Browser Profile Configuration - Quick Start Guide

Version 1.1 – 24 August 2020

Wavelink is now ivanti

Contents

Installing the Velocity Client 3
To install the Velocity Client on Android 3
To install the Velocity Client on Windows 10 3
Installation of Velocity Console 4
To install or upgrade the Velocity Console 4
Configuring Host Profiles 4
Clear cache / clear cookies 6
Screen Rotation 7
HTTPS connection/Certificates 8
Scanner Configuration 9
Scan Terminator 11
Scan Handlers 12
Full Screen Mode 17
Hide menu/Hide Toolbar 18
Immersive Mode 19
Distributing Settings to Devices 20
Deploying a project from the Console 20
To deploy a project from the Velocity Console to a local Windows Client 21
Web settings – changing application behaviour 22
Web – Insert script 22
Insert CSS or javascript as an HTML insert 24
Set the display with, disable zoom – viewport 25
Resizing, scaling of the application (SAP Logon) 26
Resizing SAP ITS Mobile 27
SAP ITS Mobile Error sound 29
SAP ITS Mobile Function Keys 32
Automatically end session (after SAP logoff) 33
Delete Demo project from main menu 35
Appendix A. Send data to Velocity using intents 41

Ivanti Velocity is an Android or Windows client that can connect to Telnet hosts (including IBM 5250/3270 and VT100/220), web apps, and Oracle SIM hosts. The Velocity Console is an application installed on your desktop or laptop and is used to configure the Velocity Client. Create host profiles for different host types, import screens and edit how they appear, and set up speech-to-text or text-to-speech using Ivanti Speakeasy. This guide will explain how to create and deploy a settings file or Host Profile in Velocity that contains the initial configuration of a telnet connection.

Both the client and Console application can be download from the Velocity Downloads page: http://www.wavelink.com/Download-Velocity_enterprise-app-modernization-Software/

Installing the Velocity Client

To install the Velocity Client on Android

1. If the device has the Google Play Store installed, you must make sure the device allows installation of non-Play Store applications. This option is frequently named Unknown sources under Settings > Security.
2. Start the installation with one these methods:
    a. Copy the installation apk file to the device. Open a file browser application and locate the folder when the file is copied. Tap the file name to start the installation.
    b. Using the device, open a browser and navigate to the Velocity downloads page on the Ivanti website. Download the latest Client appropriate for the device type.

       View the device Notifications by pulling down the status bar. When the Client download is complete, tap the notification to install the Client.
3. The installation screen appears. Tap Install.
4. The app is installed. Tap Open to launch the Velocity Client or tap Done.

To install the Velocity Client on Windows 10

1. Using the device, open a browser and navigate to the Velocity downloads page on the Ivanti website. Download the latest Client appropriate for the device type.
2. Run the installer. You may be prompted to install .NET. If you are prompted to install .NET, you must reboot before you can complete the installation.

Installation of Velocity Console

The Velocity Console can be installed on the following operating systems:

- Windows 7 64-bit
- Windows 10 64-bit

To upgrade to a new version of the Velocity Console, download the installer for the new version. When you run the new installer, it will replace the older files with the new version.4

To install or upgrade the Velocity Console

1. Download the Console installation file from the Velocity Downloads page. After you've downloaded it, launch the Velocity.msi file from an administrator account on your computer to begin the install process.
2. Click Next to begin the setup wizard.
3. Accept the terms in the license agreement and click Next.
4. (Optional) If you plan to use Velocity with Oracle SIM, select Include Support for Oracle SIM.
5. Click Install to begin the installation process.
6. Accept the terms of the Microsoft Visual C++ Redistributable and click Install. When the redistributable is finished installing, click Close.
7. Click Finish.

The application is installed on your computer. To use the Velocity Console, either leave the Launch Velocity Console option selected on the last screen of the installer or locate and launch the shortcut in the Start menu.

