SEB Config Keys

Work in progress. Doesn't yet contain specific keys available only in SEB for iOS.

List of possible keys

Some keys are not available on all platforms, see remarks. Keys not available on a platform should be ignored by SEB Starter when parsing and using the settings. The SEB Windows configuration tool and the preferences system in SEB for Mac OS X should read and keep the key/values defined for another platform untouched and safe them together with the configured key/values for the own platform. Like this it will be possible to create a “merged” .seb file with specific settings for both platforms.

If keys required by one platform aren’t found in the settings file (because it has been created for another platform), then the default value has to be used. The default value therefore always is the “safer” or “safest”, most restrictive option.

additionalResources
Array of *additionalResource dictionaries* containing additional resources which can be used during an exam. Additional resources show up in the task bar similar as permitted applications (resource icon together with the resource’s title) and they are opened in an additional browser window (if it’s a resource type the SEB browser can display, like html, pdf etc.) or in the according third party application (which can open that resource’s file type). Resources can be external, then use an URL (all allowed URL types are possible, like http:// and file:// for local files). You can also embed resources into the .seb file. This makes particularly sense if you want to use resources offline.

Keys in the *additionalResource dictionary*:

- **active**
  Boolean indicating if the additional resource is active.
- **additionalResources**
  Array of further *additionalResources dictionaries*. If this array isn’t empty, then a popup-menu is created in the SEB task bar/dock when users click the icon (if the parent additionalResource was on the root (first) level, or a sub-menu entry in case it was not on the root level (second level and higher). With this key/value you can cascade additional resources. There is no limit for the number of levels, but generally it isn’t recommended to use more than three levels:
  - First (root) level: Icons in the SEB task bar/dock
  - Second level: Menu items in the popup menu which appears when users click an additional resource icon
  - Third level: Sub-Menu which opens when users click an additional resource menu item.
- **autoOpen**
  Boolean indicating whether the additional resource is opened automatically when SEB starts. If the resource is not opened automatically, then users have to click the resource’s icon in the SEB task bar to open it.
  Default value: <true/>
- **identifier**
  String containing an unique identifier for an *additionalResource dictionary*. This is used by the SEB browser to open only one window for an additionalResource. If an additionalResource is already open, then selecting it again brings its window to the foreground. For returning to the initial URL of an already open additionalResource, the *Restart Exam/Resource Button* can be used (in the SEB task bar/dock or in the browser window toolbar in SEB for Mac OS X).
  Format: level0.level1.level2 … (example: 0.2.5)
- **title**
  String with the resource title which is displayed in the task bar. There doesn’t have to be a *title* string necessarily, the resource title will then be derived from the URL, the filename or if the resource is a web page then the browser receives the title from the web server when loading the resource. If a *title* string is indicated, then this strings takes precedence over a web page title.
- **URL**
  String containing the URL or filename of the resource. If the resource is external, then the URL has to start with the right URL scheme, for example http:// or file://. If it is an embedded resource, then this field contains an URL with the scheme embedded://, followed with the the resource’s filename or a directory path in case the embedded resource inside the gzipped data contains directories/subdirectories. The filename should in general contain the file extension in order for SEB to figure out with which application (or the built-in browser) to open the resource. If the URL doesn’t contain a file extension or file name (only the path to the directory containing the resource), then the SEB browser tries to loads an index.html file (as with a similar URL of a web server).
- **URLFilterRules**
  Array of dictionaries each containing a set of URL filter rules, see root-level *URLFilterRules* key for description.
- **resourceData**
  Data (Base64 encoded) containing a gzip archive of the embedded resource(s). **If there is no *resourceData*, then the resource is external**.
- **resourceIcons**
  Array of dictionaries containing the resource’s icon in multiple formats and resolutions (if necessary, SEB for Mac OS X is able to read and display the Windows ICO format too, which is also used for Favicons).

Keys in the *resourceIcons* array dictionaries:

- **format**
  Integer with a value representing which format the *iconData* has.

  Possible values for the *format* key:

```
enum {
    iconFormatMacIcon               = 0,
    iconFormatWinIco                = 1,
    iconFormatPNG                   = 2
};
typedef NSUInteger iconFormats;
```

- **resolution**
  Integer with the dimensions of the icon in pixels. Icons can only be square, so this value represents both width and height in pixels. The resolution value is only necessary if you use several icons in the PNG icon format (*format = iconFormatPNG*), as both the Mac Icon and the Windows Ico format can contain several bitmap images with various resolutions. Then SEB uses the size which is most appropriate for the destination icon size used in the SEB task bar and scales the PNG icon down/up if still necessary.
- **iconData**
  Data (Base64 encoded) containing the resource's icon.

First available in version

Windows	2.2
macOS	–
iOS	–

additionalResourcesIdentifierCounter
Integer containing the highest used identifier value for the sub-key additionalResources -> identifier. When adding a new *additionalResource dictionary* to the *additionalResources array*, SEB uses the number found in *additionalResourcesIdentifierCounter* and increments this by one.

allowBrowsingBackForward
Boolean indicating if browsing back to previously visited pages (and forward again) according to the browser history of this browser session (since SEB was started) is allowed or not.

Changes: This key affects only the main browser window. For all other browser windows see key *newBrowserWindowNavigation*.

Default value: <false/>

Changed functionality first available in version

Windows	2.2
macOS	–
iOS	2.1.10

allowDisplayMirroring
Boolean, indicating if mirroring the main display to another (for example an AirPlay Display) should be allowed.

Default value: <false/>
Currently Mac only

allowDictionaryLookup

Boolean, indicating if looking up text elements on a web site using the 3-finger tap on a trackpad or ctrl-cmd-D should be allowed.

Default value: <false/>
Mac only

allowDownUploads
Boolean, indicating if downloading and uploading files is allowed. This setting does not affect .seb config files (see key *downloadAndOpenSebConfig*).

Default value: <true/>
Currently Mac only

allowedDisplayBuiltin
Boolean, indicating if the built-in display (if available) should be used when only one display is allowed or when switching off display mirroring (see key *allowDisplayMirroring*).

Default value: <true/>
Currently Mac only

allowedDisplaysMaxNumber
Integer value indicating the maximum allowed number of connected displays. If the user tries to move an SEB window to a connected display and this display exceeds the allowed number, then it is blanked with an orange full screen window.

Default value: <integer>1</integer>
Currently Mac only

allowAudioCapture
Boolean indicating if web applications are allowed to access the default microphone (using HMTL 5 APIs).

Default value: <false/>

First available in version

Windows	2.1.6
macOS	–
iOS	–

allowFlashFullscreen
Boolean, indicating if Flash is allowed to switch on fullscreen presentation (mainly used in Flash video players).

Default value: <false/>
Mac only

allowPDFPlugIn
Boolean indicating if the Acrobat Reader PDF plugin (insecure) will be allowed to display PDF files in browser windows.

Default value: <false/>
Currently Mac only

allowPreferencesWindow
Boolean indicating if users are allowed to open the preferences window on exam clients. Usually it should be disabled besides for debugging purposes.

Default value: <true/>
Mac only

allowQuit
Boolean indicating if quitting SEB by key combination, menu entry or window closing button is allowed. This flag does not affect the *quit link feature* (if *quitURL* is set and detected, SEB quits regardless of this flag).

Default value: <true/>
Currently Mac only

allowScreenSharing
Boolean indicating if screen sharing is allowed to be used.

Default value: <false/>
Currently Mac only

allowSpellCheck
Boolean indicating if users are allowed to use the browser's spelling check (all kinds of, checking spelling and grammar, auto correction while typing etc.).

Default value: <false/>

allowSwitchToApplications
(replaces key ShowSebApplicationChooser on Windows)

Boolean indicating if users are allowed to switch to permitted applications. This also indicates if the application chooser is displayed or not.

