app, Menu, Tray } = require('electron')

let appIcon = null
app.whenReady().then(() => {
  appIcon = new Tray('/path/to/my/icon')
  const contextMenu = Menu.buildFromTemplate([
    { label: 'Item1', type: 'radio' },
    { label: 'Item2', type: 'radio' }
  ])

  // Make a change to the context menu
  contextMenu.items[1].checked = false

  // Call this again for Linux because we modified the context menu
```

```
  appIcon.setContextMenu(contextMenu)
})
```

- On Windows it is recommended to use `ICO` icons to get best visual effects.

If you want to keep exact same behaviors on all platforms, you should not rely on the `click` event and always attach a context menu to the tray icon.

`new Tray(image, [guid])`

- `image` (NativeImage | string)
- `guid` string (optional) *Windows* - Assigns a GUID to the tray icon. If the executable is signed and the signature contains an organization in the subject line then the GUID is permanently associated with that signature. OS level settings like the position of the tray icon in the system tray will persist even if the path to the executable changes. If the executable is not code-signed then the GUID is permanently associated with the path to the executable. Changing the path to the executable will break the creation of the tray icon and a new GUID must be used. However, it is highly recommended to use the GUID parameter only in conjunction with code-signed executable. If an App defines multiple tray icons then each icon must use a separate GUID.

Creates a new tray icon associated with the `image`.

Instance Events

The `Tray` module emits the following events:

Event: 'click'

Returns:

- `event` KeyboardEvent
- `bounds` Rectangle - The bounds of tray icon.
- `position` Point - The position of the event.

Emitted when the tray icon is clicked.

Event: 'right-click' *macOS Windows*

Returns:

- `event` KeyboardEvent
- `bounds` Rectangle - The bounds of tray icon.

Emitted when the tray icon is right clicked.

Event: 'double-click' *macOS Windows*

Returns:

- `event` KeyboardEvent
- `bounds` Rectangle - The bounds of tray icon.

Emitted when the tray icon is double clicked.

Event: 'balloon-show' *Windows*

Emitted when the tray balloon shows.

Event: 'balloon-click' *Windows*

Emitted when the tray balloon is clicked.

Event: 'balloon-closed' *Windows*

Emitted when the tray balloon is closed because of timeout or user manually closes it.

Event: 'drop' *macOS*

Emitted when any dragged items are dropped on the tray icon.

Event: 'drop-files' *macOS*

Returns:

- `event` Event
- `files` string[] - The paths of the dropped files.

Emitted when dragged files are dropped in the tray icon.

Event: 'drop-text' *macOS*

Returns:

- `event` Event
- `text` string - the dropped text string.

Emitted when dragged text is dropped in the tray icon.

Event: 'drag-enter' *macOS*

Emitted when a drag operation enters the tray icon.

Event: 'drag-leave' *macOS*

Emitted when a drag operation exits the tray icon.

Event: 'drag-end' *macOS*

Emitted when a drag operation ends on the tray or ends at another location.

Event: 'mouse-up' *macOS*

Returns:

- `event` KeyboardEvent
- `position` Point - The position of the event.

Emitted when the mouse is released from clicking the tray icon.

Note: This will not be emitted if you have set a context menu for your Tray using `tray.setContextMenu`, as a result of macOS-level constraints.

Event: 'mouse-down' *macOS*

Returns:

- `event` KeyboardEvent
- `position` Point - The position of the event.

Emitted when the mouse clicks the tray icon.

Event: 'mouse-enter' *macOS*

Returns:

- `event` KeyboardEvent
- `position` Point - The position of the event.

Emitted when the mouse enters the tray icon.

Event: 'mouse-leave' *macOS*

Returns:

- `event` KeyboardEvent
- `position` Point - The position of the event.

Emitted when the mouse exits the tray icon.

Event: 'mouse-move' *macOS Windows*

Returns:

- `event` KeyboardEvent
- `position` Point - The position of the event.

Emitted when the mouse moves in the tray icon.

Instance Methods

The `Tray` class has the following methods:

`tray.destroy()`

Destroys the tray icon immediately.

`tray.setImage(image)`

- `image` (NativeImage | string)

Sets the `image` associated with this tray icon.

`tray.setPressedImage(image)` *macOS*

- `image` (NativeImage | string)

Sets the `image` associated with this tray icon when pressed on macOS.

`tray.setToolTip(toolTip)`

- `toolTip` string

Sets the hover text for this tray icon.

`tray.setTitle(title[, options])` *macOS*

- `title` string
- `options` Object (optional)

    - `fontType` string (optional) - The font family variant to display, can be `monospaced` or `monospacedDigit`. `monospaced` is available in macOS 10.15+ and `monospacedDigit` is available in macOS 10.11+. When left blank, the title uses the default system font.

Sets the title displayed next to the tray icon in the status bar (Support ANSI colors).

`tray.getTitle()` *macOS*

Returns `string` - the title displayed next to the tray icon in the status bar

`tray.setIgnoreDoubleClickEvents(ignore)` *macOS*

- `ignore` boolean

Sets the option to ignore double click events. Ignoring these events allows you to detect every individual click of the tray icon.

This value is set to false by default.

`tray.getIgnoreDoubleClickEvents()` *macOS*

Returns `boolean` - Whether double click events will be ignored.

`tray.displayBalloon(options)` *Windows*

- `options` Object
    - `icon` (NativeImage | string) (optional) - Icon to use when `iconType` is `custom`.
    - `iconType` string (optional) - Can be `none`, `info`, `warning`, `error` or `custom`. Default is `custom`.
    - `title` string
    - `content` string
    - `largeIcon` boolean (optional) - The large version of the icon should be used. Default is `true`. Maps to `NIIF_LARGE_ICON`.
    - `noSound` boolean (optional) - Do not play the associated sound. Default is `false`. Maps to `NIIF_NOSOUND`.
    - `respectQuietTime` boolean (optional) - Do not display the balloon notification if the current user is in "quiet time". Default is `false`. Maps to `NIIF_RESPECT_QUIET_TIME`.

Displays a tray balloon.

`tray.removeBalloon()` *Windows*

Removes a tray balloon.

`tray.focus()` *Windows*

Returns focus to the taskbar notification area. Notification area icons should use this message when they have completed their UI operation. For example, if the icon displays a shortcut menu, but the user presses ESC to cancel it, use `tray.focus()` to return focus to the notification area.

`tray.popUpContextMenu([menu, position])` *macOS Windows*

- `menu` Menu (optional)
- `position` Point (optional) - The pop up position.

Pops up the context menu of the tray icon. When `menu` is passed, the `menu` will be shown instead of the tray icon's context menu.

The `position` is only available on Windows, and it is (0, 0) by default.

`tray.closeContextMenu()` *macOS Windows*

Closes an open context menu, as set by `tray.setContextMenu()`.

`tray.setContextMenu(menu)`

- `menu` Menu | null

Sets the context menu for this icon.

`tray.getBounds()` *macOS Windows*

Returns `Rectangle`

The `bounds` of this tray icon as `Object`.

`tray.isDestroyed()`

Returns `boolean` - Whether the tray icon is destroyed.