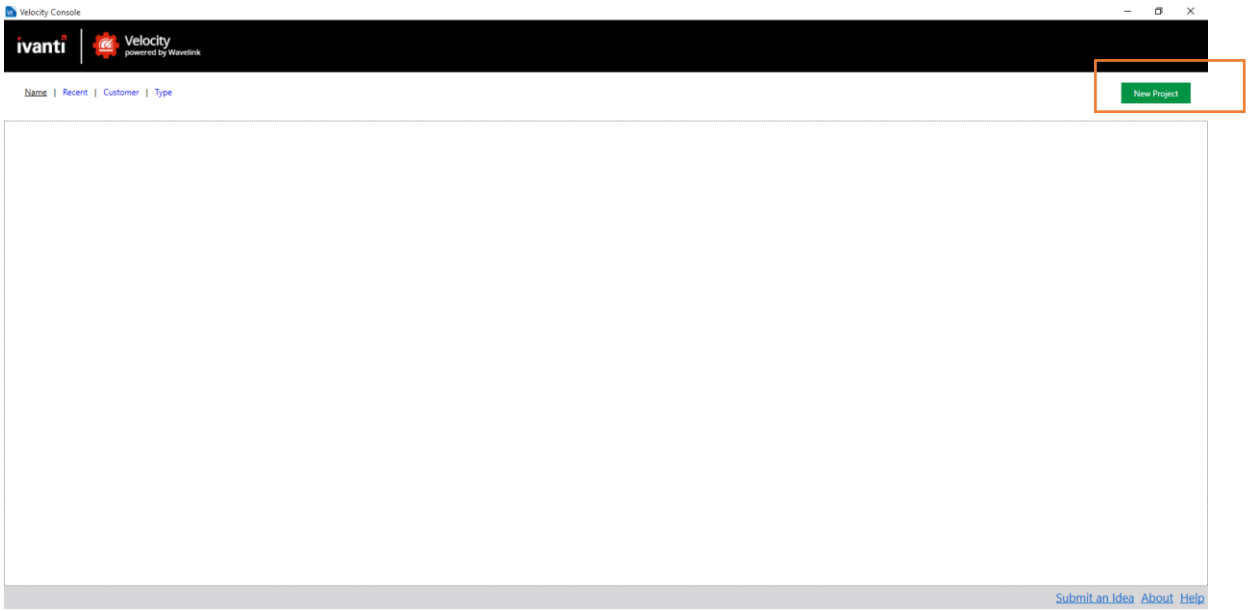
Configuring Host Profiles

A host profile defines the settings that the Velocity Client should use when it attempts to initiate a connection with a specific host. The host profile may include the emulation type, IP address of the host, or other settings. Each project should have one host profile associated with it. You cannot have more than one host profile associated with a project.

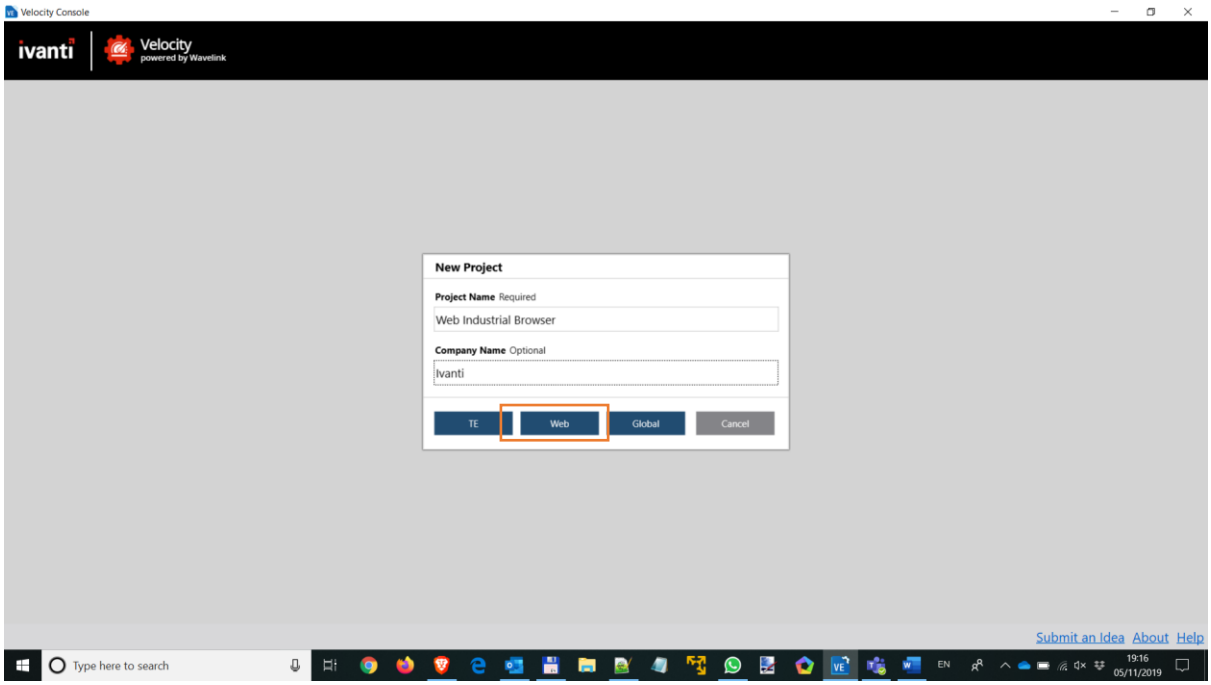
If you want similar settings for more than one host profile: export the project as a zip file, create a new project, import the zip, and change the settings that are different.

When a device user attempts to initiate a session with a host, the Velocity Client displays a list of available host profiles. The user selects the host he wants to connect to, and the Velocity Client uses the host profile settings to connect to the host. There are more settings that can be done via the Velocity Console than can be done directly in the client. Also, a configuration made via the Console can be easily deployed across multiple devices. It is therefore recommended to always create a host profile via the Velocity console.