Default value: <false/>

allowUserSwitching
Boolean indicating if fast user switching is allowed. When using SEB on students' own computers (or where examinees have access to other user accounts on the exam Macs) and especially when using third party applications during the exam, this key should be set to false. Otherwise when the computer is put to sleep and woken up again, in the lock screen "Switch user" is shown and when using a third party application the fast user switching option is displayed on the right side of the menu bar.
This feature requires the Mac OS X SEB Service being installed (this is why its default value is false).

Default value: <true/>
Mac only

allowVideoCapture
Boolean indicating if web applications are allowed to access the default camera (using HMTL 5 APIs).

Default value: <false/>

First available in version

Windows	2.1.6

macOS	–
iOS	–

allowVirtualMachine
Boolean indicating if SEB is allowed to run on a virtual machine (e.g. for exams in virtual desktop environments) or not (in order to prevent potential manipulations).

Default value: <false/>

allowWLAN
Boolean indicating if the WLAN control should be displayed in the SEB task bar.

Default value: <false/>
Currently Windows only

audioControlEnabled
Boolean indicating if the audio control should be displayed in the SEB task bar.

Default value: <false/>

First available in version

Windows	2.2
macOS	–
iOS	–

audioMute
Boolean indicating if audio should be muted when the SEB session is started.

Default value: <false/>

First available in version

Windows	2.2
macOS	–
iOS	–

audioSetVolumeLevel
Boolean indicating if the audio volume level should be set to the value of *audioVolumeLevel* when the SEB session is started.

Default value: <false/>

First available in version

Windows	2.2
macOS	–
iOS	–

audioVolumeLevel
Integer indicating the initial audio level (in percent) when the SEB session is started.

Default value: <integer>25</integer> (25% audio level)

First available in version

Windows	2.2
macOS	–
iOS	–

blockPopUpWindows

Boolean indicating if pop-up windows (often advertisement) opened by JavaScript without an user action such as a button click are blocked.

Default value: <false/>
Currently Mac only

browserMessagingPingTime
Integer with a value representing a timeframe for the XULRunner seb browser to send a keep alive ping message to the socket server.

Default value: <integer>120000</integer> (2 minutes)
Windows only

browserMessagingSocket
String containing a service URL for the socket server.

Default value: <string>ws:\\localhost:8706</string>
Windows only

browserScreenKeyboard
Boolean instructing the seb/seb2 XULRunner/Firefox browser whether it should send a socket message when a text box gets and loses input focus. This key cannot be set in the config tool UI but gets automatically set with the key *touchOptimized.*

Default value: <false/>
Windows only

browserURLSalt
Boolean instructing the seb/seb2 XULRunner/Firefox browser whether it should use the full URL of a HTTP request as salt when generating the Browser Exam Key request header field.

Default value: <true/>
Windows only

browserUserAgent
String suffix which is appended to the current user agent, determined by the other browserUserAgent setting keys. Note that the " SEB/<version number>" string is always appended in addition.

Default value: <string></string>

Windows	2.2
macOS	2.1.3
iOS	2.1.12

browserUserAgentMac
Integer with a value representing one *browserUserAgentModeMac:* Use default user agent string on the Mac (depends on installed Safari/WebKit version and therefore can differ on exam clients) or a custom user agent string, see key *browserUserAgentMacCustom*.
Possible values:

```
enum {
  browserUserAgentModeMacDefault          = 0,
  browserUserAgentModeMacCustom           = 1
};
typedef NSUInteger browserUserAgentModeMac;
```

Default value: <integer>0</integer> (browserUserAgentModeMacDefault)

browserUserAgentMacCustom
String representing a custom user agent string (SEB appends its version number automatically).

Default value: <string></string>

browserUserAgentWinDesktopMode
Integer with a value representing one *browserUserAgentModeWinDesktop:* Use default user agent string in SEB Windows running in desktop mode (starting SEB 2.2 depends on embedded Firefox version) or a custom user agent string, see key *browserUserAgentWinDesktopModeCustom*.
Possible values:

```
enum {
  browserUserAgentModeWinDesktopDefault          = 0,
  browserUserAgentModeWinDesktopCustom           = 1
};
typedef NSUInteger browserUserAgentModeWinDesktop;
```

Default value: <integer>0</integer> (browserUserAgentModeWinDesktopDefault)

browserUserAgentWinDesktopModeCustom
String representing a custom user agent string for SEB Windows running in desktop mode (SEB appends its version number automatically).

Default value: <string></string>

browserUserAgentWinTouchMode
Integer with a value representing one *browserUserAgentModeWinTouch:* Use the default (starting SEB 2.2 depends on embedded Firefox version), an iPad-like and a custom browser user agent string (see key *browserUserAgentWinTouchModeCustom)* for SEB Windows running in the touch optimized mode (on tablet computers).
Possible values:

```
enum {
  browserUserAgentModeWinTouchDefault          = 0,
  browserUserAgentModeWinTouchIPad             = 1,
  browserUserAgentModeWinTouchCustom           = 2
};
```

```
typedef NSUInteger browserUserAgentModeWinTouch;
```

Default value: <integer>0</integer> (browserUserAgentModeWinTouchDefault)

browserUserAgentWinTouchModeCustom
String representing a custom user agent string for SEB Windows running in the touch optimized mode (SEB appends its version number automatically).

Default value: <string></string>

browserViewMode
Integer with a value representing one of the *browserViewModes:* Either use a window for the SEB browser or display the browser full screen. Possible values:

```
enum {
  browserViewModeWindow                 = 0,
  browserViewModeFullscreen             = 1,
  browserViewModeTouch                  = 2
};
typedef NSUInteger browserViewModes;
```

Default value: <integer>0</integer> (browserViewModeWindow)

browserWindowAllowReload
Boolean indicating if reload is allowed in main browser window (for additional windows see *newBrowserWindowAllowReload*).

Default value: <true/>

First available in version

Windows	2.2
macOS	2.1.3
iOS	2.1.10

chooseFileToUploadPolicy
Integer with a value representing one of the *chooseFileToUploadPolicies*: SEB can let the user choose the file to upload manually (as usual) or automatically choose the same file which was downloaded before. There are three possible policies to choose the file to upload:

- **manually with file requester**
- **by attempting to upload same file downloaded before**: If the file is not found, a file requester is presented and the user can choose some other file manually.
- **by only allowing to upload the same file downloaded before**: If the file is not found, an error message is presented. This setting might bring additional security, because only files which have been downloaded before (in the same browser session, means since SEB was started) can be uploaded. If several files have been downloaded, pressing the choose file (or similarly named) button in the browser window will first choose the file most recently downloaded, pressing the button several times will cycle through all the files downloaded in this session.

Possible values:

```
enum {
  manuallyWithFileRequester             = 0,
  attemptUploadSameFileDownloadedBefore = 1,
  onlyAllowUploadSameFileDownloadedBefore = 2
};
typedef NSUInteger chooseFileToUploadPolicies;
```

Default value: <integer>0</integer> (manuallyWithFileRequester)
Currently Mac only

copyBrowserExamKeyToClipboardWhenQuitting
Boolean indicating that the Browser Exam Key should be copied to the clipboard when quitting SEB MacOSX. Since normally SEB clears the clipboard when quitting, with this option the exam admin finds the current exam key in the clipboard and can paste it into the settings of his exam system. This option is significant only for the SEB MacOSX preferences window and not saved into a .seb file!

Default value: <false/>
Mac only

createNewDesktop
Boolean indicating if SEB should be executed in a newly created desktop window (in fullscreen mode), such that e.g. the task bar and the start menu at the bottom edge of the screen are blanked out. **Important: The *createNewDesktop* setting is valid for the entire session.** That means the setting for *createNewDesktop* which is defined in local SEB client settings (*SebClientSettings.seb* file inside of Windows folders *LocalAppData* or *ProgramData*) cannot be overridden by loading a .seb file for starting an exam with another setting value for *createNewDesktop*. You can only reconfigure the local client settings by loading a .seb file *for configuring the client*, to apply this change, SEB needs to be quit and restarted manually.

