CipherLab
User Guide

Airlock Browser

for Android Mobile Computers

Version 1.01

CIPHERLAB CO., LTD.
Website: http://www.cipherlab.com

RELEASE NOTES

Version	Date	Notes
1.01	Feb. 18, 2019	New: **8.8 Use wide view port** New: **8.9 Load with overview mode**
1.00	Jan. 04, 2019	Initial Release

CONTENTS

Chapter 1

INTRODUCTION

Airlock Browser is a modern, feature rich, highly configurable web browser that allows mobile workers to be more productive. It has been specifically designed for ruggedized Android devices. Airlock Browser includes all the features you'd expect in an industrial browser such as hardware barcode reader support, client side JavaScript and CSS execution, URL rules, and remote configuration; as well as some distinctive features, such as passcode and fingerprint access, text to speech, and a fully customizable user interface.

This guide provides information on the usage and configuration of Airlock Browser, and also on developing web-based applications for Airlock Browser.

Airlock Browser for Android is compatible with devices running Android 5.0 (Lollipop) and above.

Chapter 2

OVERVIEW OF THE USER INTERFACE

2.1 LAUNCHPAD

When launching Airlock Browser, the launchpad is displayed. The launchpad contains tiles, which are quick links to web application profiles that you create (see Figure 1). You look at creating web application profiles later in the guide.

Tapping or sliding the launchpad to the left reveals the web content. By tapping the ellipsis button in the bottom right hand corner of the main view, the application bar expands.

There is an address bar, and various toolbar and menu items. Tapping the address bar allows you to enter a URL or a search query.

NOTE: The buttons and menu items in the application bar and tab menu are completely customizable. A device administrator may re-order, remove, or add buttons according to requirements. You see how to achieve this later in this guide.

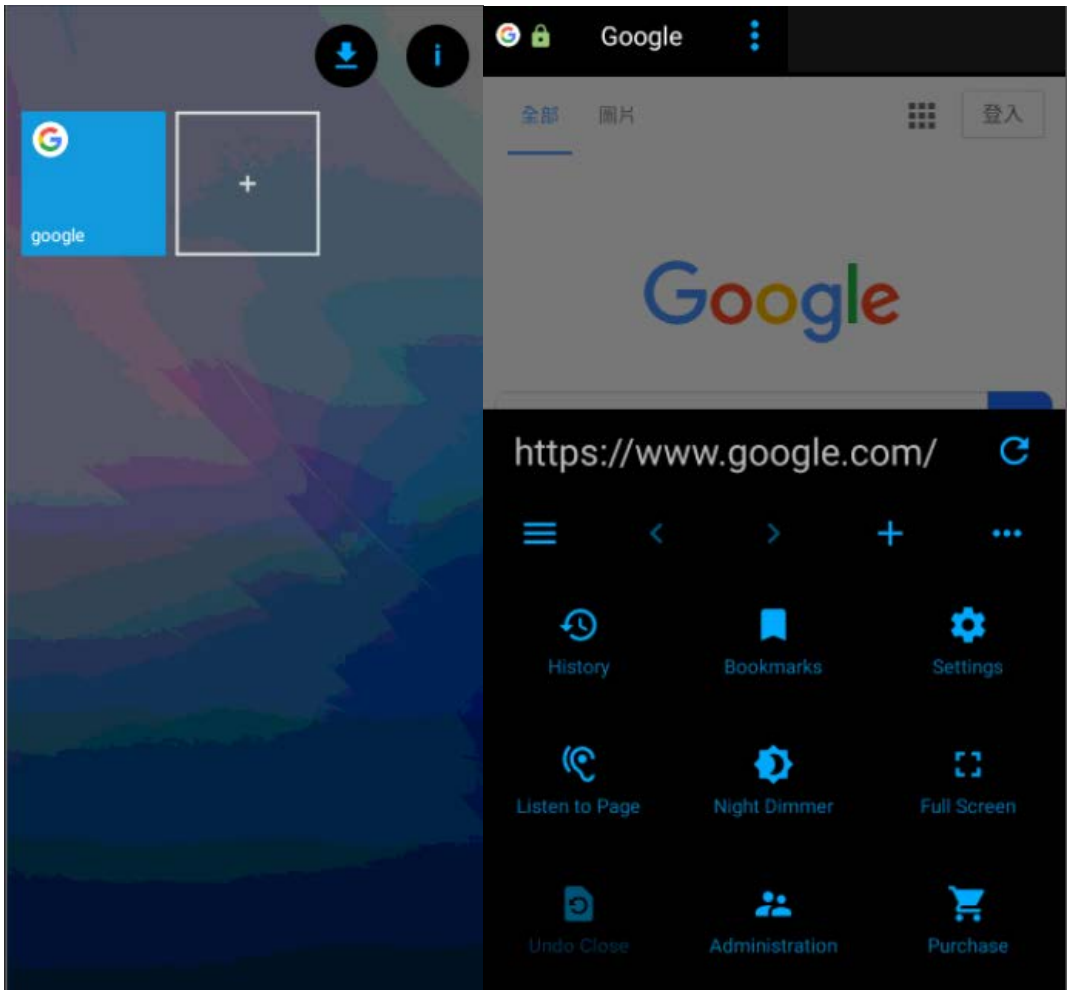


Figure 1, Launchpad and Tabbed Browser side-by-side

2.2 BROWSER TAB MENU

When the application bar is expanded, the tab headers are revealed at the top of the screen. You can switch between tabs by scrolling and tapping the desired tab. A favicon image for the page sits in each tab along with the page title and an SSL indicator that notifies the user of a secure SSL connection. A vertical ellipsis button reveals a set of commands available for the active tab (see Figure 2).

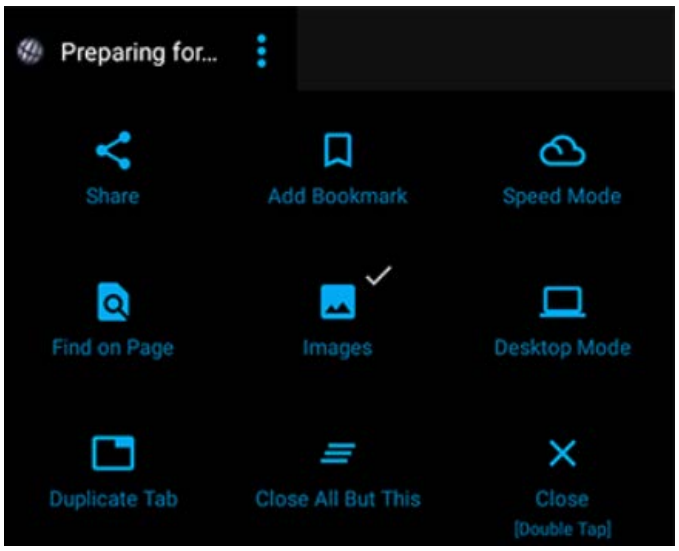


Figure 2, Browser Tab Menu

2.3 SETTINGS SCREEN

Tap the Settings button in the application bar menu to display the Settings screen (see Figure 3).

NOTE: All settings are exportable, as are web application profiles. You explore how to export the browser's configuration, and even host the exported configuration on a server, to deliver browser settings to multiple devices.

The Settings screen offers various settings, and is divided into four sections:

- General Settings
- Security Settings
- Appearance Settings
- Device Settings

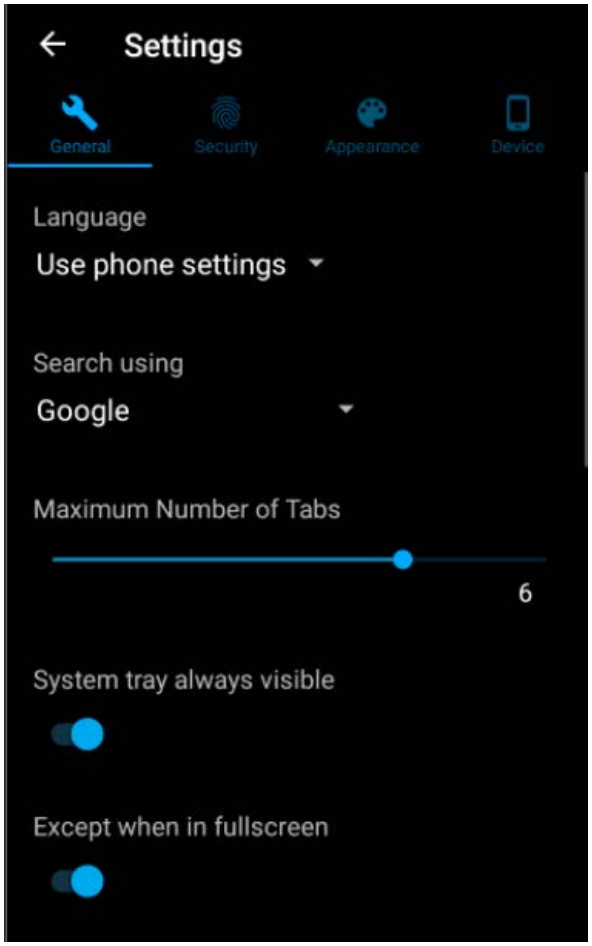


Figure 3, Settings Screen

NOTE: When first launching Airlock Browser, the app is in its non-locked-down state. All screens are visible, all settings are present on the Settings screen, and launchpad tiles are editable. However, upon setting an administration password, most settings are hidden on the Settings screen, launchpad tiles are no longer editable, and the screens available to the user are limited to those that you (as the administrator) allow in the application bar. You can decide which items, if any, are appropriate for your organization's needs and should appear in the application bar and tab menu. We explore these aspects later in this guide.

Chapter 3

CONFIGURING GENERAL SETTINGS

In this section you explore the items in the General section of the Settings screen.

3.1 LANGUAGE

By default Airlock Browser selects its user interface language according to the devices language setting. You can, however, configure the language independently, which enables administration of the device to be performed in one language, while providing Airlock Browser's interface to users of another language.

3.2 SEARCH USING

When you enter text into Airlock Browser's address bar, if the text is unable to be converted to a URL, Airlock Browser performs a search using the specified search provider.

NOTE: Selecting the *none* option from the *Search using* drop down box can disable the address bar search feature.

3.3 MAXIMUM NUMBER OF TABS

When you enter the maximum number of tabs allowed, there would be a limitation to a certain number of new windows that can be opened.

NOTE: If you want your web application to always be opened at the current window, you can select the maximum number to be 1 and Tabs Exceeded Action to be Open in current tab under the application settings in the Launchpad.

3.4 SYSTEM TRAY ALWAYS VISIBLE

You can enable to show the system tray or disable to hide the system tray to provide a fuller screen usage.

3.5 EXCEPT WHEN IN FULLSCREEN

This works corresponding with the System Tray always visible option.

This setting is applied only when System tray always visible is set to true.

3.6 MINIMIZE APPLICATION BAR ON SCROLL

You can enable/disable to show the application bar on scroll.

3.7 SCREEN MODE AT STARTUP

By default, Airlock Browser launches in full-screen mode, hiding the application bar and tab headers. Set the option to *Normal* to have the application bar and tab headers displayed at launch.

NOTE: The user is able to enable or disable full-screen mode using the *Full Screen* item in the application bar menu. To prevent this, remove the Full Screen item from the application bar menu via the *CUSTOMIZE MENUS* item on the Appearance tab of the Settings screen.

3.8 HARDWARE BACK BUTTON ACTION

To perform a back navigation of a web page, the user may either use Airlock Browser's back button in its main toolbar or tap the hardware back button in the operating system's navigation bar. To prevent the hardware back button from navigating to the previous web page, set the *Hardware Back Button Action* to None.

3.8.1 WHEN NAVIGATING AND THE NETWORK IS DISCONNECTED

During the navigation, if the browser detects that the network is disconnected, the user can have the option to either choose Prevent Navigation or to be redirected to an error page.

3.8.2 WHEN A NAVIGATION ERROR OCCURS

Any navigation error occurred under the circumstance that the network is still in connection, the user can have the option to either choose Prevent Navigation or to be redirected to an error page.

3.9 SHOW LAUNCHPAD WHEN APP STARTS

When Airlock Browser launches or is deactivated for 5 minutes, the launchpad is brought into view. To prevent this behavior, disable this option.

3.10 PAUSE BROWSER TAB WHEN INACTIVE

When switching between tabs, Airlock Browser does not stop playback of video on inactive tabs. Enable this option to prevent video playback when a tab is not the active tab.

3.11 SHOW ORIENTATION LOCK

The orientation lock appears when the device is rotated, and allows the user to temporarily prevent Airlock Browser from switching from, or to, landscape or portrait mode. On some devices, where the accelerator is malfunctioning or not properly configured, the orientation lock can be displayed incorrectly. This option allows you to prevent the orientation lock from showing.

3.12 DOWNLOAD IMAGES

This option allows you to disable the downloading of images when a web page loads, which may be useful in low bandwidth environments, or when a web application presents images that have a large file size.

3.13 AD BLOCKER

When enabled, the Ad Blocker stops most web page ads from being displayed.

Chapter 4

CONFIGURING SECURITY SETTINGS

In this section you explore the items in the Security section of the Settings screen.

4.1 PREVENT APP EXIT

Ordinarily, the devices back button may be used to exit Airlock Browser. To prevent the app from exiting via the hardware back button, enable this option.

4.2 REQUIRE PASSCODE AT LAUNCH

When *Require Passcode at Launch* is enabled, the user is presented with a passcode entry screen when the app launches or the app is deactivated for 5 minutes or longer. See Figure 4. This feature is commonly used when the device is being used by staff in a public or unrestricted environment.

When Require Passcode at Launch is enabled, the window for the app is secured. This means that there is no app preview when cycling through apps, and taking a screenshot of the app is disabled.

NOTE: The passcode is included in the exported configuration in an encrypted format. Thus, you are able to control the passcode for multiple devices using a hosted configuration file.

When you set a passcode for the browser, the Require Passcode at Launch is automatically enabled. You can, however, disable Require Passcode at Launch after a passcode is set.

NOTE: Do not confuse the Administration password with the passcode. The Administration password, set on the Administration screen, limits access to authorized administrators only, while the passcode limits access to authorized users only.

Figure 4, Passcode Entry Screen

4.3 UNLOCK USING FINGERPRINT READER

If the device has a hardware fingerprint reader, then Airlock Browser can be unlocked using a fingerprint, without requiring the user to enter the passcode.

NOTE: A passcode must be set to use the Fingerprint Reader unlock feature.

4.4 RESTORE TABS AT STARTUP

When enabled, this option causes Airlock Browser to re-open the pages that were open the last time it was running.

4.5 ALLOW ACCESS TO MY LOCATION

Some websites rely on knowing the GPS location of the device to provide, for example, navigation services. Disabling this option prevents web pages from accessing the location of the device.

4.6 ADD SHARED CONTENT TO HISTORY

When enabled, when sharing a page or image, it is added to the shared content section of the history screen.

4.7 ALLOW DOWNLOADING OF FILES.

When enabled, the user is allowed to download the files through the web app/link.

This would normally be disabled to prevent the violation and comprise of the security.

4.8 ENABLE INCOGNITO MODE AT START-UP

Incognito mode prevents the browser from retaining the history of viewed pages. An incognito button can be placed in the application bar menu by using the menu customization option.

NOTE: When incognito mode is enabled, pop-up windows are disabled.

4.9 ADDRESS BAR MODE

The address bar mode (Normal/Read Only) controls the editability of the address bar field. This setting is combined with the application allowed in the Launchpad to limit the web app and web sites that users are allowed to access.

This is invisible in lock-down mode.

4.10 URL RULES

URL rules determine which pages can or cannot be accessed by the browser. When navigating to a URL the list is evaluated from top to bottom (see Figure 5).

As soon as a match is found, the *Allow* or *Deny* rule is applied. Long press to change the order of a URL rule. Swipe a URL rule left or right to delete it. If you mistakenly delete a rule, use the undo button in the application bar to restore it. To create a new rule, tap the + button in the application bar.

Wildcards are supported by default. You may also specify more complex rules using regular expressions. To enable interpretation of the text as a regular expression, check the *Regular Expression* checkbox. For example, to specify URLs that belong to the example.com domain and all sub-pages you might use something like ^\s*https?://(www.)?example.com/?.*

NOTE: An allow rule is automatically created for the URL of active web application profile (defined in the Launchpad), and it takes precedence over the URL rules defined on the URL Rules screen.

Figure 5, URL Rules Screen

4.11 BOOKMARKS MODIFIABLE

When disabled, it would not be allowed to modify any data saved in bookmark.

4.12 SAVE PASSWORDS FOR WEB PAGES

When enabled, the browser prompts the user to save a password when it identifies a form submission with a password field. If the user agrees to save the password, the password is stored in an encrypted form. When the user navigates to the same page, the browser will attempt to populate the password field.

Disabling this option prevents the saving of passwords.

NOTE: Web page passwords are not included when exporting the browser's configuration.

4.13 DELETE DATA ON EXIT

If enabled, the browser will delete cookies, browsing history, saved passwords, autofill form data and cache if selected when the app exits.