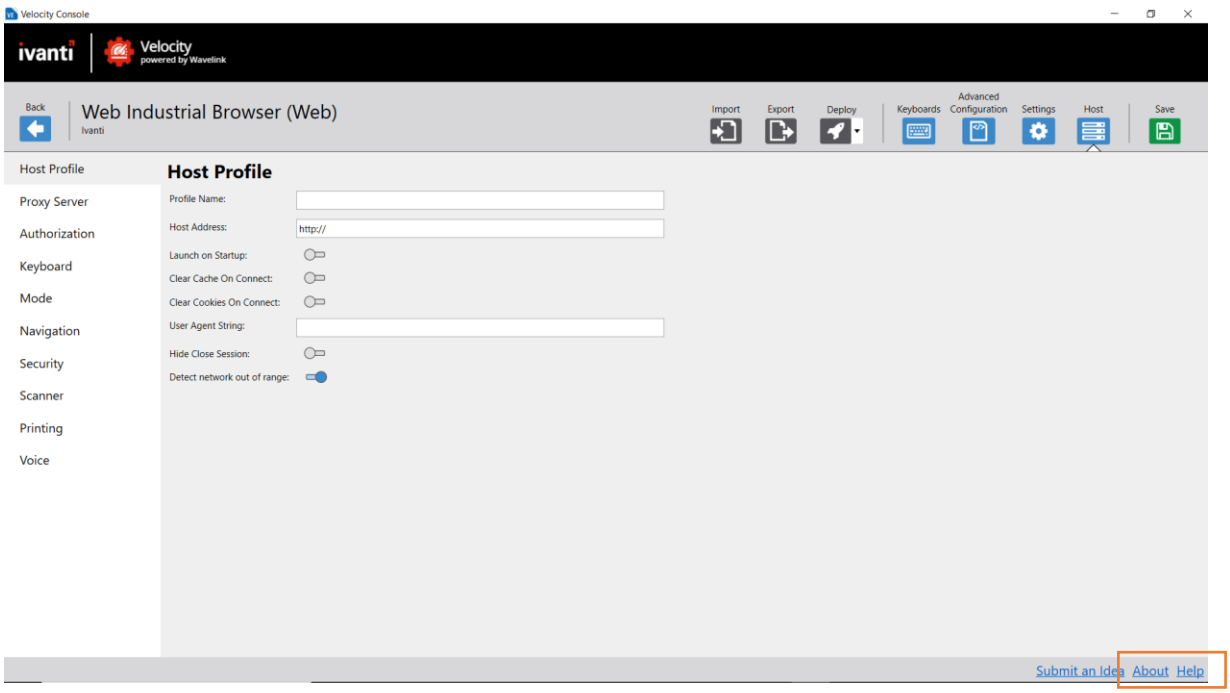
All available projects/configurations are available for the main window of the Velocity Console. Initially, this list is blank. Click New Project to create a new configuration:



The Project Name is required. This is the name that will appear in the main window of the Console. The Company Name is optional. Click the Web button to create a new browser project:

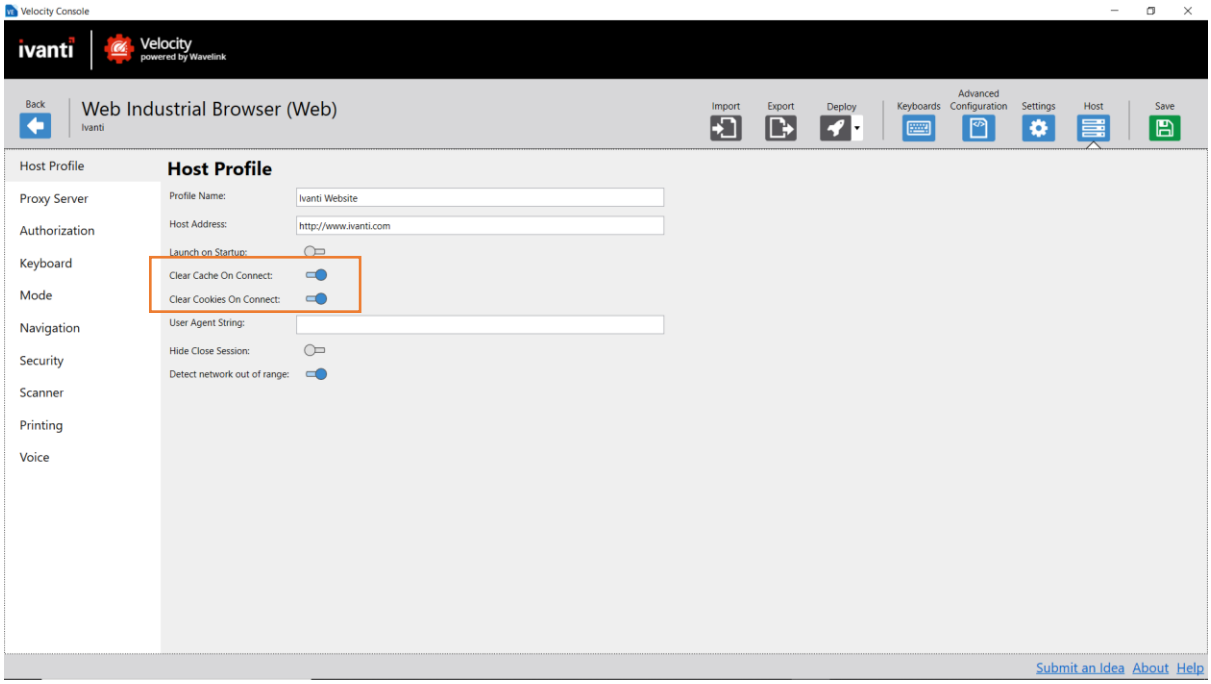


In the Host Profile tab all settings are made to connect to the web host. From the Console, online help is available by clicking Help at the bottom right.