Default value: <true/>
Windows only. The create new desktop mode is no longer available in SEB for Windows starting version 2.1.6 and the value of this key is ignored.

cryptoIdentity
Data with the 20 bytes hash of the public key used for encrypting the .seb file. This is saved for convenience, so that in the SEB MacOSX preferences window in the Config File tab the same crypto identity can be chosen in the popup list. This key/value is not necessary for loading a .seb file encrypted using an identity (the information for that identity is contained in the header of the encrypted .seb file, see chapter 2).

Default value: <data></data> (none selected)
Currently Mac only, not used anymore in version >= 2.0RC

downloadAndOpenSebConfig
Boolean indicating if .seb config files should be downloaded and opened, regardless if downloading and opening of other file types is allowed or not.

Default value: <true/>
Currently Mac only

downloadDirectoryOSX
String representing the path of the directory to which downloaded files will be saved. Paths containing the home directory of the current user are abbreviated with the tilde symbol ~.

Default value: <string>~/Downloads</string>
Mac only. Windows equivalent: downloadDirectoryWin

downloadDirectoryWin
String representing the path of the directory to which downloaded files will be saved. Paths containing the home directory of the current user should contain a Windows compatible placeholder for the home directory (eventually also for other special system directories).

Default value: *not specified yet*
Windows only. Mac equivalent downloadDirectoryMac

downloadPDFFiles

Boolean indicating if PDF files should be downloaded or displayed online inside the browser window.

Default value: <false/>
Currently Mac only

embeddedCertificates
Array of dictionaries which contain SSL client certificates and cryptographic identities with their properties which are embedded into settings. When SEB loads a .seb settings file with embedded certificates or identities, then it installs them into the macOS Keychain or into the XULRunner certificate database (TLS/SSL server certificates) or Windows Certificate Store (identities).

Keys in the *embeddedCertificates* dict:

- **certificateData**
  Data (Base64 encoded) of the certificate/identity.
- **certificateDataBase64**
  String with Base64 encoded data of the TLS/SSL certificate. Identities are only saved as certificateData.
- **certificateDataWin**
  String with Base64 encoded data of the TLS/SSL certificate. **This key is deprecated** and only in use for certificate type *certificateTypeSSLServerCertificate* for downwards compatibility to SEB Windows versions < 2.2. **Don't rely that this key contains the certificate data** but always check first for data in key **certificateDataBase64**, as future versions of SEB won't save data in this key anymore.
- **name**
  String containing the name of the certificate/identity. The name might be just the common name, the Email address, a combination of both and the public key hash value.
- **type**
  Integer with a value representing the type of the certificate. User interface strings (English): "SSL Certificate", "Identity", "CA Certificate", "Debug Certificate".

  Possible values for the *type* key:

```
enum {
    certificateTypeSSL          = 0,
    certificateTypeIdentity     = 1,
    certificateTypeCA           = 2,
    certificateTypeSSLDebug     = 3
};
typedef NSUInteger certificateTypes;
```

enableAppSwitcherCheck
Boolean indicating whether SEB checks for the command key being held down while SEB is starting up. This prevents using the application switcher to mess with SEB's kiosk mode.

Default value: <true/>
Mac only

enableBrowserWindowToolbar
Boolean indicating if a toolbar is displayed on top of the browser window which can also be hidden by the user if it's not used.

Default value: <false/>
Currently Mac only

enableJava
Boolean indicating if Java Applets are enabled. Starting SEB 2.0 this option is disabled by default because Java applets are considered a potential security risk.

Default value: <false/>
Currently Mac only

enableJavaScript
Boolean indicating if JavaScript is enabled. Please note that most modern websites need JavaScript for full functionality.

Default value: <true/>
Currently Mac only

enableLogging
Boolean indicating if SEB writes a log.

Default value: <false/>

enablePlugIns
Boolean indicating if web plugins like Flash are enabled. For security reasons it's recommended to disable this option if you don't use any plugin content.

Default value: <true/>
Currently Mac only

enableSebBrowser
Boolean indicating if the SEB browser should be used. If you don‘t want to use any browser in SEB, because SEB Starter should only act as a kiosk application starting up another application in a kiosk mode (for example a virtual desktop infrastructure client), then set *enableSebBrowser = false*.

Default value: <true/>

enableZoomPage
Boolean indicating if pages can be zoomed with cmd +/- or the commands in the view menu and buttons in browser window toolbar (Mac/Win 2.2 or higher) or with Ctrl-Mousewheel (Win).

Default value: <true/>

enableZoomText
Boolean indicating if text in browser windows can be zoomed with ctrl - cmd +/- or the commands in the view menu and buttons in browser window toolbar (Mac/Win 2.2 or higher) or with Ctrl-Mousewheel (Win).

Default value: <true/>

examKeySalt
Data representing a random salt value which is used to generate the browser exam key.

No Default value.

exitKey1
(replaces key B1 in MsgHook.ini on Windows)
Integer value representing a virtual key code of the first function key to be pressed and held down together with two other keys in the right order to exit SEB.

No default value
Windows only

exitKey2
(replaces key B2 in MsgHook.ini on Windows)
Integer value representing a virtual key code of the second function key to be pressed and held down together with two other keys in the right order to exit SEB.

No default value
Windows only

exitKey3
(replaces key B3 in MsgHook.ini on Windows)
Integer value representing a virtual key code of the third function key to be pressed and held down together with two other keys in the right order to exit SEB.

No default value
Windows only

forceAppFolderInstall
Boolean indicating if SEB enforces to be installed in an Applications folder (/Applications or ~/Applications). When true, SEB quits when it isn't installed in an Applications folder.

Default value: <true/>
Mac only

hashedAdminPassword
String containing Base16 encoded data representing a SHA256 hash of the password required to enter the preferences window (Mac) or to open a .seb configuration file for editing (Win/Mac).

Default value: <string></string> (empty string = no admin password set)

hashedQuitPassword
String containing Base16 encoded data representing a SHA256 hash of the password which is prompted when users try to quit SEB.

Default value: <string></string> (empty string = no quit password set)

hideBrowserWindowToolbar
Boolean indicating if the browser window toolbar should be hidden by default. Users can unhide the toolbar in the view menu or the contextual menu on the browser window title bar. In full screen browser view mode, the toolbar is auto hidden with this setting and appears when users move the mouse towards the screen's top border.

Notice: With the toolbar being auto hidden in full screen mode, when it appears, also the menu bar will appear (OS X restriction). So if you don't want users to have any access to the menu bar, don't use *hideBrowserWindowToolbar* together with *browserViewModeFullscreen*.

Default value: <false/>
Mac only

hookKeys
Boolean indicating if SEB should intercept input functions like key combinations (for example Alt+F4) or right mouse click.

Default value: <true/>
Windows only

List of keys for intercepting keyboard keys and input functions
Below keys are listed which represent intercepted input functions and boolean values for their default setting (on the second line).
The setting for function keys doesn't affect the exit keys.

- **enableEsc**
  <false/>

- **enableCtrlEsc**
  <false/>
- **enableAltCtrl**
  <true/>
- **enableAltEsc**
  <false/>
- **enableAltMouseWheel**
  Boolean indicating if ALT + mouse wheel is enabled for history browsing in the XULRunner browser.