4.14 DELETE DATA NOW

This button deletes all cookies, browsing history, saved passwords, autofill form data and cache right on the click.

Chapter 5

CUSTOMIZING THE BROWSER'S APPEARANCE

In this section you see how to change theme options and customize menu and toolbars via the Appearance section of the Settings screen.

5.1 CUSTOMIZING MENUS AND TOOLBARS

Depending on your organizations needs, you may wish to hide, show, or reorder some items in the browser's main application bar and tab menu.

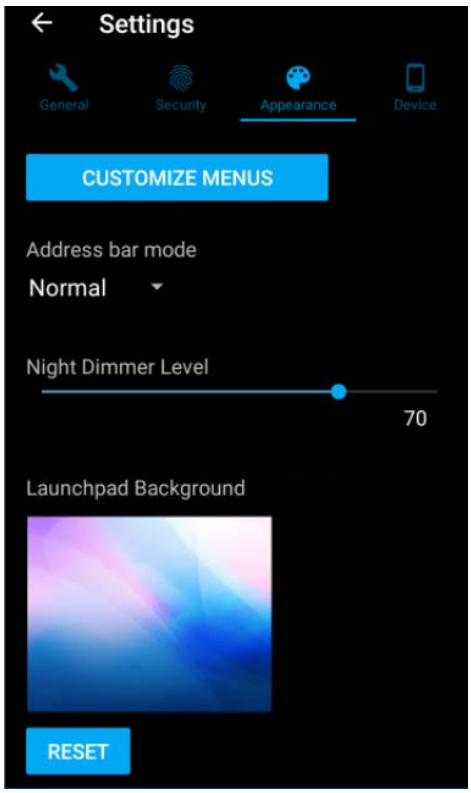


Figure 6, Customize Menus Screen

The items that appear in the main application bar and tab menu are customizable. You can remove items, change their order, and even add new items.

To open the menu and toolbar editor, tap the *Customize Menus* button on the *Appearance* tab of the Settings screen.

There are three menus to choose from:

- Main Toolbar
- Main Menu
- Tab Menu

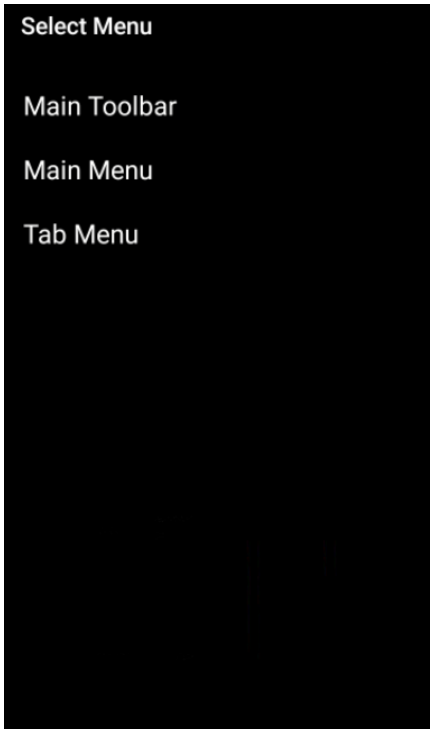


Figure 7, Menu and Toolbar Editor

The main menu and toolbar are located together on the application bar, while the tab menu is located at the top of each browser tab (see Figure 8).

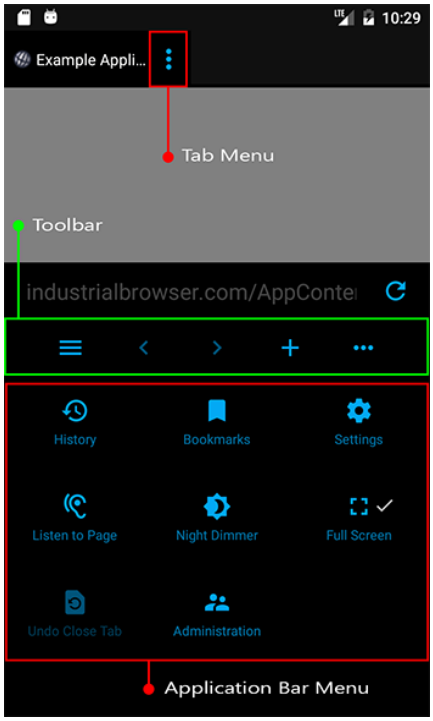


Figure 8, Application bar menu, toolbar, and tab menu

Choosing *Main Toolbar* from the *Select Menu* screen displays the *Main Toolbar* customization screen (see Figure 9).

A checkmark alongside an item indicates that it will be displayed. You can introduce new items to the toolbar, or remove items by unchecking them. Items can be reordered by long pressing and dragging the item to a new position in the list.

There are two required items that cannot be unchecked: the *Expand* button in the Main Toolbar and the *Administration* item in the Main Menu. The Expand button is required so that the user can open the application bar menu to get to the *Administration* button.

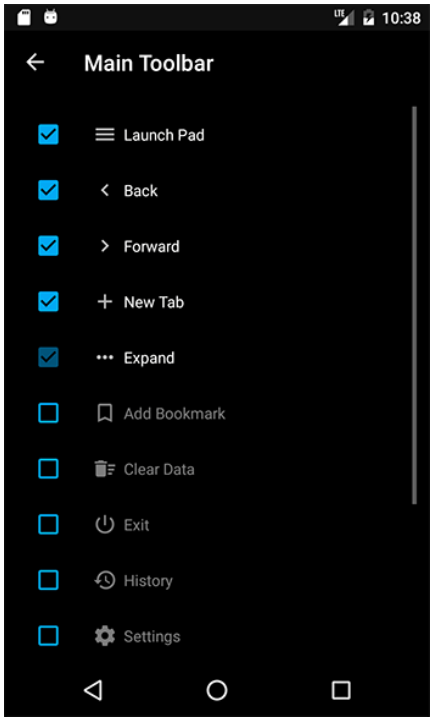


Figure 9, Customizing the toolbar

5.2 NIGHT DIMMER LEVEL

The Night Dimmer Level setting controls how dark the screen is when the Night Dimmer is engaged via the main application bar menu. This setting is visible in lock-down mode to allow it to be adjusted in different lighting conditions.

5.3 LAUNCHPAD BACKGROUND

The Launchpad's background image can be replaced by an image better reflecting your organization's branding.

To select a new image, tap the existing image. A file-picker dialog is displayed.

NOTE: When exporting Airlock Browser's configuration, the Launchpad background image is included in the configuration. The image is optimized to limit the overall file size of the exported configuration file.

To restore the default Launchpad background image, tap the *Reset* button.

5.4 STYLING THE APP WITH A CUSTOM THEME

The following three configurable theme colors allow you to create a look to the app that better represents your organization:

- Browser Tab Background
- Application Bar Background
- and Accent Color

The three color settings also determine the appearance of other UI elements within the app, such as dialog boxes, screen backgrounds, and buttons.

The Browser Tab Background setting determines the theme type of the app. If Browser Tab Background is set to a light color (such as white or yellow), the app adopts a light theme. While a dark tone, results in a dark theme. If a light theme is in place, screen and dialog boxes are white and text is gray. Conversely, a dark theme causes dark backgrounds and white text.

NOTE: Airlock Browser attempts to ensure that text is always readable. For example, if you select a light color (such as white) as your Browser Tab Background, the text color is switched to gray.

NOTE: Selecting white as both the *Accent Color* and the *Application Bar Background* causes the browsers default accent color to be used in some parts of the user interface, where readability would be otherwise compromised.

Chapter 6

USING THE DEVICE SETTINGS TAB

In this section you explore the items on the Device tab of the Settings screen.

6.1 CONFIGURATION FILE URL

Airlock Browser can be configured remotely using an exported configuration file. The file is hosted on a server and can be used to configure multiple devices. We discuss this in greater detail in the following section: *Configuring Lock-Down Mode with the Administration Screen*

Changing the Configuration File URL causes the browser to immediately attempt to download the configuration file and import the configuration. If configuration importation succeeds, a success message is displayed.

NOTE: It is recommended to relaunch the app to ensure that all the app settings are effective to the changes.

6.1.1 IMPORTING CONFIGURATION WITH MDM SOFTWARE

When Airlock Browser is launched, and periodically thereafter, the browser looks for a configuration file located at '*<External Directory>/Enterprise/Browser/Config.brs*' automatically. If the *Configuration File URL* setting has not been modified in the app's setting screen, then the *Config.brs* is automatically applied. By deploying this file using your MDM solution, you can easily configure multiple devices without needing to manually make any changes to your devices. In addition, a different *Configuration File URL* setting may be present in the imported *Config.brs* file. This allows you to instruct Airlock Browser to download its configuration file from a different location from then on.

Before Airlock Browser imports a configuration file it first verifies that the file's last modified date is more recent than the date of the last import. If it is not, the configuration file is *not* automatically imported.

NOTE: Settings from an imported configuration file override any local settings applied using the settings screen.

NOTE: To avoid any confusion, it is recommended to either use Configuration File URL or Automatic Import from '*<External Directory>/Enterprise/Browser/Config.brs*', not both ways at the same time.

6.2 LICENSE SERVER API KEY

The License Server API Key is a value that is assigned to you by Outcoder or a third-party representative. This key allows you to assign, unassign, and reassign licenses for devices within your organization. All devices in your organization should have the same License Server API Key.

You can set the value directly, using the License Server API Key field. Or, you can set this value by placing the key in a file on the device.

The file must be named *ApiKey.txt* and must have the following path: [External Storage Directory]/Enterprise/Browser/ApiKey.txt

TIP: You can find the location of the external storage directory by browsing to the IO example page on your device, and using the *Get External Directory* button.

Once you procure a license for Airlock Browser, there are two ways to apply the license. The first, is to automatically have the license applied by setting the correct License Server API Key on each device. You can then manage and distribute purchased licenses using the Outcoder License Manager.

Alternatively, if your device does not have internet access, you can manually import the license using the Import License button in Administration -> Manage License.

6.3 CONFIGURING DEVICE VENDOR SPECIFIC SETTINGS

Airlock Browser allows configuration of Cipherlab device specific settings, including reader symbologies, scan, notification and output.

NOTE: To avoid any confusion, use ONLY the Device Vendor Specific Settings to configure all the reader related settings within the Airlock Browser.

Any reader related settings configured outside the Airlock browser using ReaderConfig will NOT be effective to the browser.

The *Vendor Configurations* button on the *Device* tab of the Settings screen takes you to the device configuration specific (see Figure 10).

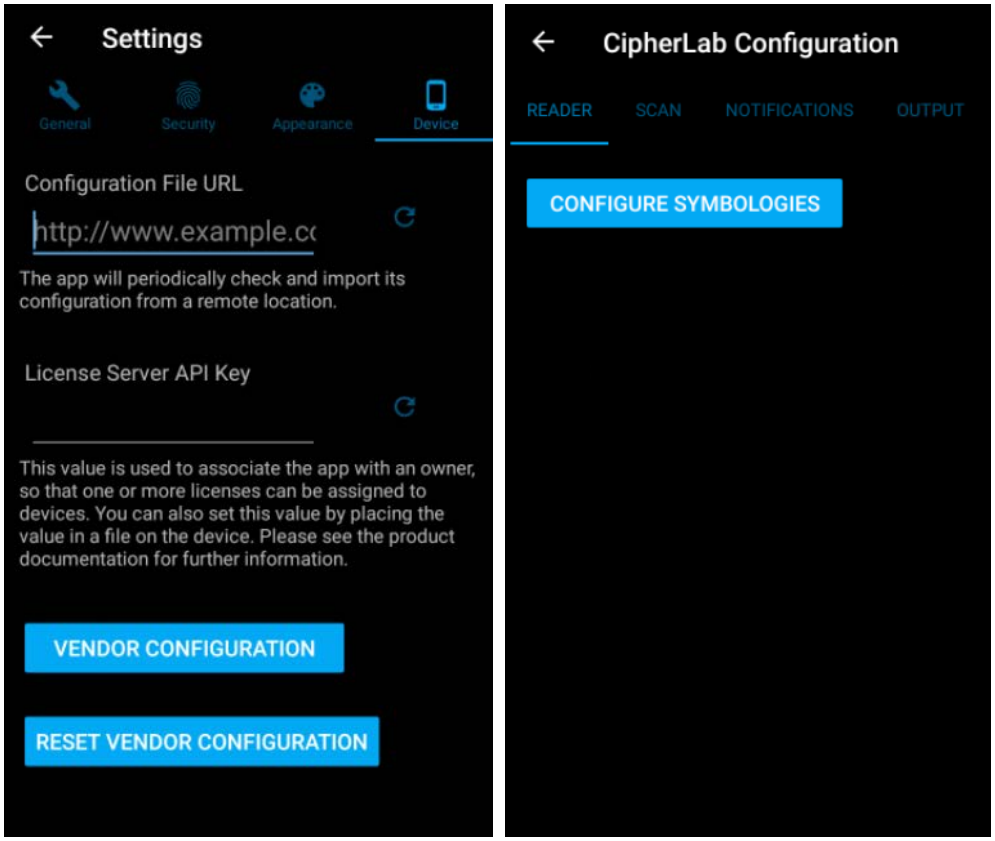


Figure 10, Device Configuration screen

Configure barcode symbologies by tapping the *CONFIGURE SYMBOLOGIES* button. A list of supported symbologies is displayed (see Figure 11).