Clear cache / clear cookies

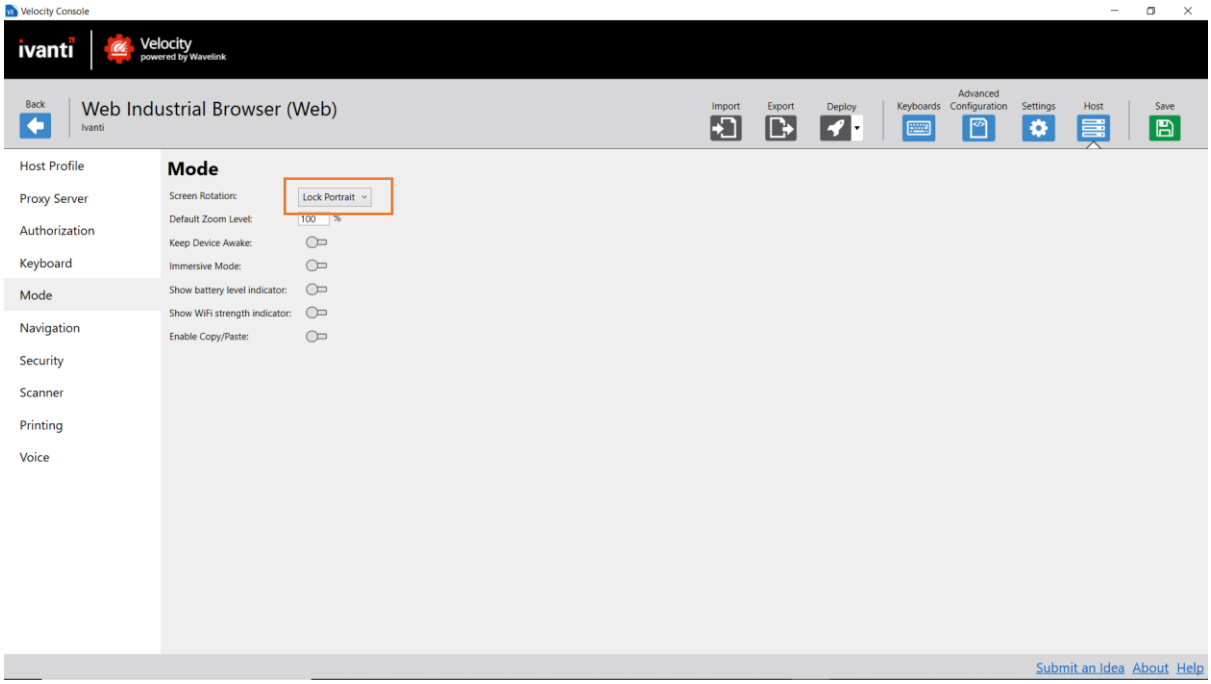
It is usually good practice to clean up all temporary data of a web session. These options clear all cached session objects and cookie data stored for all sessions when a new session is started. Be advised that the data for all sessions is stored in application memory of the Velocity client, not separately stored for every session. When these options are selected in a single session all cached data and cookie data stored will be deleted for all sessions—including active sessions—when a new session is started. Even if another host profile is set to not clear cookies, any profiles with this option active will override.