- **enableAltTab**
  <true/>
- **enableAltF4**
  <false/>
- **enablePrintScreen**
  <false/>
- **enableRightMouse**
  <false/>
- **enableStartMenu**
  <false/>
- **enableF1**
  <false/>
- **enableF2**
  <false/>
- **enableF3**
  <false/>
- **enableF4**
  <false/>
- **enableF5**
  <true/>
- **enableF6**
  <false/>
- **enableF7**
  <false/>
- **enableF8**
  <false/>
- **enableF9**
  <false/>
- **enableF10**
  <false/>
- **enableF11**
  <false/>
- **enableF12**
  <false/>

All keys Windows only

ignoreExitKeys
Boolean indicating if SEB is ignoring the exit keys for quitting SEB by pressing and holding down three function keys in a specific order (which are defined with *exitKey1-3*). This key has to be true if you want to use just a quit password.

Default value: <true/>
Windows only

ignoreQuitPassword
Boolean indicating if SEB is ignoring the quit password and can only be quit manually by pressing and holding down three function keys in a specific order (which need to be defined with *exitKey1-3*). If you want to use just a quit password also in SEB Windows, then set *ignoreExitKeys* to true.

Default value: <false/>
Windows only

Keys for Windows Security Screen registry values
These boolean values indicate if the respective options are visible and active on the Windows Security Screen which appears when the key combination Ctrl + Alt + Del is pressed.
There is an *insideSeb...* and **an optional *outsideSeb...*** variant of each of these flags. The options *insideSeb...* define the values inside of SEB (which are thus valid during an exam, when SEB is running). The options *outsideSeb...* define the values outside of SEB (which are thus valid in normal use, when SEB is not running). The standard defaults are to disable (= <false/>) all options inside SEB.
Normally exam admins should not have to use any *outsideSeb...* options. Especially on unmanaged computers (like student's own notebooks) it should not be assumed that all computers have the standard windows setting (all options enabled)! The *outsideSeb...* options could however be used in a special "recovery" .seb settings file for cases when setting and resetting the registry options didn't work properly.
There is no default value for the options outside SEB (at least if nothing went wrong badly, in that case the default value would be assumed *e nabled*/<true/>, see below):
Instead, SEB Windows 2.0 assures to set and reset each registry values strictly in an atomic way:

1. When SEB starts: Check if there is already a persistently saved registry option

- If yes: Something went wrong, possibly SEB crashed or it has been killed instead of quitted properly last time. Don't overwrite the persistently saved registry option. Proceed with step 2.
- if not: Save the current value of the registry option persistently (in a file inside it's settings directory), this saved value therefore represents the outside SEB value.

1. Set the registry option to the value of the *insideSeb...* option.

1. Perform a check to assure that the registry option was correctly set to the *insideSeb...* value (read it again and compare if the value is what it should be). If not, then react depending on the *sebServicePolicies*:

- *ignoreService* -> no reaction (just write it into log if it's active),
- *indicateMissingService* -> display error message, but offer option to continue exam
- *forceSebService* -> display error message with only option to quit SEB

1. When SEB quits, then reset the registry values:

- If there is an *outsideSeb...* option flag in the currently active settings, then reset the registry value to the value of the *outsideSeb...* flag
- Otherwise:
    - If there is a persistently saved registry option value (there should be one, otherwise something went wrong badly...) reset the registry option value to the persistently saved value.
    - If there is no persistently saved registry option value (something went wrong badly), reset the registry option to *enabled/true*. Write a warning into the log.
- Check if the registry option value was correctly reset (read it again and compare if the value is what it should be).
    - If yes: delete the persistently saved registry option value.
    - If not, don't delete the persistently saved registry option value, display an error message saying something like "The registry setting for *<name of the key>* could not be reset. Try to restart and quit SEB again, possibly after restarting your computer."
- Quit SEB.

insideSebEnableChangeAPassword / outsideSebEnableChangeAPassword
Boolean indicating if the button "Kennwort ändern..." or "Change a password..." is activated.

insideSebEnableEaseOfAccess / outsideSebEnableEaseOfAccess
Boolean indicating if the button "Erleichterter Zugriff" or "Ease of Access" in the lower left corner is activated, which offers help e.g. to visually or aurally handicapped persons, like the Magnifier Glass.

insideSebEnableLockThisComputer / outsideSebEnableLockThisComputer
Boolean indicating if the button "Computer sperren" or "Lock this computer" is activated.

insideSebEnableLogOff / outsideSebEnableLogOff
Boolean indicating if the button "Abmelden" or "Log off" is activated.

insideSebEnableShutDown / outsideSebEnableShutDown
Boolean indicating if the button "Herunterfahren" or "Shutdown" in the lower right corner is activated.

insideSebEnableStartTaskManager / outsideSebEnableStartTaskManager
Boolean indicating if the button "Task-Manager starten" or "Start Task Manager" is activated.

insideSebEnableSwitchUser / outsideSebEnableSwitchUser
Boolean indicating if the button "Benutzer wechseln" or "Switch User" is activated.

insideSebEnableVmWareClientShade / outsideSebEnableVmWareClientShade
Boolean indicating if the "Shade" bar at the upper edge of a virtual desktop is activated, if existent.

Default value for all the *insideSEB...* keys: <false/>

Default value for all the *outsideSEB...* keys: <true/>
(used only in rare cases when resetting keys didn't work properly)
Windows only

killExplorerShell
Boolean indicating if the Windows Explorer Shell should be killed when starting SEB up and restarted before quitting SEB. This makes sense in some scenarios when SEB should not run on a new desktop *(createNewDesktop = false)*.

Default value: <false/>
Windows only.

logDirectoryOSX
String representing the path of the directory to which log files will be saved. Paths containing the home directory of the current user are abbreviated with the tilde symbol ~. The special value <string>*NSTemporaryDirectory*</string> will use the temporary directory for the current user.

Default value: <string>NSTemporaryDirectory</string>
Mac only. Windows equivalent: logDirectoryWin

logDirectoryWin
String representing the Windows formatted path of the directory to which log files will be saved. Paths containing the home directory of the current user should contain a Windows compatible placeholder for the home directory (eventually also for other special system directories).

Default value: <string></string>
Windows only. Mac equivalent: logDirectoryOSX

logLevel
Integer with a value representing a *SEBLogLevel*. *Error* includes fatal application and browser level errors, *Warning* are non-fatal but non-expected or security affecting events. *Info* includes most user actions including all browser navigation actions. *Debug* is reserved for information which is only necessary for in-deep program code debugging. The log will contain the selected log level plus all levels with a lower value, a log with the *Verbose* level contains events of all levels.

Possible values:

```
enum {
    SEBLogLevelError                = 0,
    SEBLogLevelWarning              = 1,
    SEBLogLevelInfo                 = 2,
    SEBLogLevelDebug                = 3,
    SEBLogLevelVerbose              = 4
};
typedef NSUInteger SEBLogLevel;
```

Default value: integer>1</integer> (SEBLogLevelWarning)
Currently Mac only.

mainBrowserWindowHeight
String indicating the height in pixels or as percentage (followed by the % sign) of the main browser window (if it's not in full screen mode). A height of 100% means that SEB opens the window with the full usable screen height (full height of the current screen minus the height of the SEB dock/task bar if *showTaskBar = true* and on a Mac minus the height of the menu bar if *showMenuBar = true*).

Default value: <string>100%</string>

mainBrowserWindowPositioning
Integer with a value representing one of the *browserWindowPositionings*.

Possible values:

```
enum {
    browserWindowPositioningLeft            = 0,
    browserWindowPositioningCenter          = 1,
    browserWindowPositioningRight           = 2
};
typedef NSUInteger browserWindowPositionings;
```

Default value: <integer>1</integer> (browserWindowPositioningCenter)

mainBrowserWindowWidth
String indicating the width in pixels or as percentage (followed by the % sign) of the main browser window (if it's not in full screen mode). A width of 100% means that SEB opens the window with the full width of the current screen.