6.3.1 READER (SYMBOLOGYIES)

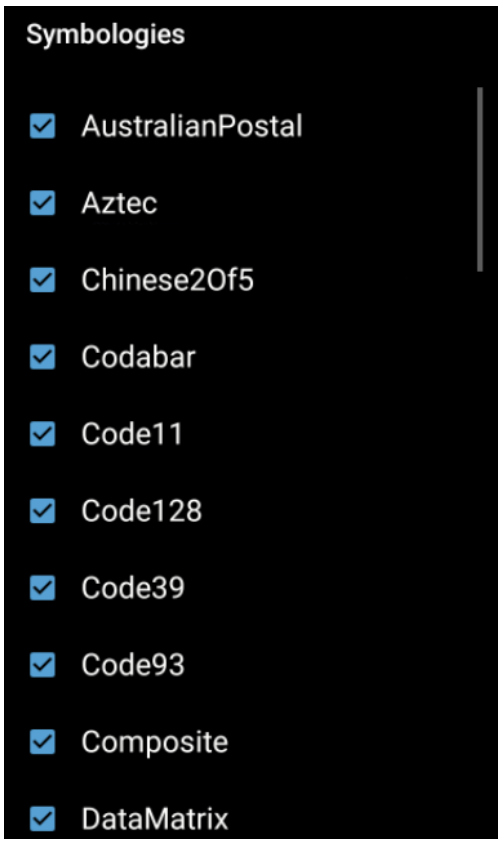


Figure 11, Symbology Selection screen

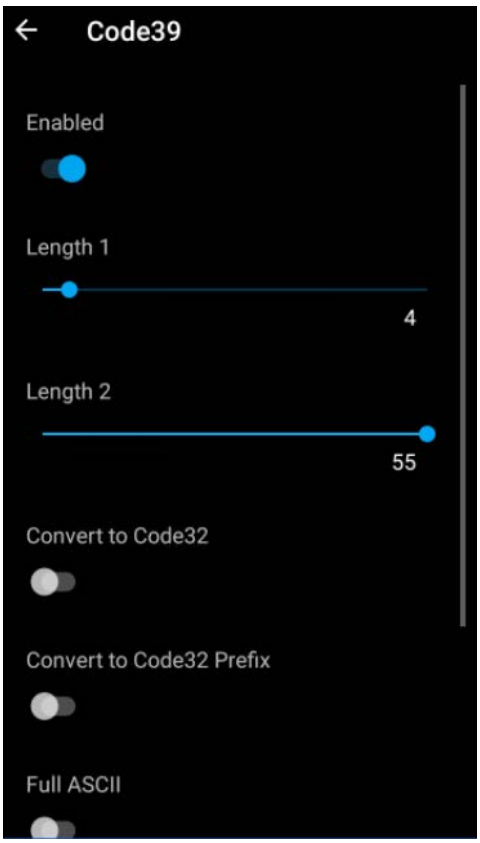


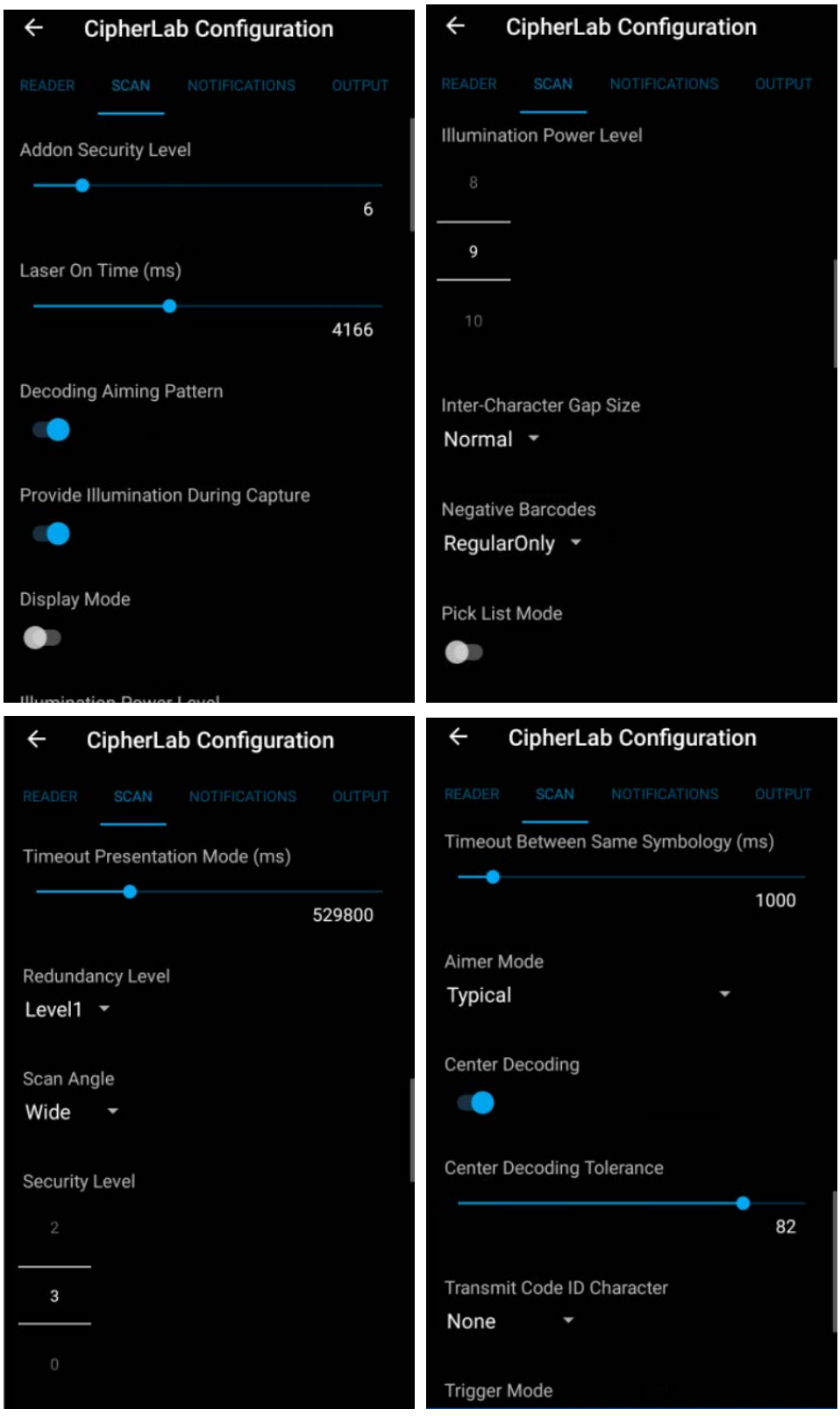
Figure 12, Symbology screen

It may improve barcode scanning performance to disable those symbologies that will not be needed. Tap on a symbology to configure it.

When a symbology is disabled, it appears dimmed-out in the Symbology Selection screen (see Figure 12).

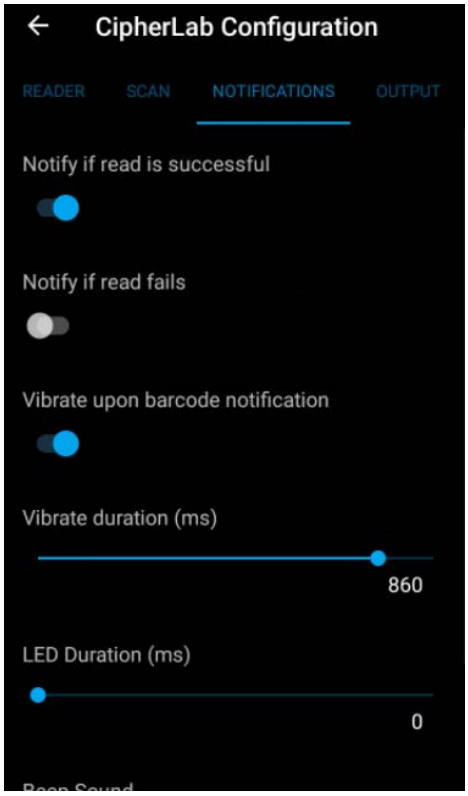
6.3.2 SCAN

This is to configure scanner settings.



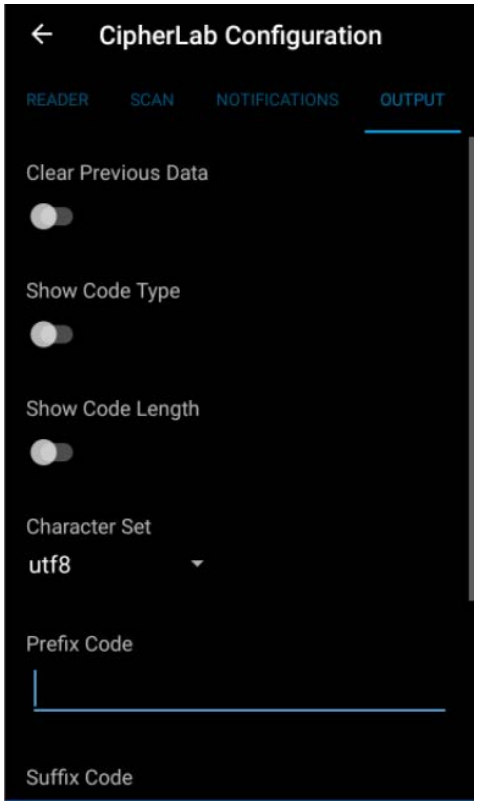
6.3.3 NOTIFICATION

This is to configure the types of notification through vibration, LED or beep during barcode scanning.



6.3.4 OUTPUT

This is to configure how the decoded barcode is shown.



6.4 RESET VENDOR CONFIGURATION.

This button reset the vendor related configuration settings to the default values within the airlock browser.

Chapter 7

CONFIGURING LOCK-DOWN MODE

The Administration Screen provides a step-by-step guide on how to set up the browser for your organization (see Figure 13).

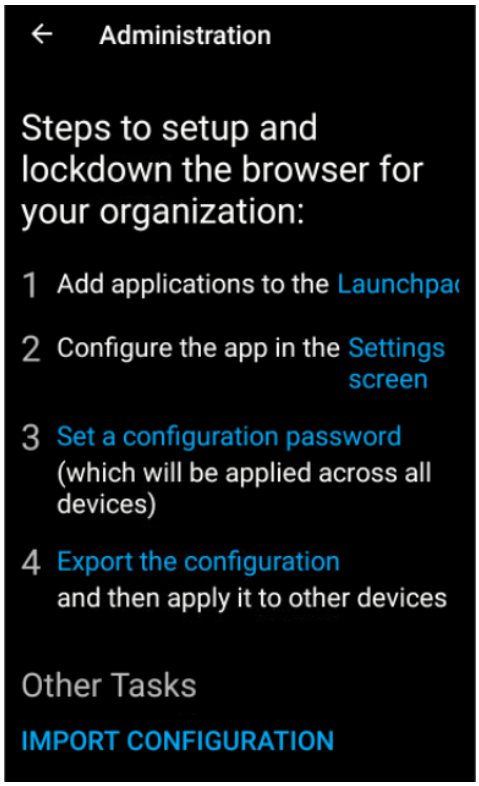
The first two steps direct you to create application profiles on the *Launchpad* and to configure the app via the *Settings* screen.

Step 3, *Set a configuration password* is important because, once set, Airlock Browser will require the user to enter the password to access the Administration screen. Other parts of the application are also placed in lock-down mode, such as the Settings screen; which displays only a subset of the settings.

NOTE: You may also decide to disable the Settings menu item in the application bar. In this case, the only way to open the Settings screen is via the Administration screen after entering the configuration password.

The final step, 'Export the configuration' saves the settings and web profiles to a file. The configuration file can then be imported to another device using the 'Import Configuration' item on the Administration screen. Alternatively, the configuration file can be hosted on a web server and automatically imported using the 'Configuration file URL' option on the Device tab of the Settings screen.

If you specify the 'Configuration file URL' option, Airlock Browser periodically checks for an updated configuration and automatically imports the configuration, thereby allowing you to easily configure multiple devices at once.



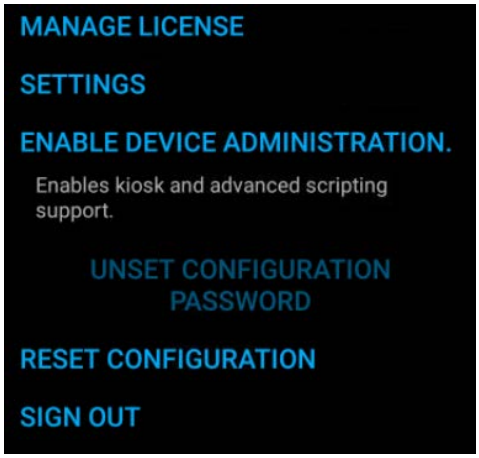


Figure 13, Administration Screen

Tapping the Manage License item displays the Manage License screen (see Figure 14).

NOTE: The Manage License screen is used to manually import a license and is designed for scenarios where internet access is not available. The preferred method for applying licenses is with the Licium License Manager. The Licium License Manager (LLM) provides an easy way to distribute licenses for Outcoder products such as Airlock Browser. LLM allows you to assign, revoke, or reassign licenses to devices within your organization. In addition, it provides you with a bird's eye view of products installed on devices within your organization, and gives you back control over license distribution.

The Manage License screen allows you to export the serial number of the product, which can be provided to Outcoder or your supplier to procure a license. You can use the Save button, to save the serial number to a file. The *Copy* button copies the serial to the devices clipboard. While the *Share* button allows you to select another app, such as an email client, to send the serial number to yourself or a third-party.

Once a license has been procured, you can import the license using the *Import* item on the Manage License screen.

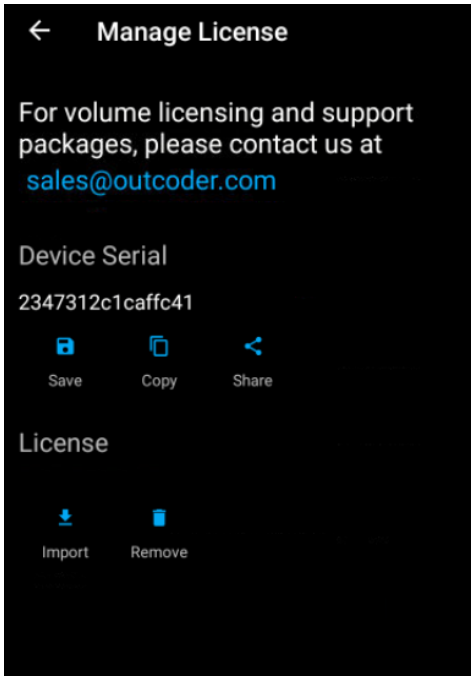


Figure 14, Manage License Screen

Chapter 8

CREATING A WEB APPLICATION PROFILE IN LAUNCHPAD

Web application profiles (AKA applications) represent web sites. They allow you to define the behavior of Airlock Browser when navigating to pages within your organization. Each application is represented as a tile on the launchpad. Tapping on a tile applies the settings for that application and opens the Web Address in a new browser tab.

To create a new web application profile, tap the + tile on the launchpad (see Figure 15).

The Web Address edit box defaults to the URL of the current active tab.

To edit an existing web application profile, long press its tile in the launchpad. A menu appears allowing you to edit the tile or delete the tile. When a tile is deleted from the applications section in the launchpad, it is deleted from the device.

The web application profile includes a *Title* field, the *Web Address* (URL), a tile color option, and several other options that we now explore.

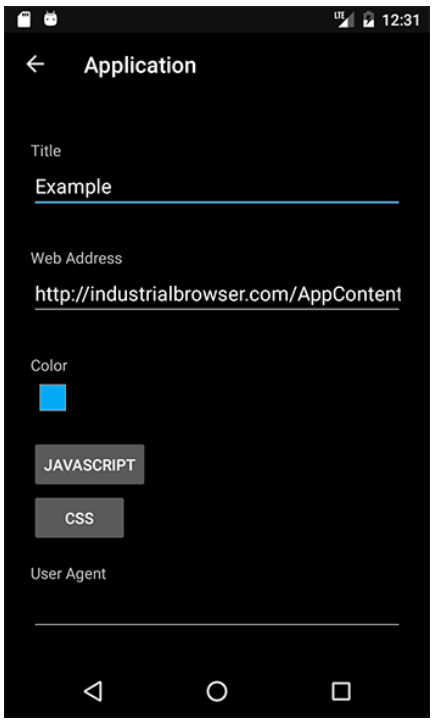


Figure 15, Web Application Profile Screen

8.1 INVOKING CUSTOM JAVASCRIPT

The Web Application Profile *JavaScript* option allows you to edit or create JavaScript that is executed when, for example, the web page loads or when a barcode scan occurs.

NOTE: For in-depth coverage of Airlock Browser's JavaScript APIs, please refer to Chapter 12 JavaScript.

The list of JavaScript items is presented on the JavaScript screen (see Figure 16). Each item displays the title of the script and the event when the JavaScript is invoked on the web page.

New JavaScript items are created using the + button in the application bar. To delete a JavaScript item, swipe it left or right. If you mistakenly delete an item, use the undo button in the application bar to restore it.

Figure 16, JavaScript List

Tapping a JavaScript item displays the JavaScript editor (see Figure 17).

The following are the three available execution events:

- On Page Load
- Before Barcode insert
- After Barcode scan

If *On Page Load* is selected, the script is invoked on the web page as soon as the page has loaded.

If *Before Barcode insert* is selected, the script is invoked before text is inserted into a field.

NOTE: When *Before Barcode insert* is selected, the Population Method *Emulated Keyboard* option must be enabled in the Web Application Profile settings for the script to be invoked.

NOTE: To receive a notification when a barcode scan occurs, you can also use the *On Page Load* option to subscribe to the airlock.scanning.onScan event. The onScan event is described later in this guide.

If *After Barcode Scan* is selected, the script is invoked after a barcode scan is performed, regardless of the *Emulate Keyboard* option in the Web Application Profile settings.

JavaScript may be entered directly into the JavaScript field. Alternatively, use the Import button in the application bar to browse for a JavaScript file. This allows you to edit the file with the convenience of a desktop editor.

When importing a file, you are given the option to append its content to the JavaScript field or replace the content in the JavaScript field.

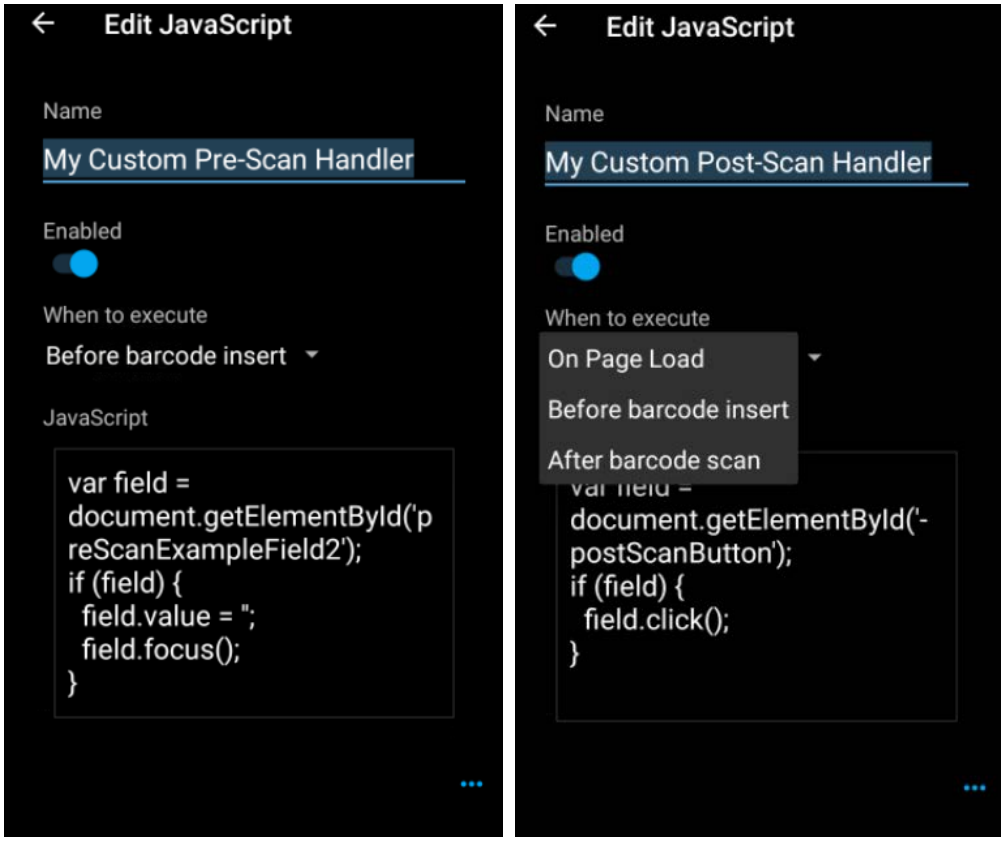


Figure 17, JavaScript Editor

8.2 HANDLING BARCODE SCAN EVENTS

Barcode data is provided as an immutable object to client-side JavaScript. The variable is named *scanData* and has the following fields:

- BarcodeData
  A string representing the barcode value.
- SourceScanner
  The ID of the scanner. E.g. 'Internal Image Scanner'
- Symbology
  The recognized symbology. E.g. Ean8
- Timestamp
  A string indicating when the scan occurred.
- BarcodeLength
  The number of characters present in the barcode data.
- KeyboardWedgeEnabled
  Indicates whether the currently active field is to be populated with the barcode data.
- InsertMode
  If the Emulate Keyboard is enabled, this field indicates if the barcode data will replace or be prepended or appended to the active text field.