Screen Rotation

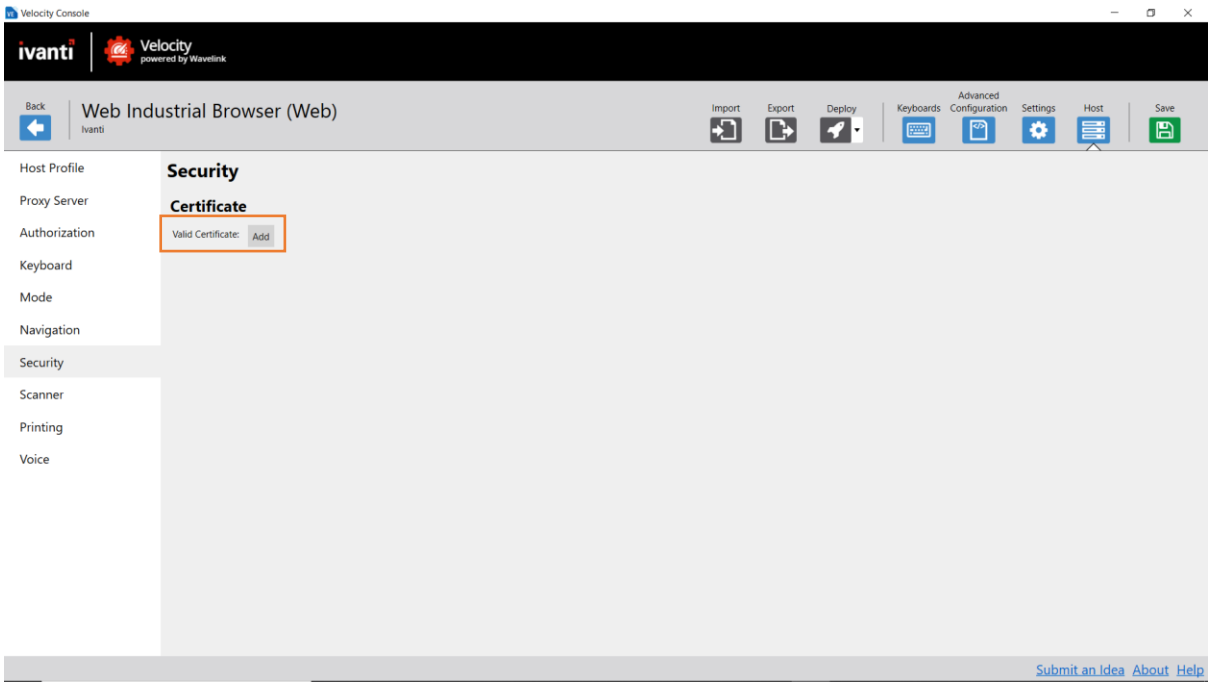
If not configured otherwise Android will rotate the display when the device is rotated. Velocity allows to keep the display orientation fixed, independent of the rotation of the device. Possible values include:

- Lock Portrait
- Lock Landscape
- Lock Reverse Portrait: Use to invert a portrait screen orientation and lock it in place. Tool bars and keyboards also invert. For use with TE and Web projects.
- Lock Reverse Landscape: Use to invert a landscape screen orientation and lock it in place. Tool bars and keyboards also invert. For use with TE and Web projects.
- Dynamic



HTTPS connection/Certificates

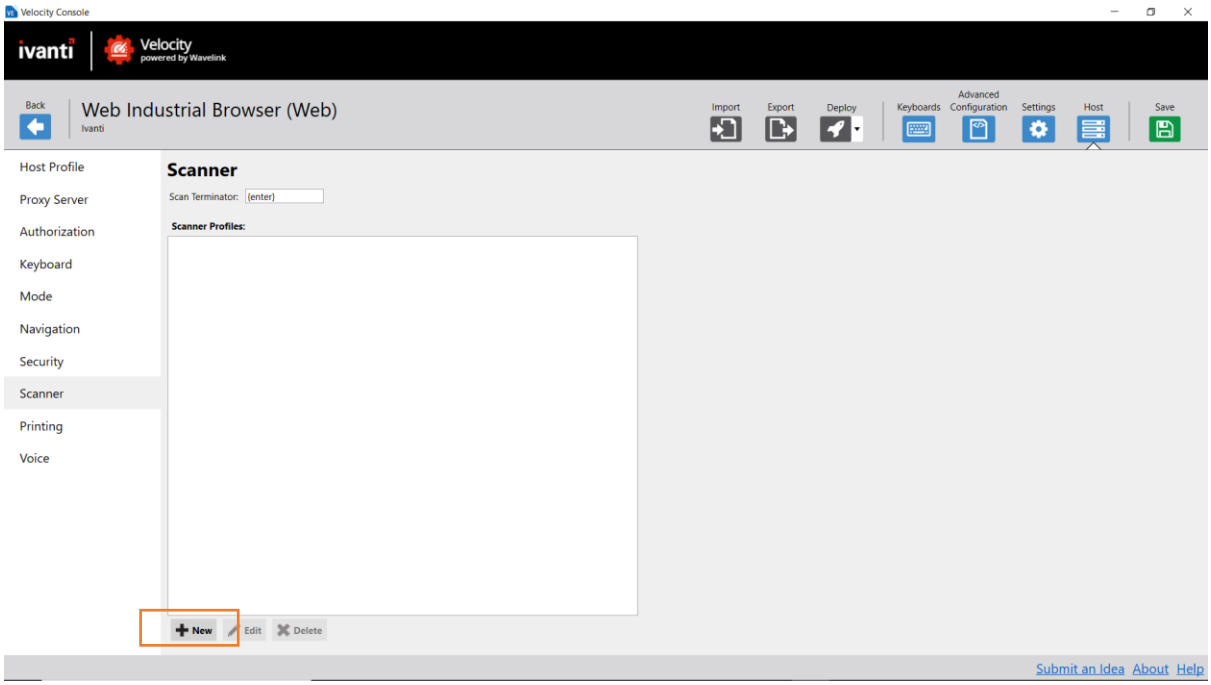
A secured connection to a web server (https) requires a certificate for this secured connection. If the certificate used is not supplied by a trusted authority – like any self-signed certificate – it can be added to the configuration of Velocity. When added, all associated subject and thumbprint information is listed here. The host address shown (either IP address of FQDN) in the certificate must match the host server location, or else the device will not trust the host. The certificate extensions supported includes .cer, .crt, and .pem.