Default value: <string>100%</string>

minMacOSVersion
Integer with a value representing one of the *SEBMinMacOSVersion*. This minimum macOS version is enforced, SEB will refuse to run on an older version.

Possible values:

```
enum {
    SEBMinOSX10_7                   = 0,
    SEBMinOSX10_8                   = 1,
    SEBMinOSX10_9                   = 2,
    SEBMinOSX10_10                  = 3,
    SEBMinOSX10_11                  = 4,
    SEBMinMacOS10_12                = 5
};
typedef NSUInteger SEBMinMacOSVersion;
```

Default value: <integer>0</integer> (Mac OS X 10.7)

monitorProcesses
Boolean indicating if SEB is monitoring which processes (and applications) are running during an exam. Third party applications and other processes which are not permitted to run during an exam (not having an entry in the permittedProcesses dictionary or being explicitly blacklisted in prohibitedProcesses or being an exception like some specific system processes) are killed by SEB if they start up during an exam. If they are running when SEB is started, then an alert/dialogue window is displayed to tell the user to quit the not permitted (prohibited) applications and to restart SEB afterwards or to let SEB kill the applications risking that there could be data loss. SEB kills not permitted background processes itself, without user confirmation. Applications which allow to be terminated nicely in OS X are automatically terminated (also not asking the user).

Default value: <false/>

newBrowserWindowAllowReload
Boolean indicating if reload is allowed in additional browser windows (for main window see *browserWindowAllowReload*).

Default value: <true/>

newBrowserWindowByLinkBlockForeign
Boolean indicating if hyperlinks which direct to a different host than the one of the current page should be ignored.

Default value: <false/>

Currently Mac only

newBrowserWindowByLinkHeight
String indicating the height in pixels or as percentage (followed by the % sign) of browser windows opened by a link requesting to be opened in a new browser window (target="_blank" or target="_new"). A height of 100% means that SEB opens the window with the full usable screen height (full height of the current screen minus the height of the SEB dock/task bar if *showTaskBar = true* and on a Mac minus the height of the menu bar if *showMenuBar = true*).

Default value: <string>100%</string> (= full usable screen height)

newBrowserWindowByLinkPolicy
Integer with a value representing one of the *newBrowserWindowPolicies*.

Possible values:

```
enum {
    getGenerallyBlocked                 = 0,
    openInSameWindow                    = 1,
    openInNewWindow                     = 2
};
typedef NSUInteger newBrowserWindowPolicies;
```

Default value: <integer>2</integer> (openInNewWindow)

newBrowserWindowByLinkPositioning

Integer with a value representing one of the *browserWindowPositionings*.

Possible values:

```
enum {
    browserWindowPositioningLeft            = 0,
    browserWindowPositioningCenter          = 1,
    browserWindowPositioningRight           = 2
};
typedef NSUInteger browserWindowPositionings;
```

Default value: <integer>2</integer> (browserWindowPositioningRight)

newBrowserWindowByLinkWidth
String containing the width in pixels or as percentage (followed by the % sign) of screen width of browser windows opened by a link requesting to be opened in a new browser window (target="_blank" or target="_new"). A width of 100% means that SEB opens the window with the full width of the current screen.

Default value: <string>1000</string>

newBrowserWindowByScriptBlockForeign
Boolean indicating if hyperlinks which direct to a different host than the one of the current page should be ignored.
Default value: <false/>

Currently Mac only

newBrowserWindowByScriptPolicy
Integer with a value representing one of the *newBrowserWindowPolicies* (see *newBrowserWindowByLinkPolicy*) for hyperlinks opened from

JavaScript or plug-ins (like Flash).

Default value: <integer>2</integer> (openInNewWindow)
Currently Mac only

newBrowserWindowNavigation
Boolean indicating if browsing back to previously visited pages (and forward again) according to the browser history of this browser session (since SEB was started) is allowed or not. This key affects only additional browser windows. For the main browser window see key *allowBrowsingBackForward*.

Default value: <true/>

Functionality first available in version

Windows	2.2
macOS	2.2
iOS	2.1.10

newBrowserWindowShowReloadWarning
Boolean indicating if a warning should be displayed before reloading the web page in an additional browser window. For the main browser window see key *showReloadWarning*.

Default value: <false/>

openDownloads
Boolean indicating if downloaded files will be opened (with the according application, which currently has to be set correctly in the system for each used file type, in a future SEB version it will be possible to include file/MIME types and according applications in the .seb file).

Default value: <false/>
Currently Mac only

originatorVersion
Version information about the SEB application which saved the .seb configuration file, in the format SEB_OS_version_build.

Example: SEB_OSX_2.0pre2_112E

oskBehavior
Integer with a value representing one of the.

Default value: <false/>
Windows only

permittedProcesses
(replaces key permittedApplications on Windows)

Array of dictionaries containing the properties of permitted third party applications and processes which are permitted to run during an exam. Permitted applications (which have a *title* value) show up in the application chooser, they can be used during an exam in addition to the SEB browser. Permitted processes (which don't have a *title* value) don't appear in the application chooser, but they are allowed to run in background even when ***monitorProcesses*** is true (they are on the whitelist).

Keys in a *permittedProcesses* dict:

- **active**

Boolean indicating if the permitted process is active.

- **os**
  Integer with a value representing on which operating system the permitted process runs.

  Possible values for the *os* key:

```
enum {
    operatingSystemOSX                  = 0,
    operatingSystemWin                  = 1
};
typedef NSUInteger operatingSystems;
```

- **title**
  String of application title which is displayed in the application chooser. Background processes don't have a *title* value, because they don't need to be chosen by users.
- **description**
  String containing a description of the process. This is only displayed in the SEB configuration tool, preferences window and in logs. It should explain what kind of process this is, because this might not be obvious only from the *name*.
- **executable**
  String of the process name (usually the file name of the executable).
- **originalName**
  String containing the original filename meta data of the executable (only available in Windows).
- **allowedExecutables**
  String with a comma separated list of names of the window handling processes of a permitted application. These are necessary if a Windows application (process) doesn't provide the *mainWindowHandle* property. This is usually the case for Java applications like OpenOffice (see SEB documentation for examples).
  Default value: <string></string> (empty string = no window handling processes)
  Windows only
- **identifier**
  String of the process identifier in reverse domain notation (Mac) or the string or substring of the main window title of a process which doesn't have a MainWindow handle (Win), this is usually the case with Java applications.
  Examples: <string>com.apple.mail</string> (Mac), <string>OpenOffice</string> (Win)
- **autostart**
  (replaces root level key AutostartProcess on Windows)
  Boolean indicating whether the process is started automatically together with SEB. Usually, the SEB browser component is started automatically.
  Default value: <false/>
  If key doesn't exist, it has the same effect as if key=false.
- **autohide**
  Boolean indicating whether a process gets hidden if it shows its menu bar, an alert, dialogue or other window (this means in OS X the process tries to become „active"). Usually both permitted applications and processes are allowed to display windows and become active, with this flag set to true permitted processes are only allowed to run in background (when ***monitorProcesses*** is true), but not to display any user interface elements. Processes with this flag set don't have any icon in the SEB task bar and cannot be selected by Alt-Tab/Cmd-Tab.
  Default value: <true/>
  If key doesn't exist, it has the same effect as if key=false.
- **path**
  String of path to the application executable's directory (excluding the file name, see key *name*). If the path is not given or relative (not absolute from the root directory/drive), then SEB searches the current and system provided paths for applications, see flag *allowUserToChooseApp*.
- **allowUserToChooseApp**
  Boolean indicating if the user is presented a requester/dialog window allowing to choose the third party application if it cannot be found at the paths specified (instead of just displaying an error message). Only applications matching the other criteria specified in the *permittedProcesses* dictionary (like *name, identifier, signature*) are accepted.
  Default value: <false/>
- **arguments**
  Array of dictionaries containing the arguments to append to the *path+name* of the application/process.