8.3 ADDING CLIENT-SIDE CSS TO PAGES

If you have a legacy web application that was not designed for a mobile device, you may apply custom CSS to the page to improve its appearance and usability. Tap the CSS button on the Web Application Profile screen to display the CSS editor.

When a page loads, the CSS is applied.

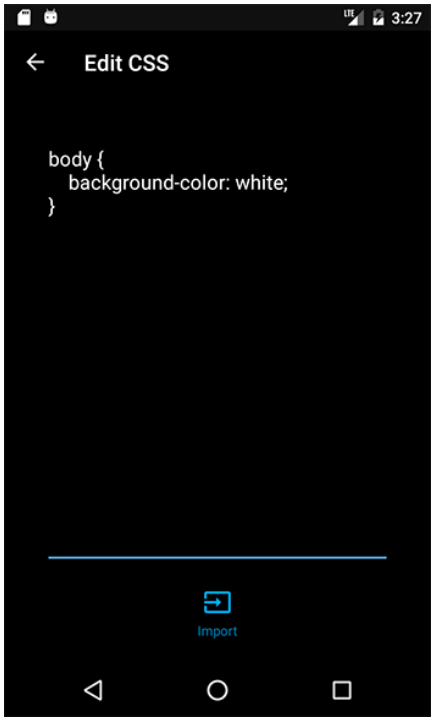


Figure 18, CSS Editor

8.4 APPLYING A CUSTOM USER AGENT

Web applications often use a device's User Agent to determine the capabilities of the device and how to render content. Sometimes it is useful to impersonate a different device to coerce the web application to render content in a particular manner.

To achieve this in Airlock Browser, paste a custom user agent into the User Agent field. The user agent is applied for the web application profile's browser tab. Each web application profile's user agent setting is applied independently.

8.5 LIMITING SCREEN ROTATION

In some scenarios it may make sense to limit the automatic rotation of the screen to portrait or landscape orientation. Some pages may not render well in landscape (or portrait). To lock the orientation for web application, set the *Screen Rotation* option to either *Lock Portrait* or *Lock Landscape*. By default, Screen Rotation is set to *Dynamic*, meaning that the device is free to rotate the screen depending on the orientation.

8.6 IMPROVING TEXT READABILITY

Sometimes web pages that were originally designed for the desktop and not for mobile devices may have text that is too small to read. To increase (or decrease) the size of the text for a page, use the *Web Page Text Size* option on the Application screen.

8.7 USER CAN ZOOM

This controls whether to allow users to zoom in/out on a web page within a web application.

8.8 USE WIDE VIEW PORT

Sets whether browser should enable support for the "viewport" HTML meta tag or should use a wide viewport.

8.9 LOAD WITH OVERVIEW MODE

Sets whether browser loads pages in overview mode, that is, zooms out the content to fit on screen by width.

8.10 INSERTING SCANNED TEXT INTO A FIELD

In Airlock Browser, when a scan completes using the hardware barcode reader, the barcode text is automatically pushed into the currently selected field of the web page. If you wish to disable this function, or if you require greater control over this capability and wish to rely solely on custom JavaScript for handling barcode events, you can disable the Emulate Keyboard function by setting the *Scan Population Method* option on the Application screen to None.

When *Scan Population Method* is set to *Set Text*, the active field content is set to the scanned barcode text using JavaScript. When *Scan Population Method* is set to *Emulate Keyboard*, keyboard keys are emulated on the web page, simulating keyboard presses by the user.

For more information of JavaScript API about "Scan Population Method", please refer to http://outcoder.com/Products/AirlockBrowser/Scripting/V2/JSDoc/Airlock/airlock.scanning.html.

8.10.1 UNDERSTANDING SCAN INSERT MODE

The *Scan Insert Mode* setting allows you to control the placement of text into the active input field. If set to *Replace*, scanned barcode text replaces all content within the active input field. If set to *Prepend*, barcode text is inserted at the beginning of the active input field. *Append* sees the barcode text appended to the existing content of the active input field.

8.10.2 PREFIXING AND POSTFIXING SCANNED DATA

Use the *Prefix scanned data with* and *Postfix scanned data with* options to prefix and postfix the scanned data with any text.

8.10.3 EMULATING KEYS WHEN A SCAN OCCURS

In addition to the *Prefix* and *Postfix* options, you can also emulate keyboard presses when a barcode is scanned. The *Emulate keys on scan* or *Emulate keys after scan* options allow you to define a set of space delimited key codes, which are sent to the web page as simulated key presses.

Emulating keys allows you to control, for example, navigating to a different field on the page after a scan; without requiring the use of JavaScript.

The *on scan* keys are emulated before scanned text has been set or emulated; and *after scan* keys are emulated after scanned text has been set or emulated.

NOTE: *Emulate keys on scan* and *Emulate keys after scan* options are independent of the *Prefix and Postfix* options. This is especially evident when using the *Set Text* population method. When *Emulate keys on scan* is used in combination with *Set Text*, the emulated text will be removed from the active input field.

See here for a complete list of available key codes.

8.11 ALLOWING LEGACY PAGES TO LAUNCH POPUPS IN THE SAME TAB

When Airlock Browser has reached its maximum number of allowed tabs (as defined in the settings), if a web page attempts to use JavaScript to open a page in a new window, Airlock Browser opens the page in the current tab. This is especially useful in lock down scenarios where the maximum number of tabs has been set to 1. To disable this behavior, set the *Tabs Exceeded Action* to *None*.

8.12 INTERACTING WITH THE BROWSER VIA ON-PAGE JAVASCRIPT

In addition to Airlock Browsers client-side JavaScript, you can also interact with the browser using on-page JavaScript. Airlock Browser offers a JavaScript API that is consumable from web pages to respond to power events, monitor network availability, and handle barcode scanning events.

Chapter 9

USING THE HISTORY SCREEN

The History screen displays a searchable list of pages URLs, grouped by day (see Figure 19).

To delete the history items, tap the bin icon in the application bar.

NOTE: To prevent access to the History screen, uncheck the *History* item on the *Main Menu* screen, located at Settings -> Appearance -> Customize Menus -> Main Menu.

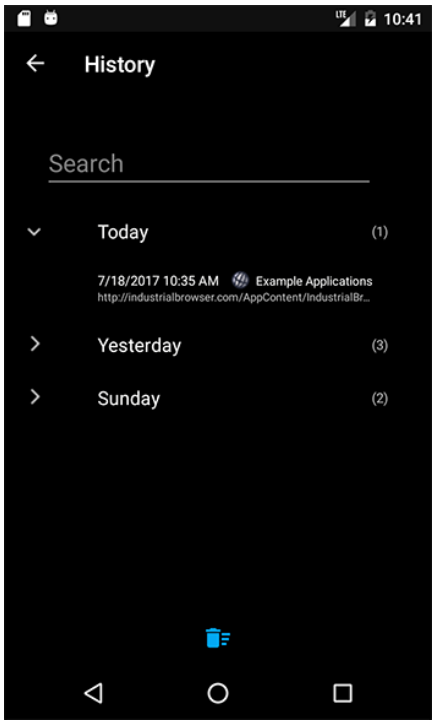


Figure 19, History screen

Chapter 10

CREATING AND EDITING BOOKMARKS

The *Bookmarks* screen allows the user to edit bookmarks, delete bookmarks, and to create new bookmark folders (see Figure 20). Tapping a bookmark closes the Bookmarks screen and opens the bookmark in a new browser tab.

To delete bookmarks or folders, select the list icon in the toolbar. Checkboxes appear next to each bookmark and folder, allowing you to delete individual bookmarks or entire folders. If you mistakenly delete a bookmark or a bookmark folder, use the undo button, in the application bar menu, to restore the item or items.

NOTE: To prevent access to the Bookmarks screen, uncheck the *Bookmarks* item on the *Main Menu* screen, located at Settings -> Appearance -> Customize Menus -> Main Menu.

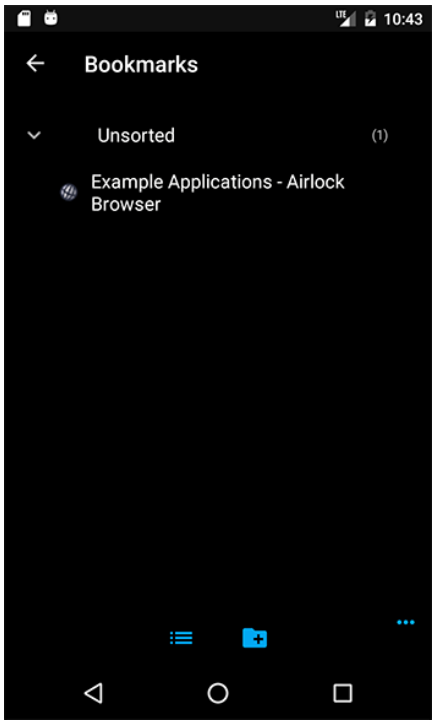


Figure 20, Bookmarks screen

To create a new bookmark, select *Add Bookmark* from the main screen's tab menu. The *Add to Bookmarks* dialog allows you to enter a name for the bookmark, which defaults to the page title. The URL is also editable.

You can choose to place the bookmark into an existing folder within the bookmarks screen, or leave the bookmark in the unsorted folder.

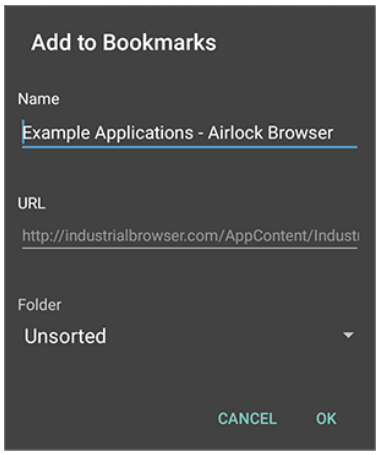
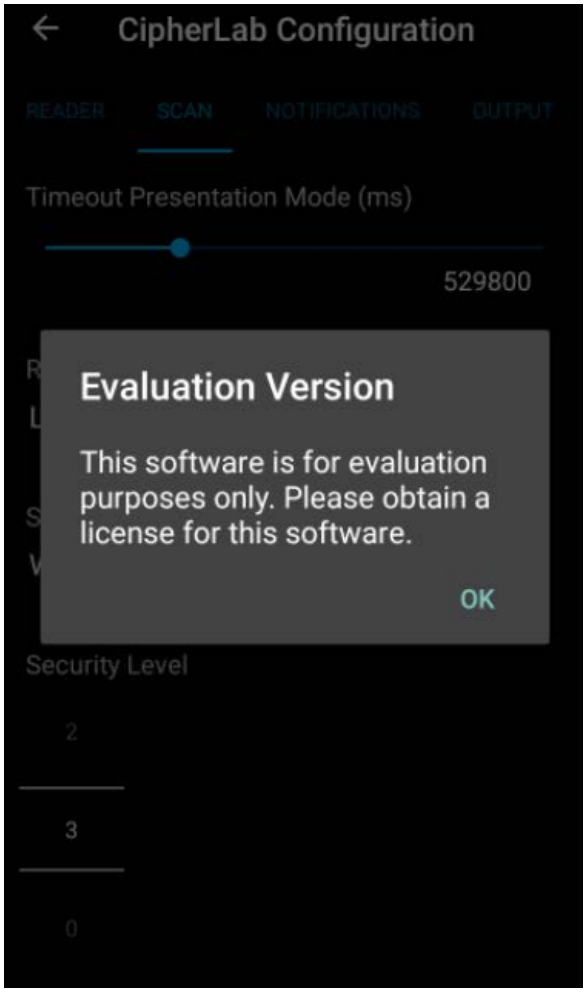


Figure 21, Add to Bookmarks dialog

Chapter 11

PURCHASING AIRLOCK BROWSER

Please contact Outcoder or Cipherlab for quotation of the Airlock browser licenses. Full functions of the Airlock Browser will be available for evaluation and testing purpose, however, an evaluation version reminder message will be shown randomly in an unlicensed Trial Airlock Browser.



Chapter 12

JAVASCRIPT

Airlock JavaScript APIs let you embed the Browser functions in your own webpages, such as retrieving system information as well as manipulating scan engine(s), buzzer and vibrator.

12.1 CUSTOMIZED JAVASCRIPT

The browser provides a build-in JavaScript class for HTML pages to access various functions; no plugin call is required. Please refer to the JavaScript topic of this document.

Example:

```
<script type="text/javascript">
airlock.onReady("handleAirlockReady()");
function handleAirlockReady(){
   alert("Initialization success");
}
</script>
```

12.1.1 CALLBACK FUNCTION FOR SCAN OUTPUT NOTIFICATION

BarCodeSetCallBack	
Purpose	When the barcode reader successfully scans a barcode and has its output ready, the JavaScript callback function will be called so that the web page can process the data.
Syntax	**function BarCodeSetCallBack(***jsFunctionName***);**
Parameters	jsFunctionName is a string of the JS callback function name.

Example

```
<script type="text/javascript">
airlock.onReady("handleAirlockReady()");

function handleAirlockReady(){
    alert("Initialization success");
}

BarCodeSetCallBack("handleBarcodeScan");

handleBarcodeScan(args){
   var text=args.text;
   var name=args.symbologyName;
   var times=args.timestamp;
   //or
   var value=BarCodeGetReaderData();
}
</script>
```

12.1.2 GET THE READER OUTPUT DATA

BarCodeGetReaderData

Purpose	When the reader scans a barcode, and a callback is received, the scanned data can be obtained.
Syntax	**function BarCodeGetReaderData();**
Parameters	none
Return Value	Reader data in string
Example	Please refer to section 5.1.1

12.2 DEVICE API

12.2.1 BROWSER FULL SCREEN MODE

JSFullScreenMode

Purpose	Set the screen mode.
Syntax	**function JSFullScreenMode(***mode***);**
Parameters	*mode*

integer	
1	FullScreen
0	Show title bar

Return Value	True if success, false if error.

Example	`<script type="text/javascript">`
	`  var ret = JSFullScreenMode(1);`
	`</script>`

12.2.2 MINIMIZE BROWSER

JSMinimizeBrowser

Purpose	Minimize the browser.
Syntax	**function JSMinimizeBrowser();**
Parameters	none
Return Value	True if success, false if error.

Example	`<script type="text/javascript">`
	`var ret=JSMinimizeBrowser();`
	`</script>`

12.2.3 BRIGHTNESS SETTING

JSGetBrightness

Purpose	Get the brightness value.
Syntax	**function JSGetBrightness();**
Parameters	none
Return Value	*brightness* float, 0 to 1.0, the brightness from dark to full bright

Example	`<script type="text/javascript">`
	`var ret=JSGetBrightness();`
	`</script>`

JSSetBrightness

Purpose	Switch the brightness between dark and full bright.
Syntax	**function JSSetBrightness(***brightness***);**
Parameters	*brightness* float, 0 to 1.0, the brightness from dark to full bright
Return Value	True if success, false if failure.

Example	`<script type="text/javascript">`
	`var ret=JSSetBrightness(1);`
	`</script>`

12.2.4 BROWSER TEXT SIZE

JSGetTextZoomLevel

Purpose	Get the browser text size.
Syntax	**function JSGetTextZoomLevel();**
Parameters	none
Return Value	*level*
	float, 0.1 to 2.5, the level from smallest to Largest

Example	`<script type="text/javascript">`
	`var ret=JSGetTextZoomLevel();`
	`</script>`

JSSetTextZoomLevel

Purpose	Adjust the browser text size.
Syntax	**function JSSetTextZoomLevel(***level***);**
Parameters	*level*
	float, 0.1 to 2.5, the level from smallest to Largest
Return Value	True if success, false if failure.

Example	`<script type="text/javascript">`
	`var ret=JSSetTextZoomLevel(1);`
	`</script>`

12.2.5 SOUND/VIBRATOR SETTING

JSGetRingerMode

Purpose	Get ring mode.
Syntax	**function JSGetRingerMode();**
Parameters	none
Return Value	*mode*

integer	
0	Silent mode (vibration off, sound off)
1	Vibrate mode (vibration on, sound off)
2	Normal mode (vibration on, sound on)

Example

```
<script type="text/javascript">
var ret=JSGetRingerMode();
</script>
```

JSSetRingerMode

Purpose	Configure ring mode.
Syntax	**function JSSetRingerMode(***mode***);**
Parameters	*mode*

integer	
0	Silent mode (vibration off, sound off)
1	Vibrate mode (vibration on, sound off)
2	Normal mode (vibration on, sound on)

Return Value	True if success, false if failure.