  Keys in the *arguments* dict:
- **active**
  Boolean indicating if the argument will be used/appended to the *path+name* of the executable (meant for testing).
- **argument**
  String representing one argument to append to the *path+name* of the application/process executable.

- **signatures**
  Array of dictionaries containing metadata and the actual signatures of the application/process executable. Used to identify the process binary/application securely. Multiple signatures are possible to identify several versions of a binary. The signature algorithm is platform specific and has yet to be defined.

  Keys in the *signatures* dict:
- **description**
  String containing a description of the signature. For example it should be mentioned which application/binary version is covered by the

signature.

- **relativePath**
  By default the signature is calculated over the executable file of the process (identified by the key *name*). If you want the signature be calculated using for example the directory the executable lies in or you want an additional signature of another binary or a subdirectory to be used to identify the application this process belongs to, then specify a relative path (starting from the directory the process executable *name* lies in).
- **signatureData**
  Data (Base64 encoded) representing the signature of the binary/directory/bundle .

- **strongKill**
  Boolean indicating whether an application (or process) may be killed in a not-nice way, what may cause data loss if the application had unsaved data in memory or was just writing to a persistent memory/drive. If this application is safe to be killed anytime, then setting this flag to true helps to avoid bothering users: If this flag is set to false and the application is running already when SEB is started, then an alert/dialogue window is displayed to ask the user to quit this permitted application together with other permitted applications and to restart SEB afterwards (or to let SEB kill the applications risking that there could be data loss).
  Setting this flag to false does not mean that processes (and applications) are not terminated: Depending on the platform's capabilities, SEB tries to terminate permitted applications nicely or asking the user to do it themselves.
  This flag should not be set for OS X applications which allow to be terminated nicely (they are anyways automatically terminated, without asking the user).
  Default value: <false/>

Default value: **Important: When SEB Client on Windows loads a .seb settings file, it should not only check if there is a value (array) for the *permittedProcesses* key, but also if there is an entry (array item) for the standard process for Firefox/XULRunner. If not, it has to save this default entry representing *xulrunner.exe* inside the *permittedProcesses* array.** The reason for this is that a .seb file saved by SEB on Mac OS X might contain some permitted processes (so the key/value exists) but there won't be any *xulrunner.exe* process.

Disclaimer: For now XULRunner will be started with standard executable and arguments directly by SEB Starter, even without this default entry.

pinEmbeddedCertificates

Boolean indicating if the certficate store of macOS or the embedded Firefox browser (SEB for Windows) should not be used to evaluate the validity of a server certificate when SEB connects to a secure server using https. You have to embed TLS or CA certificates into SEB settings which establish trust for the secure servers you want SEB to connect to. The used certificates must be valid (not expired, containing the server's host address in "common name" or "alternative names"). If you want to use a certificate which fails validation (for the mentioned reasons), use a "Debug Certificate" which you can add using the debug certificate option (in the "Advanced" window in SEB for macOS), where you can override "common name" and "alternative names" by changing the displayed name of the certificate into the server's host address (you can also specify a non-default port number).

If "Pin embedded certificates" isn't enabled and you embed TLS, Debug and/or CA certificates, these certificates extend the system trust store (as if you had manually added them to the system trust store).

Default value: <false/>

Functionality first available in version

Windows	2.2
macOS	2.1.1
iOS	–

prohibitedProcesses

Array of dictionaries which contain the properties of processes which are prohibited to run during an exam (they are on the blacklist) when ***monitorProcesses*** is true. This blacklist of processes makes sense because SEB on both platforms usually allows to run system processes (SEB MacOSX allows all processes and applications signed by Apple to run), but some of them might not be wanted during an exam. With *prohibitedProcesses* you can prevent some specific background processes and applications from running together with SEB. Use this with care, test if the system continues to run safely when the blacklisted processes are killed by SEB.

Keys in a *prohibitedProcesses* dict:

- **active**

Boolean indicating if the prohibited process is active.

- **allowedExecutables**
  String with a comma separated list of names of the window handling processes of a prohibited application. These are necessary if a Windows application (process) doesn't provide the *mainWindowHandle* property. This is usually the case for Java applications like OpenOffice (see SEB documentation for examples).
  Default value: <string></string> (empty string = no window handling processes)
  Windows only
- **currentUser**
  Boolean indicating that the prohibited process has to run under the currently logged in user (not system users). Use it instead of indicating the user identifier (*user* key).
  Default value: <false/>
- **description**
  String containing a description of the process. This is only displayed in the SEB configuration tool, preferences window and in logs. It should explain what kind of process this is, because this might not be obvious only from the *name*.
- **executable**
  String of the process name (usually the file name of the executable).
- **originalName**
  String containing the original filename meta data of the executable (only available in Windows).
- **identifier**
  String of the process identifier in reverse domain notation (Mac) or the string or substring of the main window title of a process which doesn't have a MainWindow handle (Win), this is usually the case with Java applications.
- **os**
  Integer with a value representing on which operating system the permitted process runs.

  Possible values for the *os* key:

```
enum {
    operatingSystemOSX                  = 0,
    operatingSystemWin                  = 1
};
typedef NSUInteger operatingSystems;
```

- **signatures**
  Array of dictionaries containing metadata and the actual signatures of the process executable. Used to identify the process binary/application securely. Multiple signatures are possible to identify several versions of a binary. The signature algorithm is platform specific and has yet to be defined.

  Keys in the *signatures* array dictionaries:
  - **description**
    String containing a description of the signature. For example it should be mentioned which application/binary version is covered by the signature.
  - **relativePath**
    By default the signature is calculated over the executable file of the process (identified by the key *name*). If you want the signature be calculated using for example the directory the executable lies in or you want an additional signature of another binary or a subdirectory to be used to identify the application this process belongs to, then specify a relative path (starting from the directory the process executable *name* lies in).
  - **signatureData**
    Data (Base64 encoded) representing the signature of the binary/directory/bundle.

- **strongKill**
  Boolean indicating whether an application (or process) may be killed in a not-nice way, what may cause data loss if the application had unsaved data in memory or was just writing to a persistent memory/drive. If this application is safe to be killed anytime, then setting this flag to true helps to avoid bothering users: If this flag is set to false and the application is running when SEB is started, then an alert/dialogue window is displayed to ask the user to quit this prohibited application together with other not permitted applications and to restart SEB afterwards (or to let SEB kill the applications risking that there could be data loss).
  Setting this flag to false does not mean that processes (and applications) are not killed: Depending on the platform's capabilities, SEB first tries to terminate prohibited processes and applications nicely or asking the user to do it themselves, if this doesn't work then it kills them strongly anyways (as long as ***monitorProcesses*** is set to true). But setting *strongKill* will speed up this process.
  This flag should not be set for OS X applications which allow to be terminated nicely (they are anyways automatically terminated, without asking the user).
  Default value: <false/>
- **user**
  String with the user identifier under which this process is running. If no user is indicated, then the process is killed regardless under which user it's running. Instead of this identifier also the key *currentUser* can be used.

proxies
Dictionary containing key/values of proxy settings for the exam client computers which override the system's proxy settings on the clients if *proxy SettingsPolicy* is set to *useSEBProxySettings*.
Note: The key names in the proxies dictionary are taken from the proxies dictionary in the OS X network service dictionary. That's why they follow a different notation (first letter is capital).

Keys in the *proxies* dictionary:

- **ExceptionsList**
  Array of strings containing host and domain names of network resources that should be accessed without a proxy server.
- **ExcludeSimpleHostnames**
  Boolean indicating whether simple host names are excluded.
  Default value: <false/>
- **AutoDiscoveryEnabled**
  Boolean indicating if automatic proxy discovery is used.
  Default value: <false/>
- **AutoConfigurationEnabled**
  Boolean indicating if automatic proxy configuration is used.
  Default value: <false/>
- **AutoConfigurationJavaScript**
  String containing the full JavaScript source for the proxy autoconfiguration (PAC) file.
- **AutoConfigurationURL**
  String specifying the location of the proxy autoconfiguration (PAC) file.
- **FTPPassive**
  Boolean indicating if passive FTP mode (PASV) is used.
  Default value: <true/>
- **FTPEnable**
  Boolean indicating if the FTP proxy is enabled.
  Default value: <false/>
- **FTPPort**
  Integer representing the port number (between 1 and 65535) of the FTP proxy.
  Default value: <integer>21</integer>
- **FTPProxy**
  String containing either the hostname or IP number of the FTP proxy host.
- **FTPRequiresPassword**
  Boolean indicating if a password (and username) is required when contacting the proxy.
- **FTPUsername**
  String containing the username to be used when contacting the proxy.
- **FTPPassword**
  String containing the plain password for this proxy server.
- **HTTPEnable**
  Boolean indicating if the FTP proxy is enabled.
  Default value: <false/>
- **HTTPPort**
  Integer representing the port number (between 1 and 65535) of the FTP proxy.
  Default value: <integer>80</integer>
- **HTTPProxy**
  String containing either the hostname or IP number of the FTP proxy host.
- **HTTPRequiresPassword**
  Boolean indicating if a password (and username) is required when contacting the proxy.
- **HTTPUsername**
  String containing the username to be used when contacting the proxy.
- **HTTPPassword**
  String containing the plain password for this proxy server.
- **HTTPSEnable**
  Boolean indicating if the HTTPS proxy is enabled.
  Default value: <false/>
- **HTTPSPort**
  Integer representing the port number (between 1 and 65535) of the HTTPS proxy.
  Default value: <integer>443</integer>
- **HTTPSProxy**
  String containing either the hostname or IP number of the HTTPS proxy host.
- **HTTPSRequiresPassword**
  Boolean indicating if a password (and username) is required when contacting the proxy.
- **HTTPSUsername**
  String containing the username to be used when contacting the proxy.
- **HTTPSPassword**
  String containing the plain password for this proxy server.
- **RTSPEnable**
  Boolean indicating if the RTSP proxy is enabled.
  Default value: <false/>
- **RTSPPort**
  Integer representing the port number (between 1 and 65535) of the RTSP proxy.
  Default value: <integer>554</integer>
- **RTSPProxy**
  String containing either the hostname or IP number of the RTSP proxy host.

- **RTSPRequiresPassword**
  Boolean indicating if a password (and username) is required when contacting the proxy.
- **RTSPUsername**
  String containing the username to be used when contacting the proxy.
- **RTSPPassword**
  String containing the plain password for this proxy server.
- **SOCKSEnable**
  Boolean indicating if the SOCKS proxy is enabled.
  Default value: <false/>
- **SOCKSPort**
  Integer representing the port number (between 1 and 65535) of the SOCKS proxy.
  Default value: <integer>1080</integer>
- **SOCKSProxy**
  String containing either the hostname or IP number of the SOCKS proxy host.
- **SOCKSRequiresPassword**
  Boolean indicating if a password (and username) is required when contacting the proxy.
- **SOCKSUsername**
  String containing the username to be used when contacting the proxy.
- **SOCKSPassword**
  String containing the plain password for this proxy server.

proxySettingsPolicy
Integer with a value representing one of the *proxySettingsPolicies*

Possible values:

```
enum {
    useSystemProxySettings          = 0,
    useSEBProxySettings             = 1
};
typedef NSUInteger proxySettingsPolicies;
```

Default value: <integer>0</integer> (*useSystemProxySettings*)

quitURL
String containing the full URL (starting with http:// or https://) of the link to quit SEB after exam.

Default value: <string></string> (empty string = quit link feature not active)

quitURLConfirm
Boolean indicating if the user is asked to confirm quitting SEB after the quit URL has been detected by SEB. If set to *false*, SEB quits without displaying any dialog.

Default value: <true/>

removeBrowserProfile
Boolean indicating if the XULRunner browser profile should be removed when quitting SEB for Windows. This deletes various browser and session information, as caches and also local storage.

Default value: <false/>
Windows only

removeLocalStorage
Boolean indicating if the browser's local storage database should be disabled. Local storage (which is stored per origin, e.g. the combination of protocol, hostname, and port number as defined in the same origin policy) will persist after the browser is closed, its contents will still be available after restarting SEB. This might be a security issue though if you don't know how securely an exam system is handling information in localStorage. Many modern web applications rely on local storage, so be careful if disabling it.

Default value: <false/>
Mac only (use removeBrowserProfile in SEB for Windows)

restartExamPasswordProtected
Boolean indicating if The quit/restart password (if set, see General pane) must be entered when the restart exam button was tapped. Exam support/invigilators should be told this password to be able to restart the exam if there is a problem.

Default value: <true/>

restartExamText
This text is displayed as the title of the confirmation alert and as tool tip on the icon. Leave empty for a standard text (which is localized to the SEB user interface languages).

restartExamURL
Either check the "Use Start URL" option or enter a link to which the exam is redirected when the Back to Start (formerly restart exam) Button is pressed. The Back to Start Button is displayed in the SEB task bar when either the "Use Start URL" option is selected or a link is entered.

restartExamUseStartURL
Boolean indicating if the Start URL should be used when the Back to Start Button is pressed.

Default value: <false/>

sebBrowser
String containing the executable's file name of the SEB browser application, which is specified with an entry in the *permittedProcesses* dictionary (*executable* key). To be started automatically, it also needs to be added to the array *autostartApplications*. If you don't want to use any browser in SEB, because SEB Starter should only act as a kiosk application starting up another application in a kiosk mode (for example a virtual desktop infrastructure client), then you need to set *enableSebBrowser = false*. Default value: <string>xulrunner.exe</string>
Windows only

sebConfigPurpose
Integer with a value representing one of the *sebConfigPurposes* which indicate for what the SEB settings file will be used. This option is significant only for the preferences window and config tool.

Possible values:

```
enum {
    sebConfigPurposeStartingExam        = 0,
    sebConfigPurposeConfiguringClient   = 1
};
typedef NSUInteger sebConfigPurposes;
```

Default value: <integer>0</integer> (*sebConfigPurposeStartingExam*)

sebMode
Integer with a value representing one of the *sebModes* which indicate if SEB will use local settings and load the start URL or if it will try to connect to the SEB Server.

Possible values:

```
enum {
    sebModeStartURL             = 0,
    sebModeSebServer            = 1
};
typedef NSUInteger sebModes;
```

Default value: <integer>0</integer> (*sebModeStartURL*)

sebServerFallback
Boolean indicating if SEB should connect to the start URL and use local settings in case it cannot connect to the SEB Server.

Default value: <false/>

sebServerURL
String containing the full URL (starting with http:// or https://) of a SEB Server. See key *sebMode*.

Default value: <string></string>

sebServicePolicy
(replaces key ForceWindowsService on Windows)
Integer with a value representing one of the *sebServicePolicies* which indicate whether SEB Starter is allowed to run without the SEB Service (background process), if a warning is displayed when the service is not running or if SEB Starter is only allowed to run when the service is running.

Possible values:

```
enum {
    ignoreService                   = 0,
    indicateMissingService          = 1,
    forceSebService                 = 2
};
typedef NSUInteger sebServicePolicies;
```

Default value: <integer>2</integer> (*forceSebService*)

sendBrowserExamKey
Boolean indicating if the *Browser Exam Key* should be send in a custom HTTP request header, combined with the URL of the HTTP request into a SHA256 hash. The exam key is generated dynamically using the application's code signature, a special check sum over all relevant settings of the SEB client and a random salt, saved in key *examKeySalt* when the settings have been created.