Example	`<script type="text/javascript">`
	`var ret=JSSetRingerMode(1);`
	`</script>`

12.2.6 VOLUME SETTING

JSGetDeviceVolume

Purpose	Get the volume setting for the specified stream.
Syntax	**function JSGetDeviceVolume(***stream***);**
Parameters	*stream*

integer (0-5)	
0	Voice Call
1	System
2	Ring
3	Music
4	Alarm
5	Notification

Return Value	*level*

integer - between 0 to 8 of Volume level; 0 is lowest level; 8 means full volume

Example	`<script type="text/javascript">`
	`var ret=JSGetDeviceVolume(1);`
	`</script>`

JSSetDeviceVolume

Purpose	Adjust the device volume for five audio streams as follows.
Syntax	**function JSSetDeviceVolume(***stream, level***);**

Parameters *stream*

integer (0-5)	
0	Voice Call
1	System
2	Ring
3	Music
4	Alarm
5	Notification

level

integer - between 0 to 7 of Volume level; 0 is lowest level; 7 means full volume.

Return Value True if success, false if failure.

Remarks Ring, System and Notification are coupled.

Example
```
<script type="text/javascript">
var ret=JSSetDeviceVolume(1, 6);
</script>
```

12.2.7 LOG MESSAGE

JSLog

Purpose Configure log message.

Syntax **function JSLog (***level, message***);**

Parameters *level*

Integer (1-5)	
1	All
2	Debug
3	Info
4	Warn
5	Error

string *message* - string of the message

Return Value True if success, false if failure.

Example Please refer to section 5.2.8.

12.2.8 READ LOG MESSAGES

JSReadLogFile

Purpose Read log message.

Syntax	**function JSReadLogFile();**
Parameters	none
Return Value	String that contains the entire message log

Example	`<script type="text/javascript">`
	`var promise=JSReadLogFile();`
	`promise.then(function(logEntries){`
	`  var text="";`
	
	`  for(var i=0; i < logEntries.length; i++){`
	`  var entry=logEntries[i];`
	`  text += getPropertyValues(entry)+"\n";`
	`  }`
	
	`  alert("Log:"+text);`
	`}).catch(function(error){`
	`  alert("[Error]"+error);`
	`});`
	`</script>`

12.2.9 CLEAN LOG MESSAGES

JSCleanLogFile	
Purpose	Empty the entire log file.
Syntax	**function JSCleanLogFile();**
Parameters	none
Return Value	True if success, false if failure.

Example	`<script type="text/javascript">`
	`var ret=JSCleanLogFile();`
	`</script>`

12.2.10 PLAY BEEP SOUND

JSBeep	
Purpose	Play the beep sound with specified settings.
Syntax	**function JSBeep(***onTime, offTime, freq, count***);**
Parameters	*onTime*: beep duration in 100ms *offTime*: beep stop duration in 100ms *freq*: frequency of sound *count*: repeat count
Return Value	True if success, false if failure.

Example

```
<script type="text/javascript">
var ret=JSBeep(10, 30, 700, 3);
</script>
```

12.2.11 VIBRATE

JSVibrate

Purpose: Vibrate the device with specified settings.

Syntax: **function JSVibrate(***onTime, offTime, count***);**

Parameters:
onTime: vibration duration in 100ms
offTime: vibration stop duration in 100ms
count: repeat count

Return Value: True if success, false if failure.

Example

```
<script type="text/javascript">
var ret=JSVibrate(20, 10, 3);
</script>
```

12.2.12 ENABLE/DISABLE DISPLAY SLEEP

JSGetDisplaySleep

Purpose: Get display sleep information.

Syntax: **function JSGetDisplaySleep();**

Parameters: none

Return Value: True if display sleep is enabled, false if disabled.

Example

```
<script type="text/javascript">
var ret=JSGetDisplaySleep();
</script>
```

JSSetDisplaySleep

Purpose: Enable/disable display sleep.

Syntax: **function JSSetDisplaySleep(***enable***);**

Parameters: *enable*

true	Enable display sleep
false	Disable display sleep

Return Value: True if success, false if failure.

Example

```
<script type="text/javascript">
var ret=JSSetDisplaySleep(true);
</script>
```

12.2.13 SET SCREEN AUTO ROTATION

JSGetAutoRotate

Purpose	Get auto rotation mode.
Syntax	**function JSGetAutoRotate ();**
Parameters	none
Return Value	*mode*

1	Enable auto rotation
2	Disable auto rotation and set to portrait mode
3	Disable auto rotation and set to landscape mode

Example

```
<script type="text/javascript">
var ret=JSGetAutoRotate();
</script>
```

JSSetAutoRotate

Purpose	Configure auto rotation mode.
Syntax	**function JSSetAutoRotate(***mode***);**
Parameters	*mode*

1	Enable auto rotation
2	Disable auto rotation and set to portrait mode
3	Disable auto rotation and set to landscape mode

Return Value	True if success, false if failure.
Example	

```
<script type="text/javascript">
var ret=JSSetAutoRotate(2);
</script>
```

12.2.14 GET BATTERY LEVEL

JSGetBatteryLevel

Purpose	Get battery level information.
Syntax	**function JSGetBatteryLevel();**
Parameters	none
Return Value	Battery level in percentage.
Example	

```
<script type="text/javascript">
var ret=JSGetBatteryLevel();
alert("JSGetBatteryLevel:"+ret);
</script>
```

12.2.15 GET DISPLAY LANGUAGE

SGetDisplayLanguage	
Purpose	Get the display language in device.
Syntax	**function JSGetDisplayLanguage();**
Parameters	none
Return Value	String indicating the device display language.

Example	`<script type="text/javascript">`
	`var ret=JSGetDisplayLanguage();`
	`alert("JSGetDisplayLanguage:"+ret);`
	`</script>`

12.2.16 GET CURRENT WIFI INFO

JSGetCurrentSsid	
Purpose	Get the Wi-Fi info string.
Syntax	**function JSGetCurrentSsid();**
Parameters	none
Return Value	String contains the following wifi info: - SSID
Remarks	We call Android's android.net.wifi.WifiInfo.toString() to get the wifi info string.

Example	`<script type="text/javascript">`
	`var ret=JSGetCurrentSsid();`
	`alert("JSGetCurrentSsid:"+ret);`
	`</script>`

12.2.17 GET SSID LIST

JSGetSsidList	
Purpose	Get the SSID list.
Syntax	**function JSGetSsidList();**
Parameters	none
Return Value	String contains a list of SSID and the signal strength in DB.

Example

```
<script type="text/javascript">
var promise_ssid=JSGetSsidList();
promise_ssid.then(function(result){
   alert("JSGetSsidList.result="+result);
   var text='';

   // Get the SSID's for each network.
   for(var i=0; i < result.length; i++){
      var network=result[i];
      text += network.ssid+"\n";
   }

    alert("SSID:"+text);
}).catch(function(error){
   alert("JSGetCurrentSsid[Error]:"+error);
});
</script>
```

12.2.18 WI-FI POWER ON/OFF

JSGetWifiPower

Purpose	Get the current wifi power status.
Syntax	**function JSGetWifiPower();**
Parameters	None
Return Value	True if WiFi is enabled; false if Wifi is disabled

Example	`<script type="text/javascript">`
	`function GetWifi(){`
	`  var ret=JSGetWifiPower();`
	`  alert("JSGetWifiPower:"+ret);`
	`};`
	`</script>`

JSSetWifiPower

Purpose	Set the wifi power to on or off.
Syntax	**function JSSetWifiPower(***mode***);**
Parameters	*mode* **0** Turn Wi-Fi off **1** Turn Wi-Fi on
Return Value	True if success, false if error

Example

```
<script type="text/javascript">
function SetWifi(){
    var ret=JSSetWifiPower(0);
    alert("JSSetWifiPower off:"+ret);

    ret=JSSetWifiPower(1);
    alert("JSSetWifiPower on:"+ret);
};
</script>
```

12.3 READER API

12.3.1 RELEASE BAR CODE READER

BarCodeRelease

Purpose	Release the barcode reader.
Syntax	**function BarCodeRelease();**
Parameters	none
Return Value	none

Example	`<script type="text/javascript">`
	`BarCodeRelease();`
	`</script>`

12.3.2 GET READER STATE

BarCodeGetActive

Purpose	Get the reader active state.
Syntax	**function BarCodeGetActive();**
Parameters	none
Return Value	True if enabled; false if disabled

Example	`<script type="text/javascript">`
	`var ret=BarCodeGetActive();`
	`alert("bar code canner is active="+ ret);`
	`</script>`

12.3.3 SET READER STATE

BarCodeSetActive

Purpose	Set the reader active state.
Syntax	**function BarCodeSetActive(***bActive***);**

Parameters *bActive*

true	Enable
false	Disable

Return Value True if success; false if failure

Example
```
<script type="text/javascript">
var ret=BarCodeSetActive(true);
if(ret){
   alert("bar code canner set active");
}
else{
   alert("bar code canner set active failed");
}
</script>
```

12.3.4 GET ERROR MESSAGE

BarCodeGetErrorMsg

Purpose Get the last error message when the following JS API returned false :
BarCodeSetDecodersStatus(),
BarCodeSetSymbology(),
BarCodeSetUserPreferences()

Syntax **function BarCodeGetErrorMsg();**

Parameters none

Return Value String - contains the last error message

Example
```
<script type="text/javascript">
alert("error="+BarCodeGetErrorMsg());

var ret=BarCodeSetUserPreferences(userPreference);
alert("BarCodeSetUserPreferences="+ret);

if(ret==false){
   alert("error:"+BarCodeGetErrorMsg() );
}
</script>
```

12.3.5 GET READER TYPE

BarCodeGetReaderType

Purpose Get the available reader types.

Syntax **function BarCodeGetReaderType();**

Parameters none

Return Value	A string that contains reader types, separated by space if more than 1 is available. If the bar code reader is not initialized, the ruturn value will be “NONE”.
	Moto_1D_SE955
	Moto_2D_4500
	Moto_1D_SE965
	Moto_2D_PL4507
	Intermec_2D_EX25
	Moto_1D_SE1524
	CL_1D_SM1
	SE4750SR_2D
	SE4750MR_2D
	Moto_1D_SE965I
	Moto_1D_SE965E

Example

```
<script type="text/javascript">
var ret=BarCodeGetReaderType();
alert("Bar Code type:"+ret);
</script>
```

12.3.6 GET READER OUTPUT CONFIGURATION

BarCodeGetReaderOutputConfiguration	
Purpose	Get reader output configuration.
Syntax	**function BarCodeGetReaderOutputConfiguration();**
Parameters	none
Return Value	A configuration JavaScript object with the definition as in the Set Reader Output Configuration section. If failed, null will be returned.

Example

```
<script type="text/javascript">
var config=BarCodeGetReaderOutputConfiguration();
alert("BarCodeGetReaderOutputConfiguration:"+config);
alert("charsetName:"+config["showCodeType"]);
</script>
```

12.3.7 SET READER OUTPUT CONFIGURATION

BarCodeSetReaderOutputConfiguration	
Purpose	Set reader output configuration.
Syntax	**function BarCodeSetReaderOutputConfiguration(***configuration***);**

Parameters

The configuration is a JavaScript object with the following definition:

Configuration object

config["showCodeType"]
config["showCodeLen"]
config["szPrefixCode"]
config["szSuffixCode"]
config["useDelim"]
config["szCharsetName"]
config["clearPreviousData"]

showCodeType

true
false

showCodeLen

true
false

szPrefixCode

prefix code string

szSuffixCode

suffix code string

useDelim

0
1-127

szCharsetName

0	UTF-8
1	Windows-1250
2	Windows-1251
3	Windows-1252
4	Windows-1253
5	Windows-1254
6	Windows-1255
7	Windows-1256
8	Windows-1257
9	Windows-1258
10	Big5
11	Shift_JIS
12	GBK

clearPreviousData

true
false

Return Value: True if success, false if error.

Example:

```
<script type="text/javascript">
var config=BarCodeGetReaderOutputConfiguration();
config["showCodeType"]=true;
config["showCodeLen"]=true;
config["szPrefixCode"]= "pre";
config["szSuffixCode"]= "suf";
config["useDelim"]= ",";
config["szCharsetName"]=1;
config["clearPreviousData"]= true;

var ret=BarCodeSetReaderOutputConfiguration(config);
alert("BarCodeSetReaderOutputConfiguration:"+ret);
</script>
```

12.3.8 GET READER SERVICE VERSION

BarCodeGetReaderServiceVersion

Purpose: Get reader service version.

Syntax: **function BarCodeGetReaderServiceVersion();**

Parameters: none

Return Value: String contains the reader service version. If failed, null will be returned.

Example

```
<script type="text/javascript">
var version=BarCodeGetReaderServiceVersion();
alert("Version:"+version);
</script>
```

12.3.9 GET NOTIFICATION SETTINGS

BarCodeGetNotificationParams

Purpose: Get notification settings.

Syntax: **function BarCodeGetNotificationParams();**

Parameters: none

Return Value: A JavaScript object defined as :

Notification object

```
notification["ledDuration"] =0;
notification["barcodeSuccessVibrateMS"] =50;
notification["barcodeVibrate"] =true;
notification["ReaderBeep"] =2;
```

ReaderBeep

0	Mute
1	Default
2	Hwandsw
3	MenuPop
4	MsgBox
5	Notify
6	VoiceBeep
7	Alarm2
8	Alarm3
9	LowBatt

enableVibrator

true	Enable vibrator
false	Disable vibrator

barcodeSuccessVibrateMS

Number	
1-1000	in increments of 0.1 seconds

ledDuration

Number	
0	Disable
1-5000	lighting duration in ms

If failed, null will be returned.

Example

```
<script type="text/javascript">
var notification=BarCodeGetNotificationParams();
alert("ledDuration:"+notification["ledDuration"]);
</script>
```

12.3.10 SET NOTIFICATION SETTINGS

BarCodeSetNotificationParams

Purpose: Set notification parameters.

Syntax: **function BarCodeSetNotificationParams(***notificationParams***);**

Parameters: *notificationParams*

a JavaScript object as defined in "Get Notification Settings"

Return Value: True if success, false if failure

Example

```
<script type="text/javascript">
var config=BarCodeGetNotificationParams();
notification["ledDuration"]=200;
notification["barcodeSuccessVibrateMS"]=3;
notification["barcodeVibrate"]=true;
notification["ReaderBeep"]=2;

var ret=BarCodeSetNotificationParams(notification);
alert("BarCodeSetNotificationParams:"+ret);
</script>
```

12.3.11 GET DECODERS STATUS

BarCodeGetDecodersStatus

Purpose: Get decoder status.

Syntax: **function BarCodeGetDecodersStatus();**

Parameters: none

Return Value

A JavaScript object defined as :

Decoders object

```
decoders["enableCodabar"]=true;
decoders["enableCode11"]=true;
decoders["enableCode39"]=true;
decoders["enableTriopticCode39"]=true;
decoders["enableCode93"]=true;
decoders["enableCode128"]=true;
decoders["enableGs1128"]=true;
decoders["enableIsbt128"]=true;
decoders["enableChinese2Of5"]=true;
decoders["enableIndustrial2Of5"]=true;
decoders["enableInterleaved2Of5"]=true;
decoders["enableMatrix2Of5"]=true;
decoders["enableKorean3Of5"]=true;
decoders["enableUccCoupon"]=true;
decoders["enableGs1DataBar14"]=true;
decoders["enableGs1DataBarLimited"]=true;
decoders["enableGs1DataBarExpanded"]=true;
decoders["enableMsi"]=true;
decoders["enableEanJan8"]=true;
decoders["enableEanJan13"]=true;
decoders["enableUpcA"]=true;
decoders["enableUpcE"]=true;
decoders["enableUpcE1"]=true;
decoders["enableComposite"]=true;
decoders["enableUSPostal"]=true;
decoders["enableUKPostal"]=true;
decoders["enableJapanPostal"]=true;
decoders["enableAustralianPostal"] =true;
decoders["enableDutchPostal"]=true;
decoders["enableUSPSPostal"]=true;
decoders["enableUPUFICSPostal"]=true;
decoders["enablePlessey"]=true;
decoders["enableTelepen"]=true;
decoders["enableAztec"]=true;
decoders["enableDataMatrix"]=true;
decoders["enableMaxiCode"]=true;
decoders["enablePDF417"]=true;
decoders["enableMicroPDF417"]=true;
decoders["enableQRcode"]=true;
decoders["enableMicroQR"]=true;
```

Each setting specifies whether the reader is able to decode the corresponding symbology.

false	off
true	On

If failed, null will be returned.

Example

```
<script type="text/javascript">
var ret_promise=BarCodeGetDecodersStatus();
ret_promise.then(function(decoders){
   var value=decoders["enableAustralianPostal"];
   alert("enableAustralianPostal:"+value);
}).catch(function(error){
   alert("[Error]"+error);
});
</script>
```

12.3.12 SET DECODERS STATUS

BarCodeSetDecodersStatus

Purpose: Set decoder status.

Syntax: **function BarCodeSetDecodersStatus(***decodersStatus***);**

Parameters: *decodersStatus*

a Javascript object as defined in “Get Decoders Status”

Return Value: True if success, false if failure

Example

```
<script type="text/javascript">
var ret_promise=BarCodeGetDecodersStatus();
ret_promise.then(function(decoders){
}).then(function(decoders){
   decoders["enableCodabar"]=true;
   decoders["enableCode11"]=true;
   decoders["enableCode39"]=true;
   decoders["enableTriopticCode39"]=true;
   decoders["enableCode93"]=true;
   decoders["enableCode128"]=true;
   decoders["enableGs1128"]=true;
   decoders["enableIsbt128"]=true;
   decoders["enableChinese2Of5"]=true;
   decoders["enableIndustrial2Of5"]=true;
   decoders["enableInterleaved2Of5"]=true;
   decoders["enableMatrix2Of5"]=true;
   decoders["enableKorean3Of5"]=true;
   decoders["enableUccCoupon"]=true;
   decoders["enableGs1DataBar14"]=true;
   decoders["enableGs1DataBarLimited"]=true;
   decoders["enableGs1DataBarExpanded"]=true;
   decoders["enableMsi"]=true;
   decoders["enableEanJan8"]=true;
   decoders["enableEanJan13"]=true;
   decoders["enableUpcA"]=true;
   decoders["enableUpcE"]=true;
   decoders["enableUpcE1"]=true;
   decoders["enableComposite"]=true;
   decoders["enableUSPostal"]=true;
   decoders["enableUKPostal"]=true;
   decoders["enableJapanPostal"]=true;
   decoders["enableAustralianPostal"]=true;
   decoders["enableDutchPostal"]=true;
   decoders["enableUSPSPostal"]=true;
   decoders["enableUPUFICSPostal"]=true;
   decoders["enablePlessey"]=true;
   decoders["enableTelepen"]=true;
   decoders["enableAztec"]=true;
   decoders["enableDataMatrix"]=true;
   decoders["enableMaxiCode"]=true;
   decoders["enablePDF417"]=true;
   decoders["enableMicroPDF417"]=true;
   decoders["enableQRcode"]=true;
   decoders["enableMicroQR"]=true;

   var ret=BarCodeSetDecodersStatus(decoders);
   alert("BarCodeSetDecodersStatus:"+ret);
}).catch(function (error){
   alert("[Error]"+error);
});
</script>
```

12.3.13 GET USERPREFERENCES

BarCodeGetUserPreferences	
Purpose	Get user preference.
Syntax	**function BarCodeGetUserPreferences();**
Parameters	none

Return Value	A JavaScript object defined as :
	
	userPreference["addonSecurityLevel"]=10;
	userPreference["displayMode"]=false;
	userPreference["laserOnTime"]=3000;
	userPreference["negativeBarcodes"]=0;
	userPreference["pickListMode"]= false;
	userPreference["redundancyLevel"]=1;
	userPreference["scanAngle"]=0;
	userPreference["securityLevel"]=0;
	userPreference["timeoutBetweenSameSymbology"]=1000;
	userPreference["transmitCodeIdChar"]=0;
	userPreference["triggerMode"]=0;
	userPreference["decodingIllumination"]= true;
	userPreference["decodingAimingPattern"]=true;
	userPreference["interCharGapSize"]=6;
	userPreference["timeoutPresentationMode"]=900000;
	userPreference["decodingIlluminationPowerLevel"]=5;
	userPreference["aimerMode"]=0;
	userPreference["centerDecoding"]=0;
	userPreference["centerDecodingTolerance"]=0;
	userPreference["triggerPresentationMode"]=false;

addonSecurityLevel

number, A value that specifies decode security level for reading UPC/EAN when “Decode with Addons (=Auto-discriminate)” is applied. The value can be from 2 to 30 to specify times of supplementary decoding.

displayMode

boolean (false=disable, true=enable), specifies whether to enable display mode.

laserOnTime

number, specifies the maximum time in ms for decoding a printed barcode during a scan process.

negativeBarcodes

number, specifies the negative barcode status.

0	Regular barcode
1	Negative barcode
2	Autodetection

pickListMode

boolean (false=disable, true=enable), specifies whether to enable picklist mode for decoding accuracy.

redundancyLevel

number, specifies decode redundancy. Higher redundancy levels should be selected for deteriorated barcode quality. This can be from 1 to 4, depicted below:

1	*The following barcodes must be successfully read twice before being decoded:

Barcode Types	Code Length
Codabar	All
MSI	4 characters or less
Industrial 25 (Discrete 25)	8 characters or less
Interleaved 25	8 characters or less

2	All barcodes must be successfully read twice before being decoded.
3	All barcodes must be successfully read twice before being decoded except for the following that must be read three times:

Barcode Types “Excluded”	Code Length
MSI	4 characters or less
Industrial 25 (Discrete 25)	8 characters or less
Interleaved 25	8 characters or less

4	All barcodes must be successfully read three times before being decoded.

scanAngle

number, specifies the scan angle.	
0	Narrow Angle (35 degrees).
1	Wide Angle (47 degrees)

securityLevel

number, specifies decode security level, which is useful to fix some printed quality issues when reading delda barcodes such as Code 128, Code 93, UPC/EAN. Value is from 0 to 3, meaning as below	
0	The default. It allows the scan engine to operate aggressively enough to decode most “in-spec” barcodes.
1	Select this option if misdecodes occur. This level should fix most misdecodes.
2	Select this option if Security Level 1 fails to fix misdecodes.
3	Select this option if Security Level 2 also fails to fix misdecodes. However, selecting this option impairs the decoding ability of the scan engine. If this level of security is necessary, try to improve the barcode quality.

timeoutBetweenSameSymbology

number, specifies the minimum time for the scan engine to resurrect the ability to read the same barcode for the 2nd time. This helps escape accidently scanning the same barcode twice. Value can be from 0 to 9900 (ms)

transmitCodeIdChar

number, specifies whether to transmit Code ID characters.	
0	Transmits none
1	Transmits AIM Code ID Character

triggerMode

number, specifies the mode to scan.	
0	Level mode
1	Continuous mode
2	Auto Aim mode
3	Presentation mode

decodingIllumination

boolean (false=disable, true=enable), specifies whether to provide flash illumination during every barcode capture to aid decoding.

decodingAimingPattern

boolean (false=disable, true=enable), specifies whether to project the aiming pattern on every barcode capture.

interCharGapSize

number, specifies the intercharacter gap size for Code 39 and Codabar, which is typically quite small. Due to various barcode printing technologies, this gap can grow larger than the maximum size allowed and prevent the scan engine from decoding a barcode. If this problem occurs, set it to “Large Intercharacter Gaps” to tolerate these out-of-specification barcodes.	
6	Normal intercharacter gaps
10	Large intercharacter gaps

decodingIlluminationPowerLevel

number, specifies the decodingIlluminationPowerLevel, 0 to 10.

timeoutPresentationMode

number, specifies the time in ms to enable the Presentation mode. This parameter applies to Continuous mode. 60000~ 1800000 (default is 900000, i.e. 15mins, ranging from 1 to 30 minutes)

aimerMode

number, the aimer mode is used to change how the aimer behaves:	
0	Typical Aimer
1	One pull to aim and read
2	One pull to aim, second pull to read

centerDecoding

boolean (false=disable, true=enable), enables or disables center decoding

centerDecodingTolerance

number, sets the center decoding tolerance ranging from 0 to 100.

triggerPresentationMode

boolean (false=disable, true=enable), enables or disables presentation mode

If failed, null will be returned.

Example

```
<script type= "text/javascript">
var userPreference=BarCodeGetUserPreferences();
alert("securityLevel:"+userPreference["securityLevel"]);
</script>
```

12.3.14 SET USERPREFERENCES

BarCodeSetUserPreferences	
Purpose	Set user preference.
Syntax	**function BarCodeSetUserPreferences(***userPreference***);**
Parameters	A userPreference JavaScript object as defined in “Get User Preference”. For all class members of Number type.

Return Value: True if success, false if failure

Example:

```
<script type="text/javascript">
var userPreference=BarCodeGetUserPreferences();

userPreference["addonSecurityLevel"]=10;
userPreference["displayMode"]=false;
userPreference["laserOnTime"]=3000;
userPreference["negativeBarcodes"]=0;
userPreference["pickListMode"]= false;
userPreference["redundancyLevel"]=1;
userPreference["scanAngle"]=0;
userPreference["securityLevel"]=0;
userPreference["timeoutBetweenSameSymbology"]=1000;
userPreference["transmitCodeIdChar"]=0;
userPreference["triggerMode"]=0;
userPreference["decodingIllumination"]= true;
userPreference["decodingAimingPattern"]=true;
userPreference["interCharGapSize"]=6;
userPreference["timeoutPresentationMode"]=900000;
userPreference["decodingIlluminationPowerLevel"]=5;
userPreference["aimerMode"]=0;
userPreference["centerDecoding"]=0;
userPreference["centerDecodingTolerance"]=0;
userPreference["triggerPresentationMode"]=false;

var ret=BarCodeSetUserPreferences(userPreference);
alert("BarCodeSetUserPreferences="+ret);
</script>
```

12.3.15 GET SYMBOLOGY

BarCodeGetSymbology

Purpose: This function provides parameters of symbologies implemented as objects.

Syntax: **function BarCodeGetSymbology(***symbologyName***);**

Parameters: *symbologyName*

string, specify the symbology setting to obtain, can be one of these:
Codabar, Code11, Code39, TriopticCode39, Korean3Of5, Code93, Code128, GS1128, ISBT128, Chinese2Of5, Industrial2Of5, Interleaved2Of5, Matrix2Of5, UccCoupon, GS1DataBar14, GS1DataBarLimited, GS1DataBarExpanded, Msi, EanJan8, EanJan13, UpcA, UpcE, UpcE1, Composite, USPostal, UKPostal, JapanPostal, AustralianPostal, DutchPostal, USPSPostal, UPUFICSPostal, PDF417, MicroPDF417, DataMatrix, MaxiCode, QRCode, MicroQR, Aztec, Telepen, Plessey

For all the class members of Number type supported.

Return Value

A JavaScript object for the “symbologyName” of a symbology. Each symbology has its own object definition.

Codabar object

codabar["name"]
codabar["enabled"]
codabar["transmitCheckDigit"]
codabar["verifyCheckDigit"]
codabar["notisEditingType"]
codabar["length1"]
codabar["length2"]
codabar["clsiEditing"]
codabar["notisEditing"]

name

“Codabar” - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Codabar.

transmitCheckDigit

boolean (false=disable, true=enable), specifying whether to transmit check digit.

verifyCheckDigit

number, specifying whether and how to verify check digit.	
0	None
1	Modulo_16
2	Modulo_7DR
3	Modulo_Both

notisEditingType

number, specifying whether to transform it to NOTIS editing format (Start/Stop characters) and the way it is transformed.	
0	None
1	ABCD
2	abcd

length1, length2

number, Length qualification (0-55)

clsiEditing

boolean (false=disable, true=enable), specifying whether to edit CLSI.

notisEditing

boolean (false=disable, true=enable), reserved.

Return Value

Code11 object

code11["name"]
code11["enabled"]
code11["length1"]
code11["length2"]
code11["numberOfCheckDigits"]
code11["transmitCheckDigit"]

name

"Code11" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Code 11.

length1, length2

number, Length qualification (0-55)

numberOfCheckDigits

number, specifying whether and how to verify check digit. (0-2), where 0=None

transmitCheckDigit

boolean (false=disable, true=enable), specifying whether to transmit check digit.

Return Value

Code39 object

code39["name"]
code39["enabled"]
code39["length1"]
code39["length2"]
code39["checkDigitVerification"]
code39["transmitCheckDigit"]
code39["fullASCII"]
code39["convertToCode32"]
code39["convertToCode32Prefix"]

name

"Code39" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Code39.

length1, length2

number, Length qualification (0-55)

checkDigitVerification

boolean (false=disable, true=enable), specifies whether to verify check digit.

transmitCheckDigit

boolean (false=disable, true=enable), specifying whether to transmit check digit.

fullASCII

boolean (false=disable, true=enable), specifies whether to support Code 39 Full ASCII.

convertToCode32

boolean (false=disable, true=enable), specifies whether to convert Code 39 to Code 32 (= Italian Pharmacode).

convertToCode32Prefix

boolean (false=disable, true=enable), specifies whether to transmit prefix for Code 32 data.

Return Value

TriopticCode39 object

tiopticCode39["name"]

tiopticCode39["enabled"]

name

"TriopticCode39" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable tiopticCode39.

Code93 object

cde93["name"]

cde93["enabled"]

cde93["length1"]

cde93["length2"]

name

"Code93" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Code93.

length1, length2

number, Length qualification (0-55)

Code128 object

cde128["name"]

cde128["enabled"]

cde128["securityLevel"]

cde128 ["length1"]

cde128 ["length2"]

name

"Code128" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable Code128.

securityLevel

number (0=normal, 1=high), specifies the decode security level while reading Code128.

length1, length2

number, Length qualification (0-55)

Return Value

GS1128 object

gs1128["name"]

gs1128["enabled"]

gs1128["fieldSeparator"]

gs1128["enableApplicationIdentifier"]

gs1128["applicationIdentifierMark1"]

gs1128["applicationIdentifierMark2"]

name

"GS1128" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable GS1128.

fieldSeparator

number, specifies whether to apply a field separator of ASCII whose value is zero or larger. It's set to zero by default.

enableApplicationIdentifier

boolean (false=disable, true=enable), specifies whether to enable the application identifier.

applicationIdentifierMark1

string, specifies Application ID mark1.

applicationIdentifierMark2

string, specifies Application ID mark2.

ISBT128 object

isbt128["name"]

isbt128["enabled"]

isbt128["concatenation"]

isbt128["concatenationRedundancy"]

name

"ISBT128" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable ISBT128.

concatenation

number, specifies whether to decode and concatenate pairs of ISBT barcodes. 0=disable, 1=enable, 2=auto

concatenationRedundancy

number, specifies concatenation redundancy (2~20 times) when auto-discriminate of ISBT concatenation is enabled. By default, it is set to 10 times.

Return Value

Chinese2Of5 object

chinese2Of5["name"]

chinese2Of5["enabled"]

name

"chinese2Of5" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable chinese2Of5.

Industrial2Of5 object

industrial2Of5["name"]

industrial2Of5["enabled"]

industrial2Of5["length1"]

industrial2Of5["length2"]

name

"Industrial2Of5" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Industrial2Of5.

length1, length2

number, Length qualification (0-55)

Return Value **Interleaved25 object**

iterleaved25["name"]
iterleaved25["enabled"]
iterleaved25["length1"]
iterleaved25["length2"]
iterleaved25["checkDigitVerification"]
iterleaved25["transmitCheckDigit"]
iterleaved25["convertToEan13"]
iterleaved25["securityLevel"]

name

"Interleaved2Of5" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Interleaved25.

length1, length2

number, Length qualification (0-55)

checkDigitVerification

number, specifies whether and how to verify check digit.	
0	The default. disable
1	USS
2	OPCC
3	Modulo_10
4	French_CIP_HR

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

convertToEan13

boolean (false=disable, true=enable), specifies whether to convert Interleaved 25 to EAN-13.

securityLevel

number, specifies the decode security level while reading GS1 DataBar 14. Value is from 0 to 3, meaning as below	
0	Zero.
1	The default. One.
2	Two.
3	Three.

Return Value

Matrix25 object

matrix25["name"]
matrix25["enabled"]
matrix25["length1"]
matrix25["length2"]
matrix25["redundancy"]
matrix25["checkDigitVerification"]
matrix25["transmitCheckDigit"]

name

"Matrix2Of5" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Matrix2Of5.

length1, length2

number, Length qualification (0-55)

redundancy

boolean (false=disable, true=enable), specifies whether to enable decode redundancy.

checkDigitVerification

boolean (false=disable, true=enable), whether to verify check digit.

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

Korean3Of5 object

koean3Of5["name"]
koean3Of5["enabled"]

name

"Korean3Of5" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable korean3Of5.

Return Value

UccCoupon object

uccCoupon["name"]

uccCoupon["enabled"]

name

"UccCoupon" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable UccCoupon.

GS1DataBar14 object

gs1DataBar14["name"]

gs1DataBar14["enabled"]

gs1DataBar14["convertToUpcEan"]

gs1DataBar14["securityLevel"]

name

"GS1DataBar14" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable UccCoupon.

convertToUpcEan

boolean (false=disable, true=enable), specifies whether to convert RSS to UPC/EAN barcodes.

securityLevel

number (1-4), specifies the decode security level while reading gs1DataBar14.