Default value: <false/>

showMenuBar
Boolean indicating if the Mac OS X menu bar including all menus should be displayed or not. The menu bar and the menus are not at all required for using SEB and might distract examines, but from SEB MacOSX version 2.x this option allows to use menus in SEB similar to standard Mac OS X applications.

Default value: <false/>
Mac only

showReloadWarning
Boolean indicating if a warning should be displayed before reloading the web page in the main browser window. For additional browser windows see key *newBrowserWindowShowReloadWarning*.

Default value: <true/>

showTaskBar

Boolean indicating if the SEB dock/task bar should be displayed.

Default value: <true/>

startURL
String containing the full URL (starting with http:// or https://) of the page to open when SEB is started. This can be for example the URL of an exam or of a exam portal page.

Default value: <string>http://www.safexambrowser.org</string>

taskBarHeight
Integer indicating the height in pixels of the SEB dock/task bar if *showTaskBar* = *true*.

Default value: <integer>40</integer>

touchOptimized
Boolean indicating touch optimized appearance.

Default value: <false/>
Windows only

URLFilterEnable
Boolean indicating if URLs are filtered using the *URLFilterRules* dictionary.

Default value: <false/>

URLFilterEnableContentFilter
Boolean indicating if not only URLs are filtered using the URLFilterRules dictionary, but also all embedded resources. If set to true, the filter patterns will be applied for all embedded resources (js, css, images...).

Default value: <false/>

URLFilterRules
Array of dictionaries each containing a set of URL filter rules. URL filtering is enabled/disabled with the keys *URLFilterEnable* and *URLFilterEnableContentFilter*.

Keys in the *URLFilterRules* array dictionaries:

- **action**
  Integer with a value representing one of the *URLFilterRuleActions*. The URL filter processes first the expressions of block actions and then one by one. If the URL doesn't match one expression, then the next expression is processed. The actions described below are used to decide what to do if the URL matches the expression.

  Possible actions:
  - **block**
    If the URL matches the expression, then it is rejected and processing of following actions inside this rule and processing of rules following the current one is stopped.
  - **allow**
    If the URL matches the expression, then it is accepted and processing of following actions inside this rule and processing of rules following the current one is stopped.

If the URL filter reaches the end of the last allow rule (means there was no matching *block* or *allow* expression found), then the URL is discarded. If you want the URL to be accepted if no matching *block* (or *allow*) expression was found, then add an <*allow* *> action/expression.

Possible values for the *action* key (in the user interface only the actual action name is shown, see above):

```
enum {
   URLFilterActionBlock                = 0,
   URLFilterActionAllow                = 1
};
typedef NSUInteger URLFilterRuleActions;
```

- **active**
  Boolean indicating if the action is active.
- **expression**
  String containing the filtering expression or pattern, either in the regular expression format (*regex* = *true*) or (*regex* = *false*) a simpler filter expression containing the wildcard char <*>. A filter expression can filter against all elements of a URL/URI according to RFC 3986:

  scheme://user:password@host:port/path?query_string#fragment_id

  Format for a filter expression in the non-regex format:
  - Scheme is optional, and must be followed by '://'.
  - The host field is required (besides when filtering against a protocol like about:blank or data:), and is either a partial or full hostname or an IP address. It can also contain or be replaced completely with the wildcard '*' char, see below for details. The URL filter doesn't resolve hostnames itself, so if you allow everything and only block 'hostname.com', then that host could still be reached using its IP address. You should therefore mainly use whitelisting to allow accessing only specific sites during an exam.
  - An optional '.' (dot) can prefix the host field to disable subdomain matching, see below for details.
  - An optional port can come after the host and always has to start with the character ':'. It must be a valid port value from 1 to 65535.
  - An optional path can come after the host or after the port and always has to start with the character '/'. Parts of the path can be replaced with the wildcard char '*'.
  - URL parameters like a query string can be indicated and always have to start with the character '?'. Parts of the query can be replaced with the wildcard char '*'.
  - Filtering against a fragment usually doesn't make sense, as the content can be reached by scrolling the loaded page. Allowing specific fragments could force people to use a link to a particular anchor on a page (other links would not work).

  - Examples for filter expressions:
  - <example.com> matches <example.com>, <www.example.com> and <www.mail.example.com> (internally processed as a host name search for example.com and a search for *.example.com)
  - <.www.example.com> matches exactly <*://www.example.com> (no other subdomains)
  - <mail.*> matches all hosts having a subdomain or domain 'mail', like <mail.ethz.ch>, <www.mail.gov.to>, <mail.com>
  - <*:8088> matches all requests to port 8088
  - <example.com/stuff/*> matches all requests to any subdomain of <example.com> that have <stuff> as the first segment of the path
  - <example.com/images/*.png> matches all requests to any subdomain of <example.com> that have </images/> as the first segment of the path and <.png> as the path extension of an file URL (means it matches all PNG images in the </images> directory or its subdirectories)
  - <*.net> matches all host with any kind of subdomains in the .net top level domain like <example.net>, <www.example.net>, <www.mail.example.net>
  - <*/*.net> matches all files with a <.net> file extension on any host

  You should consider using a regular expression if performing complex filtering (when a simple filter doesn't cover all possible cases how that complex URL could be formatted) or split the expression into several filter rules. The order of filter rules is not relevant.

- **regex**
  Boolean indicating if the action rule is a regular expression. If *regex* is set to *false*, then the rule is formatted using the wildcard * (stands for an arbitrary string of any length).

urlFilterRegex
Boolean indicating whether all the white- and blacklist URL filters are regular expressions. Only used in the Windows version communicating with the internal XULRunner browser (is set according to *URLFilterRules*, doesn't need to be set in .seb settings).

Default value: <false/>
Windows only

urlFilterTrustedContent
Boolean indicating if not only URLs are filtered using the URLFilterRules dictionary, but also all embedded resources. If set to true, the filter patterns will be applied for all embedded resources (js, css, images…). Only used in the Windows version communicating with the internal XULRunner browser (is set equal to *URLFilterEnableContentFilter* despite the misleading key name, doesn't need to be set in .seb settings).

Default value: <false/>

useAsymmetricOnlyEncryption
Boolean indicating if the old asymmetric-only encryption for config files encrypted with a X.509 identity certificate should be used (compatible with older SEB versions).

Default value: <false/>

First available in version

Windows	2.2
macOS	2.2
iOS	2.2

blacklistURLFilter
String containing semicolon separated URL blacklist filter pattern. The blacklist is a list of semicolon separated URL filter patterns for disallowed resources and will be first executed. The whitelist is a list of semicolon separated URL filter patterns for explicitly allowed resources. All other resources will be denied. Only used in the Windows version communicating with the internal XULRunner browser (is set according to *URLFilterRules*, doesn't need to be set in .seb settings).

Default value: <string></string> (empty string = no URL filtering)
Windows only

whitelistURLFilter
String containing semicolon separated URL whitelist filter pattern. The blacklist is a list of semicolon separated URL filter patterns for disallowed resources and will be first executed. The whitelist is a list of semicolon separated URL filter patterns for explicitly allowed resources. All other resources will be denied. Only used in the Windows version communicating with the internal XULRunner browser (is set according to *URLFilterRules*, doesn't need to be set in .seb settings).

Default value: <string></string> (empty string = no explicitly allowed resources)
Windows only

zoomMode
Integer with a value representing one of the *SEBZoomModes.* Zoom whole web pages or just text using Ctrl-Mousewheel (only in Windows version). In the macOS version both zoom modes can be used at the same time.

Possible values:

```
enum {
   SEBZoomModePage                  = 0,
   SEBZoomModeText                  = 1
};

typedef NSUInteger SEBZoomModes;
```

Default value: <integer>0</integer> (*SEBZoomModePage*)
Windows only