GS1DataBarLimited object

gs1DataBarLimited["name"]

gs1DataBarLimited["enabled"]

gs1DataBarLimited["convertToUpcEan"]

gs1DataBarLimited["securityLevel"]

name

"GS1DataBarLimited" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable GS1DataBarLimited.

convertToUpcEan

boolean (false=disable, true=enable), specifies whether to convert RSS to UPC/EAN barcodes.

securityLevel

number (1-4), specifies the decode security level while reading GS1 DataBar Limited.

Return Value

GS1DataBarExpanded object

gs1DataBarExpanded["name"]

gs1DataBarExpanded["enabled"]

gs1DataBarExpanded["fieldSeparator"]

gs1DataBarExpanded["securityLevel"]

name

"GS1DataBarExpanded" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable GS1DataBarExpanded.

fieldSeparator

number, specifies whether to apply a field separator of ASCII whose value is zero or larger. It's set to zero by default.

securityLevel

number (0-3), specifies the decode security level while reading GS1DataBarExpanded.

Return Value

Msi object

msi["name"]
msi["enabled"]
msi["length1"]
msi["length2"]
msi["checkDigitOpiton"]
msi["transmitCheckDigit"]
msi["checkDigitAlgorithm"]

name

“Msi” - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Msi.

length1, length2

number, Length qualification (0-55)

checkDigitOption

number, specifies how to verify check digit (1=Onedigit, 2=TwoDigits).

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

checkDigitAlgorithm

number, specifies which algorithm to apply.	
1	Modulo_10_11
2	DoubleModulo_10

Return Value

EanJan8 object

ean8["name"]

ean8["enabled"]

ean8["addon2"]

ean8["addon5"]

ean8["transmitCheckDigit"]

ean8["convertToEan13"]

name

"EanJan8" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable EanJan8.

addon2

number, specifies the way processing addon2 (1=IgnoresAddon, 2=AutoDiscriminate).

addon5

number, specifies the way processing addon5 (1=IgnoresAddon, 2=AutoDiscriminate).

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

convertToEan13

boolean (false=disable, true=enable), specifies whether to convert EAN-8 to EAN-13.

Return Value

EanJan13 object

ean13["name"]
ean13["enabled"]
ean13["addon2"]
ean13["addon5"]
ean13["convertToISBN"]
ean13["convertToISSN"]
ean13["booklandISBNFormat"]
ean13["transmitCheckDigit"]

name

“EanJan13” - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable EanJan13.

addon2

number, specifies the way processing addon2 (1=IgnoresAddon, 2=AutoDiscriminate).

addon5

number, specifies the way processing addon5 (1=IgnoresAddon, 2=AutoDiscriminate).

convertToISBN

boolean (false=disable, true=enable), specifies whether to convert EAN-13 to ISBN.

convertToISSN

boolean (false=disable, true=enable), specifies whether to convert EAN-13 to ISSN.

booklandISBNFormat

number, If you enabled Bookland EAN, select one of the following formats for Bookland data.	
0	ISBN_10
1	ISBN_13

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

Return Value

UPCA object

upca["name"]

upca["enabled"]

upca["addon2"]

upca["addon5"]

upca["transmitCheckDigit"]

upca["transmitSystemNumber"]

upca["convertToEan13"]

name

“UpcA” - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable UpcA.

addon2

number, specifies the way processing addon2 (1=IgnoresAddon, 2=AutoDiscriminate).

addon5

number, specifies the way processing addon5 (1=IgnoresAddon, 2=AutoDiscriminate).

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

transmitSystemNumber

number, specifies whether to verify (transmit) UPC-A preamble.	
0	None
1	SysNumOnly
2	SysNumAndCtyCode

convertToEan13

boolean (false=disable, true=enable), specifies whether to convert EAN-8 to EAN-13.

Return Value

UPCE object

upce["name"]
upce["enabled"]
upce["addon2"]
upce["addon5"]
upce["transmitCheckDigit"]
upce["transmitSystemNumber"]
upce["convertToUpcA"]

name

“UpcE” - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable UpcE.

addon2

number, specifies the way processing addon2 (1=IgnoresAddon, 2=AutoDiscriminate).

addon5

number, specifies the way processing addon5 (1=IgnoresAddon, 2=AutoDiscriminate).

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

transmitSystemNumber

number, specifies whether to verify (transmit) UPC-A preamble.	
0	None
1	SysNumOnly
2	SysNumAndCtyCode

convertToUpcA

boolean (false=disable, true=enable), specifies whether to convert UPC-A.

Return Value

UPCE1 object

upce1["name"]

upce1["enabled"]

upce1["addon2"]

upce1["addon5"]

upce1["transmitCheckDigit"]

upce1["transmitSystemNumber"]

upce1["convertToUpcA"]

name

"UpcE1" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable UpcE1.

addon2

number, specifies the way processing addon2 (1=IgnoresAddon, 2=AutoDiscriminate).

addon5

number, specifies the way processing addon5 (1=IgnoresAddon, 2=AutoDiscriminate).

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

transmitSystemNumber

number, specifies whether to verify (transmit) UPC-A preamble.	
0	None
1	SysNumOnly
2	SysNumAndCtyCode

convertToUpcA

boolean (false=disable, true=enable), specifies whether to convert UPC-A.

Return Value

Composite object

composite["name"]
composite["enableCc_C"]
composite["enableCc_AB"]
composite["enableTlc39"]
composite["enableUpcMode"]
composite["enableEmulationMode"]

name

"Composite" - to specify the name of this class.

enableCc_C

boolean (false=disable, true=enable), specifying whether to enable Composite CC-C.

enableCc_AB

boolean (false=disable, true=enable), specifying whether to enable Composite CC-A/B.

enableTlc39

boolean (false=disable, true=enable), specifying whether to enable Composite TLC-39 (=TCIF Linked).

enableUpcMode

number, specifies whether to enable UPC Composite Mode.	
0	NeverLinksUPC
1	AlwaysLinksUPC
2	Auto

enableEmulationMode

boolean (false=disable, true=enable), specifies whether to enable GS1-128 Emulation Mode for UCC/EAN Composite Codes.

Return Value

USPostal object

usPostal["name"]

usPostal["enablePlanet"]

usPostal["enablePostnet"]

usPostal["transmitCheckDigi"]

name

"USPostal" - to specify the name of this class.

enablePlanet

boolean (false=disable, true=enable), specifying whether to enable US Planet.

enablePostnet

boolean (false=disable, true=enable), specifying whether to enable US Postnet.

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

UKPostal object

ukPostal["name"]

ukPostal["enabled"]

ukPostal["transmitCheckDigit"]

name

"UKPostal" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable UK Postal

transmitCheckDigit

boolean (false=disable, true=enable), specifies whether to transmit check digit.

JapanPostal object

japanPostal["name"]

japanPostal["enabled"]

name

"JapanPostal" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable JapanPostal

Return Value

AustralianPostal object

australianPostal["name"]

australianPostal["enabled"]

name

"AustralianPostal" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable Australian Postal

DutchPostal object

dutchPostal["name"]

dutchPostal["enabled"]

name

"DutchPostal" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable Dutch Postal

USPSPostal object

uspspostal["name"]

uspspostal["enabled"]

name

"USPSPostal" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable USPS Postal

UPUFICSPostal object

upuficspostal["name"]

upuficspostal["enabled"]

name

"UPUFICSPostal" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifies whether to enable UPUFICS Postal.

Return Value

Aztec object

aztec["name"]

aztec["enabled"]

name

"Aztec" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable Aztec.

DataMatrix object

datamatrix["name"]

datamatrix["enabled"]

datamatrix["fieldSeparator"]

datamatrix["mirrorImage"]

name

"DataMatrix" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable DataMatrix.

fieldSeparator

number, specifies whether to apply a field separator of ASCII whose value is zero or larger. It's set to zero by default.

mirrorImage

number, specifies whether to decode mirror image Data Matrix barcodes..	
0	Never
1	Always
2	Auto

MaxiCode object

maxicode["name"]

maxicode["enabled"]

name

"MaxiCode" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable MaxiCode.

Return Value

PDF417 object

pdf417["name"]
pdf417["enabled"]
pdf417["transmitMode"]
pdf417["escapeCharacter"]
pdf417["transmitControlHeader"]

name

"PDF417" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable PDF417.

transmitMode

(Reserved) number, specifies how to handle decoding.

escapeCharacter

(Reserved) boolean (false=disable, true=enable), specifies whether to use the escape character.

transmitControlHeader

(Reserved) boolean (false=disable, true=enable), specifies whether to transmit the control header.

MicroPDF417 object

microPDF417["name"]
microPDF417["enabled"]
microPDF417["code128Emulation"]

name

"MicroPDF417" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable MicroPDF417.

code128Emulation

boolean (false=disable, true=enable), specifies whether to enable Code 128 Emulation for certain MicroPDF417 barcods.

Return Value

QRCode object

qrcode["name"]

qrcode["enabled"]

name

"QRCode" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable QRCode.

MicroQR object

microQR["name"]

microQR["enabled"]

name

"MicroQR" - to specify the name of this class.

enabled

boolean (false=disable, true=enable), specifying whether to enable MicroQR.

Example

```
<script type= "text/javascript">
var codaBar=BarCodeGetSymbology("Codabar");
alert("CodaBar.transmitCheckDigit:"+codaBar["transmitCheckDigit
"]);
</script>
```

12.3.16 SET SYMBOLOGY

BarCodeSetSymbology

Purpose: This function provides parameters of symbologies implemented as objects.

Syntax: **function BarCodeSetSymbology(***symbologyObject***);**

Parameters: *symbologyObject*

JavaScript object, as symbologyObject defined in "Get Symbology".

For all the class members of Number type.

Return Value: True if success, false if failure.

Example

```
<script type="text/javascript">
var codabar=BarCodeGetSymbology("Codabar");
codabar["enabled"]=true;
codabar["transmitCheckDigit"]=false;
codabar["verifyCheckDigit"]=0;
codabar["notisEditingType"]=1;
codabar["length1"]=4;
codabar["length2"]=55;
codabar["clsiEditing"]=false;

var ret=BarCodeSetSymbology("codabar");
alert("Set CodaBar settings success:"+ret);
</script>
```

12.3.17 RESET READER

BarCodeReset	
Purpose	Reset reader module(s).
Syntax	**function BarCodeReset();**
Parameters	none
Return Value	True if success, false if failure. The barcode reader needs to take about 2 seconds to reset.

Example	`<script type="text/javascript">`
	`var ret=BarCodeReset();`
	`</script>`

12.4 SYSTEM API

12.4.1 GET SYSTEM INFORMATION

JSGetSystemInfo	
Purpose	Get the device system information.
Syntax	**function JSGetSystemInfo();**
Parameters	none

Return Value — JS object that contains the system information:

```
sysInfo["manufacturer"]
sysInfo["brand"]
sysInfo["model"]
sysInfo["board"]
sysInfo["hardware"]
sysInfo["serial"]
sysInfo["deviceId"]
sysInfo["apiLevel"]
sysInfo["sdk"]
sysInfo["buildId"]
sysInfo["buildTime"]
sysInfo["buildVersion"]
```

Example

```
<script type="text/javascript">
var sysInfo=JSGetSystemInfo();
alert("JSGetSystemInfo:"+sysInfo["manufacturer"]+" "
       +sysInfo["model"]+" "
       +sysInfo["sdk"]+" "
       +sysInfo["serial"]+" "
       +sysInfo["buildVersion"]+" "
      );
</script>
```

12.4.2 QUIT BROWSER

JSCloseBrowser

Purpose — Close the browser.

Syntax — **function JSCloseBrowser(***appPackageName***);**

Parameters — *appPackageName*

string - If application appPackageName is present on the system, after browser quit it will launch the application.

Return Value — True if success, false if failure.

Example

```
<script type= "text/javascript">
JSCloseBrowser("com.mediatek.filemanager");
</script>
```

12.4.3 LAUNCH OTHER APPLICATION

JSLaunchApp

Purpose — Launch the specified application.

Syntax — **function JSLaunchApp(***appPackageName***);**

Parameters	*appPackageName*
	string - String, the package name of the Android application. If application package appPackageName is present on the system, browser will launch the application.
Return Value	True if success, false if failure.

Example	`<script type="text/javascript">`
	`JSLaunchApp("com.mediatek.filemanager");`
	`</script>`

12.5 FILE API

12.5.1 COPY LOCAL FILE

JSCopyFile

Purpose	Copy an existing file to a new file.
Syntax	**function JSCopyFile(***existingFilename, newFileName***);**
Parameters	*existingFilename*
	string - the existing file to be copied.
	newFilename
	string - the new filename.
Return Value	True if success, false if failure.

Example	`<script type="text/javascript">`
	`var sCurrDir=JSFileGetCurrentDir();`
	`var file=sCurrDir+"/test.txt";`
	`var fileCopy=sCurrDir+"/testCopy.txt";`
	`var promise=JSCopyFile(file,fileCopy);`
	`promise.then(function(){`
	`  alert("Copy:"+file+" to "+fileCopy);`
	`}).catch(function(error){`
	`  alert("[Error]"+error);`
	`});`
	`</script>`

12.5.2 DELETE LOCAL FILE

JSDeleteFile

Purpose	Delete a file on the local file system.
Syntax	**function JSDeleteFile(***filename***);**
Parameters	*filename*
	string - the existing file to be deleted

Return Value	True if success, false if failure.

Example	<script type="text/javascript">
	var sCurrDir=JSFileGetCurrentDir();
	var testFileName=sCurrDir+"/test.txt";
	var promise_deled=JSDeleteFile(testFileName);
	promise_deled.then(function(){
	alert("Deleted:"+testFileName);
	}).catch(function(error){
	alert("[Error]"+error);
	});
	</script>

12.5.3 RENAME LOCAL FILE

JSRenameFile

Purpose	Rename an existing file.
Syntax	**function JSRenameFile(***oldFilename, newFileName***);**
Parameters	*oldFilename* string - the existing file to be renamed *newFilename* string - the new filename
Return Value	True if success, false if failure.

Example	<script type="text/javascript">
	var sCurrDir=JSFileGetCurrentDir();
	var file=sCurrDir+"/test.txt";
	var promise=JSRenameFile(file,file+".txt");
	promise.then(function(){
	alert("Rename:"+file+" to "+file+".txt");
	}).catch(function(error){
	alert("[Error]"+error);
	});
	</script>

12.5.4 CHECK LOCAL FILE EXISTS

JSFileExists

Purpose	Check whether a local file exists.
Syntax	**function JSFileExists(***filename***);**
Parameters	*filename* string - the file to be checked
Return Value	True if success, false if failure.

Example

```
<script type="text/javascript">
var sCurrDir=JSFileGetCurrentDir();
var testFileName=sCurrDir+"/test.txt";
var promise=JSFileExists(testFileName);
promise.then(function(bExists){
   if(bExists==true){
      alert("Get file:"+testFileName);
   }
    else{
       alert("Cannot find:"+testFileName);
    }
}).catch(function(error){
    alert("[Error]"+error);
});
</script>
```

12.5.5 GET LOCAL DIRECTORY PATH

JSFileGetCurrentDir	
Purpose	Get the current directory path.
Syntax	**function JSFileGetCurrentDir();**
Parameters	none
Return Value	The current path of the Application if success, null if failure.

Example	`<script type="text/javascript">`
	`var sCurrDir=JSFileGetCurrentDir();`
	`alert("JSFileGetCurrentDir:"+sCurrDir);`
	`</script>`

12.5.6 GET SD CARD DIRECTORY PATH

JSFileGetSDDir	
Purpose	Get the path of the SD card.
Syntax	**function JSFileGetSDDir();**
Parameters	none
Return Value	The path of the SD card if success, null if failure.

Example	`<script type="text/javascript">`
	`var sSDDir=JSFileGetSDDir();`
	`alert("JSFileGetSDDir:"+sSDDir);`
	`</script>`

12.5.7 OPEN LOCAL FILE

JSFileOpen	
Purpose	Open an existing file.
Syntax	**function JSFileOpen(***filename***);**
Parameters	*filename* string - the file to be opened
Return Value	An integer file ID if file exists, -1 if error
Example	Please refer to section 5.6.13 Write Text Data To Local File (Text).

12.5.8 CREATE LOCAL FILE

JSFileCreate	
Purpose	Create a new file.
Syntax	**function JSFileCreate(***filename***);**
Parameters	*filename* string - the file to be created
Return Value	An integer file ID if the file exists, -1 if error.
Example	

```
<script type="text/javascript">
var sCurrDir=JSFileGetCurrentDir();
var testFileName=sCurrDir+"/"+"test.txt";
var promise_created=JSFileCreate(testFileName);
promise_created.then(function(fileID){
   var promise_closed=JSFileClose(fileID);
   promise_closed.then(function(){
      alert("Created and closed:"+testFileName+",fileID:"+fileID);
    }).catch(function(error){
       alert("JSFileClose[Error]"+error);
     });
}).catch(function(error){
   alert("[Error]"+error);
});
</script>
```

12.5.9 CLOSE LOCAL FILE

JSFileClose	
Purpose	Close a file.
Syntax	**function JSFileClose(***fileID***);**

Parameters	*fileID*
	integer - the fileID is obtained when opening a file.
Return Value	True if success, false if failure.
Example	Please refer to section 5.6.13 Write Text Data To Local File (Text).

12.5.10 READ BINARY DATA FROM LOCAL FILE

JSFileReadBase64

Purpose	Read binary data from an existing file.
Syntax	**function JSFileReadBase64(***fileID, length***);**
Parameters	*fileID*
	integer - the fileID is obtained when opening a file.
	length
	integer - the number of byte to read.
Return Value	String of a base64 encoded data, null if failed.
Example	Please refer to section 5.6.12 Write Binary Data to Local File.

12.5.11 READ TEXT DATA FROM LOCAL FILE (TEXT)

JSFileRead

Purpose	Read text data from an existing file.
Syntax	**function JSFileRead(***fileID, length***);**
Parameters	*fileID*
	integer - the fileID is obtained when opening a file.
	length
	integer - the number of byte to read.
Return Value	String of the text data, null if failed.
Example	Please refer to section 5.6.13 Write Text Data To Local File (Text).

12.5.12 WRITE BINARY DATA TO LOCAL FILE

JSFileWriteBase64

Purpose	Write binary data to an existing file.
Syntax	**function JSFileWriteBase64(***fileID, base64string***);**

Parameters	*fileID*
	integer - the fileID is obtained when opening a file
	base64string
	string - data encoded in a base64 string
Return Value	True if success, false if failure.
Example	

```
<script type="text/javascript">
var sCurrDir=JSFileGetCurrentDir();
var testFileName=sCurrDir+"/test.txt";
var promise_created=JSFileCreate(testFileName);
promise_created.then(function (fileID){
    var str="1234567890";
    var encStr=window.btoa(str);
    var promise_wrote=JSFileWriteBase64(fileID, encStr);
    promise_wrote.then(function(){
        alert("Successfully write encoded base64 data:"+encStr);
        var promise_closed=JSFileClose(fileID);
        promise_closed.then(function()
        {
            alert("Closed:"+testFileName);
        }).catch(function(error){
           alert("JSFileClose[Error]"+error);
        });

    }).catch(function(error){
        alert("JSFileWriteBase64 [Error]"+error);
     });


}).catch(function(error){
    alert("JSFileCreate[Error]"+error);
});
</script>
```

12.5.13 WRITE TEXT DATA TO LOCAL FILE (TEXT)

JSFileWriteText	
Purpose	Write text data to an existing file.
Syntax	**function JSFileWriteText(***fileID, dataString***);**
Parameters	*fileID*
	integer - the fileID is obtained when opening a file
	dataString
	string - the data to be written

Return Value	True if success, false if failure.
Example	

```
<script type="text/javascript">
var sCurrDir=JSFileGetCurrentDir();
var testFileName=sCurrDir+"/test.txt";
var promise_created=JSFileCreate(testFileName);
promise_created.then(function (fileID){
   var promise_wrote= JSFileWriteText(fileID, "1234567890");
   promise_wrote.then(function(){
      alert("Successfully write encoded base64 data:"+encStr);
      var promise_closed=JSFileClose(fileID);
      promise_closed.then(function(){
         alert("Closed:"+testFileName);
      }).catch(function(error){
        alert("JSFileClose[Error]"+error);
      });

   }).catch(function(error){
      alert("JSFileWriteBase64[Error]"+error);
    });


}).catch(function(error){
   alert("JSFileCreate[Error]"+error);
});
</script>
```

12.5.14 GET CURRENT FILE OFFSET

JSGetFileOffset	
Purpose	Get the current file offset.
Syntax	**function JSGetFileOffset(***fileID***);**
Parameters	*fileID* integer - the file ID is obtained when opening a file.
Return Value	The current file offset. -1 if failed.
Example	Please refer to section 5.6.16 Get File Size.

12.5.15 SEEK FILE OFFSET

JSFileSeek	
Purpose	Set the file-pointer offset, measured from the beginning of the file with specified file ID, at which the next read or write occurs.
Syntax	**function JSFileSeek(***fileID, offset***);**

Parameters	*fileID*
	integer - the file ID of the target file, which can be obtained when opening the file
	offset
	integer - the offset of the file pointer, measured from the beginning of the file.
Return Value	True if success, false if failure.
Example	Please refer to section 5.6.16 Get File Size.

12.5.16 GET FILE SIZE

JSGetFileSize

Purpose	Get the file size in bytes.
Syntax	**function JSGetFileSize(***fileID***);**
Parameters	*fileID*
	integer - the file ID is obtained when opening the target file
Return Value	Integer: size of the file in bytes; -1 if failed.
Example	

```
<script type="text/javascript">
var sCurrDir=JSFileGetCurrentDir();
var file=sCurrDir+"/test.txt";
var promise_opened=JSFileOpen(file);
promise_opened.then(function (fileID){
    var length=JSGetFileSize(fileID);
    var fileOffset=JSGetFileOffset(fileID);
    alert("Current offset of "+file+":"+fileOffset);
    JSFileSeek(fileID,fileOffset+1);
    var promise_read=JSFileRead(fileID, length);
    promise_read.then(function(text){
        alert("Read file:"+text);
        var promise_closed=JSFileClose(fileID);
        promise_closed.then(function(){
           alert("Closed file:"+file);
        }).catch(function(error){
            alert("JSFileClose[Error]"+error);
        });

    }).catch(function(error){
       alert("JSFileRead[Error]"+error);
    });

}).catch(function(error){
    alert("JSFileOpen[Error]"+error);
});
</script>
```

12.6 HTTP HANDLING API

No customized JavaScript is required for handling HTTP request. Users can use the HTML5 XMLHttpRequest instead. Please refer to https://xhr.spec.whatwg.org/ for further details.

12.6.1 HTTP ERROR HANDLING

JSSetHttpErrorAction

Purpose	When HTTP error occurs and a page cannot be loaded, set the action the browser will take.
Syntax	**function JSSetHttpErrorAction(***action, url***);**
Parameters	*action*

an integer	
0	Displays the error page
1	Prevents navigation to the page and does not display an error

url

string - The url to go when http error occurs

Return Value	True if success, false if failure.

Example	`<script type="text/javascript">`
	`JSSetHttpErrorAction(0,"http://error.html");`
	`JSSetHttpErrorAction(1, "http://njijhasdf.sjkhdfkjsd");`
	`</script>`

12.7 AIRLOCK BROWSER NATIVE API

For more information, please refer to

http://outcoder.com/Products/AirlockBrowser/UserGuides/V2/ProgrammingGuide/.

Appendix I

SCAN ENGINE SETTINGS

The mobile computer is equipped with one of the barcode readers as follows. Reader availability depends on the hardware integrated on the mobile computer.

Scan Engine		ID
1D	*CCD*	*SM1*
1D	*Laser*	*SE955*
1D	*Extended Range Laser (ER Laser)*	*SE1524*
2D	*2D Imager*	*SE4500*
		SE4500 + PL4507
		SE4750MR
2D	*Near/far 2D Imager (N/F 2D)*	*EX25*

SYMBOLOGIES SUPPORTED

		Laser	ER Laser	2D	N/F 2D
Codabar		✓	✓	✓	✓
Code 11		✓	✘	✓	✓
Code 39	Code 39	✓	✓	✓	✓
	Trioptic Code 39	✓	✓	✓	✓
	Italian Pharmacode (Code 32)	✓	✓	✓	✘
Code 93		✓	✓	✓	✓
Code 128	Code 128	✓	✓	✓	✓
	GS1-128 (EAN-128)	✓	✓	✓	✓
	ISBT 128	✓	✓	✓	✓
Code 2 of 5	Chinese 25	✓	✘	✓	✘
	Industrial 25 (Discrete 25)	✓	✓	✓	✓
	Interleaved 25	✓	✓	✓	✓
	Convert Interleaved 25 to EAN-13	✓	✓	✓	✘
	Matrix 25	✘	✘	✓	✓
Composite Code	Composite CC-A/B	✘	✘	✓	✓
	Composite CC-C	✘	✘	✓	✓
	Composite TLC 39	✘	✘	✓	✘
GS1 DataBar (RSS)	GS1 DataBar-14 (RSS-14)	✓	✓	✓	✓
	GS1 DataBar Limited (RSS Limited)	✓	✓	✓	✓
	GS1 DataBar Expanded (RSS Expanded)	✓	✓	✓	✓
	Convert to UPC/EAN	✓	✓	✓	✘
Inverse	Inverse 1D barcodes	✘	✘	✓	✘
Korean 3 of 5		✘	✘	✓	✘
MSI		✓	✓	✓	✓
Postal Codes	Australian Postal	✘	✘	✓	✓
	Japan Postal	✘	✘	✓	✓
	Netherlands KIX Code	✘	✘	✓	✓
	US Postnet	✘	✘	✓	✓
	US Planet	✘	✘	✓	✓

	UK Postal	✘	✘	✓	✘
EAN / UPC	EAN-8	✓	✓	✓	✓
	EAN-8 Extend	✓	✘	✓	✘
	EAN-13	✓	✓	✓	✓
	Bookland EAN (ISBN)	✓	✓	✓	✓
	ISSN EAN	✘	✘	✓	✓
	UPC-A	✓	✓	✓	✓
	UPC-E	✓	✓	✓	✓
	Convert UPC-E to UPC-A	✓	✓	✓	✓
	UPC-E1	✓	✓	✓	✓
	Convert UPC-E1 to UPC-A	✓	✓	✓	✘
2D Symbologies	Aztec	✘	✘	✓	✓
	Data Matrix	✘	✘	✓	✓
	Maxicode	✘	✘	✓	✓
	MicroPDF417	✘	✘	✓	✓
	MicroQR	✘	✘	✓	✘
	PDF417	✘	✘	✓	✓
	QR Code	✘	✘	✓	✓

CONFIGURABLE PROPERTIES SUPPORTED

Depending on the reader types, supported configurable properties of symbologies will differ as listed below.

Symbologies	Properties	CCD	Laser	ER laser	2D	N/F 2D
General Properties						
Codabar	transmitCheckDigit	✓	✓	✓	✓	✓
	verifyCheckDigit	✓	✓	✓	✓	✓
	notisEditingType	✓	✓	✓	✓	✓
	enable	✓	✓	✓	✓	✓
	length1	✓	✓	✓	✓	✓
	length2	✓	✓	✓	✓	✓
	clsiEditing	✓	✓	✓	✓	✓
	notisEditing					
Code11	enable		✓		✓	✓
	length1		✓		✓	✓
	length2		✓		✓	✓
	numberOfCheckDigits		✓		✓	✓(1)
	transmitCheckDigit		✓		✓	✓
Code39	enable	✓	✓	✓	✓	✓
	length1	✓	✓	✓	✓	✓
	length2	✓	✓	✓	✓	✓
	checkDigitVerification	✓	✓	✓	✓	✓
	transmitCheckDigit	✓	✓	✓	✓	✓
	fullASCII	✓	✓	✓	✓	✓
	convertToCode32	✓	✓	✓	✓	
	convertToCode32Prefix	✓	✓	✓	✓	
TriopticCode39	enable		✓	✓	✓	✓
Korean3Of5	enable				✓	
Code93	enable	✓	✓	✓	✓	✓
	length1	✓	✓	✓	✓	✓
	length2	✓	✓	✓	✓	✓
Code128	enable	✓	✓	✓	✓	✓
	securityLevel	✓(2)				
GS1128	enable	✓	✓	✓	✓	✓
	fieldSeparator	✓	✓	✓	✓	✓
ISBT128	enable	✓	✓	✓	✓	✓
	concatenation				✓	✓
	concatenationRedundancy				✓	

Chinese2Of5	enable		✓		✓	
Industrial2Of5	enable	✓	✓	✓	✓	✓
	length1	✓	✓	✓	✓	✓
	length2	✓	✓	✓	✓	✓
Interleaved2Of5	enable	✓	✓	✓	✓	✓
	length1	✓	✓	✓	✓	✓
	length2	✓	✓	✓	✓	✓
	checkDigitVerification	✓	✓	✓	✓	✓
	transmitCheckDigit	✓	✓	✓	✓	✓
	convertToEan13		✓	✓	✓	
Matrix2Of5	enable	✓			✓	✓
	length1	✓			✓	✓
	length2	✓			✓	✓
	redundancy	✓			✓	
	checkDigitVerification	✓			✓	
	transmitCheckDigit	✓			✓	
UccCoupon	enable		✓	✓	✓	
GS1DataBar14	enable	✓	✓	✓	✓	✓
	convertToUpcEan		✓	✓	✓	
	securityLevel				✓	
GS1DataBarLimited	enable	✓	✓	✓	✓	✓
	convertToUpcEan		✓	✓	✓	
	securityLevel				✓	
GS1DataBarExpanded	enable	✓	✓	✓	✓	✓
	fieldSeparator	✓	✓	✓	✓	✓
	securityLevel				✓	
Msi	enable	✓	✓	✓	✓	✓
	length1	✓	✓	✓	✓	✓
	length2	✓	✓	✓	✓	✓
	checkDigitOption	✓	✓	✓	✓	✓
	transmitCheckDigit	✓	✓	✓	✓	✓
	checkDigitAlgorithm	✓	✓	✓	✓	
Ean8	enable	✓	✓	✓	✓	✓
	addon2	✓	✓	✓	✓	✓
	addon5	✓	✓	✓	✓	✓
	transmitCheckDigit	✓	✓	✓	✓	✓
	convertToEan13	✓	✓		✓	✓

Ean13	enable	✓	✓	✓	✓	✓
	addon2	✓	✓	✓	✓	✓
	addon5	✓	✓	✓	✓	✓
	convertToISBN	✓	✓	✓	✓	✓
	convertToISSN				✓	✓
	booklandISBNFormat	✓	✓	✓	✓	
	transmitCheckDigit	✓	✓	✓	✓	✓
UpcA	enable	✓	✓	✓	✓	✓
	addon2	✓	✓	✓	✓	✓
	addon5	✓	✓	✓	✓	✓
	transmitCheckDigit	✓	✓	✓	✓	✓
	transmitSystemNumber	✓	✓	✓	✓	✓(3)
	convertToEan13	✓	✓	✓	✓	✓
UpcE	enable	✓	✓	✓	✓	✓
	addon2	✓	✓	✓	✓	✓
	addon5	✓	✓	✓	✓	✓
	transmitCheckDigit	✓	✓	✓	✓	✓
	transmitSystemNumber	✓	✓	✓	✓	✓(4)
	convertToUpcA	✓	✓	✓	✓	✓
UpcE1	enable	✓	✓	✓	✓	✓
	addon2	✓	✓	✓	✓	✓
	addon5	✓	✓	✓	✓	✓
	transmitCheckDigit	✓	✓	✓	✓	
	transmitSystemNumber	✓	✓	✓	✓	
	convertToUpcA	✓	✓	✓	✓	
Composite	enableCc_C				✓	✓
	enableCc_AB				✓	✓
	enableTlc39				✓	
	enableUpcMode				✓	
	enableEmulationMode				✓	
USPostal	enablePlanet				✓	✓
	enablePostnet				✓	✓
	transmitCheckDigit				✓	✓
UKPostal	enable				✓	
	transmitCheckDigit				✓	
JapanPostal	enable				✓	✓
AustralianPostal	enable				✓	✓
DutchPostal	enable				✓	✓

USPSPostal	enable				✓	
UPUFICSPostal	enable				✓	
PDF417	enable				✓	✓
	transmitMode					
	escapeCharacter					
	transmitControlHeader					
MicroPDF417	enable				✓	✓
	code128Emulation				✓	✓
DataMatrix	enable				✓	✓
	fieldSeparator				✓	✓
	mirrorImage				✓	✓(5)
MaxiCode	enable				✓	✓
QRCode	enable				✓	✓
MicroQR	enable				✓	
Aztec	enable				✓	✓
Plessey	enable					✓
	unconventionalStop					✓
	transmitCheckDigit					✓
	length1					✓
	length2					✓
Telepen	enable					✓
	format					✓
	length1					✓
	length2					✓
User Preferences						

All	addonSecurityLevel	✓	✓	✓	✓	
	displayMode				✓	
	laserOnTime	✓	✓	✓	✓	
	negativeBarcodes				✓	
	pickListMode				✓	
	redundancyLevel	✓	✓	✓	✓	
	scanAngle		✓			
	securityLevel		✓	✓	✓	
	timeoutBetweenSameSymbol	✓	✓	✓	✓	✓
	transmitCodeIdChar	✓	✓	✓	✓	✓
	triggerMode	✓	✓	✓	✓	✓
	decodingIllumination				✓	✓
	decodingAimingPattern				✓	
	interCharGapSize				✓	
	timeoutPresentationMode				✓	✓
	decodingIlluminationPowerLevel				✓	
	aimerMode					✓
	centerDecoding					✓
	centerDecodingTolerance					✓

Remarks
(1) EX25 do not support Zero number.
(2) SM1 supports securityLevel only when the reader firmware version (BarCodeGetScannerVersion()) >= 1.15.
(3) EX25 only support Disable, Modulo_10 and French_CIP_HR.
(4) EX25 only support None and SysNumAndCtyCode for transmitSystemNumber.
(5) Ex25 only support Never and Auto for mirrorImage.