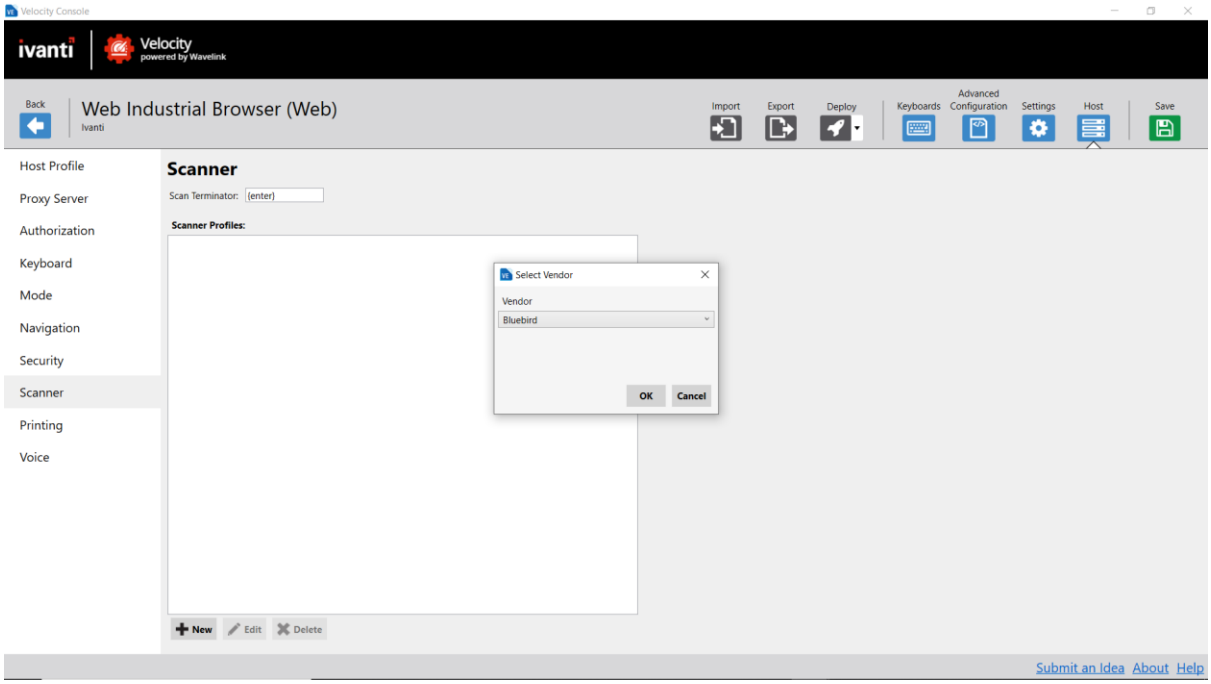
Scanner Configuration

Velocity allows for configuration of certain scanners and imagers that are built in a mobile device. Although the list of supported makes and models is ever growing not every hardware vendor is supported. ***Also, Velocity cannot configure any scanners that are connected via Bluetooth.*** In the case where the scanner is not support by Velocity the manufacturer's wedge software needs to be configured like is described in Appendix A *Send data to Velocity using intents*.

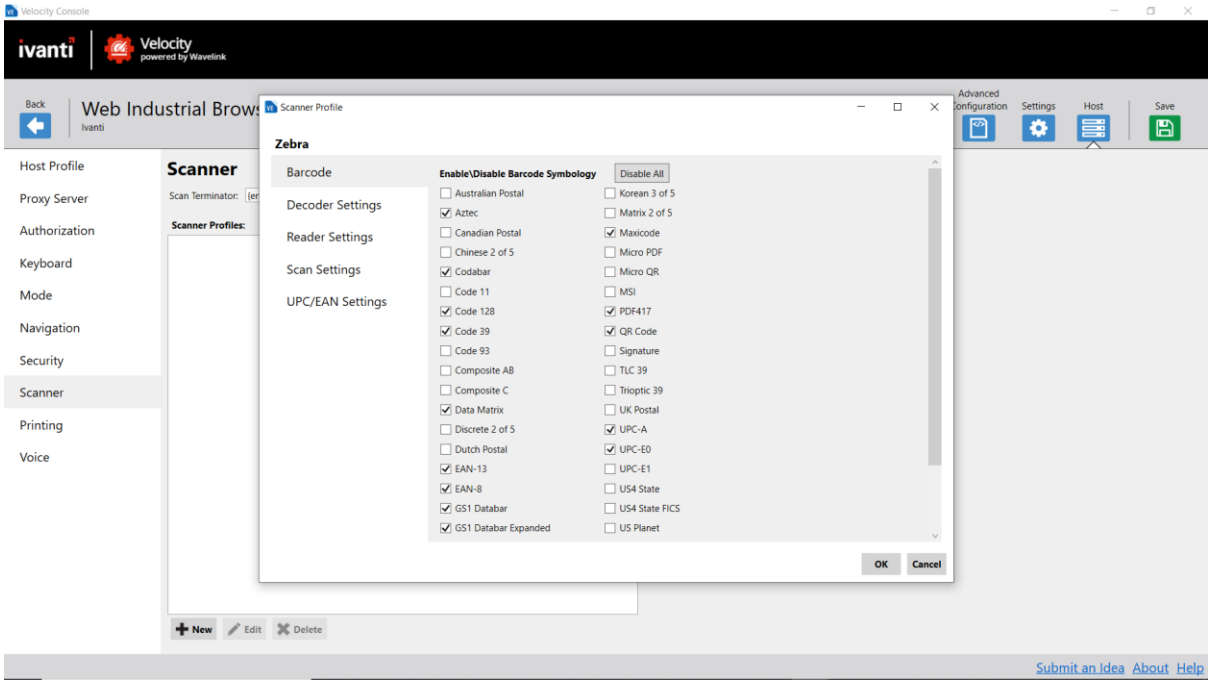
To configure the scanner of a supported model, click 'New' in the Scanner Section:



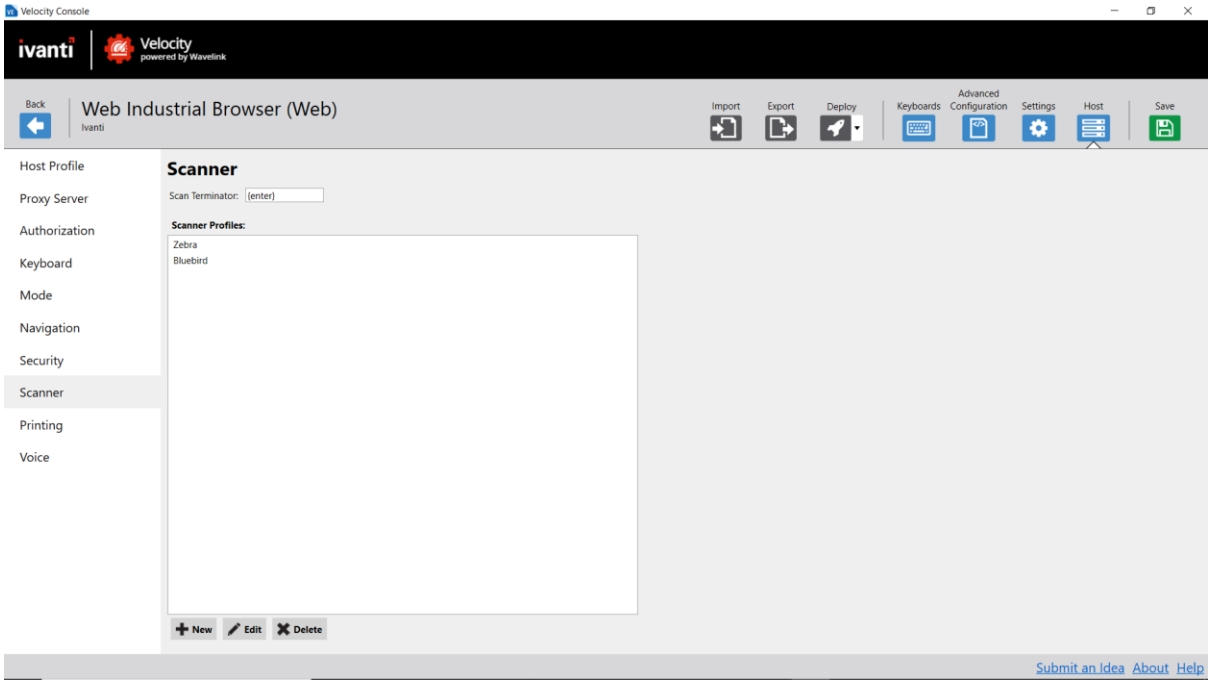
Select the manufacturer of the device from the list:



Then, configure the available parameters where needed:

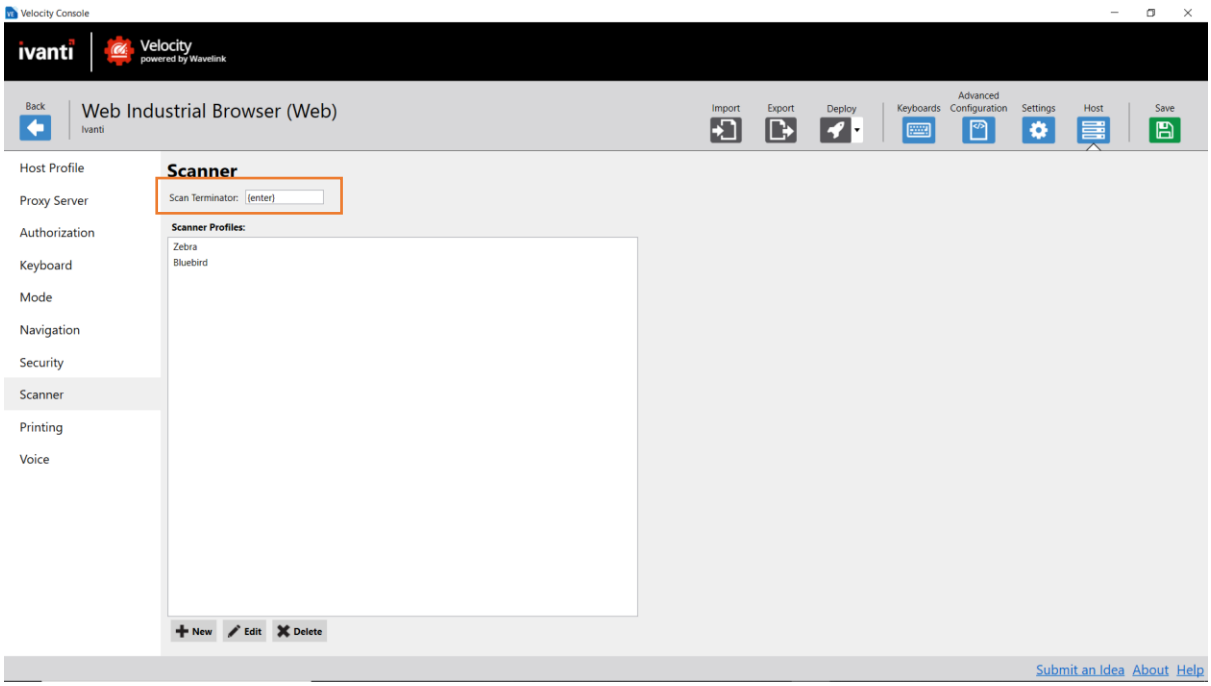


Additional Scan Data Formatting and Scan Handlers can be done via Advanced Configuration, as is described in the next paragraph. A Velocity project can contain multiple scanner configurations for different manufacturers. The Velocity client will use the appropriate scanner configuration at run time.



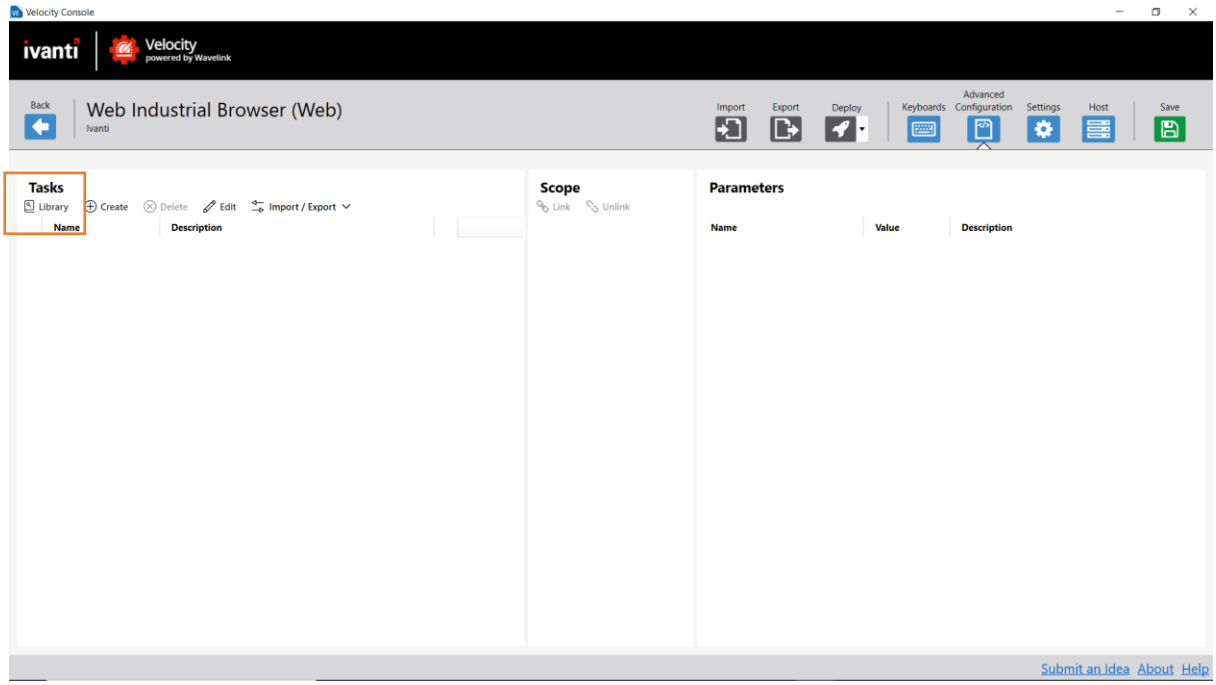
Scan Terminator

When a barcode is scanned it is usually necessary to have the data followed by an enter. By default, this scan terminator is set to the tab character. To change it to enter, change the value for Scan Terminator to '{enter}'.

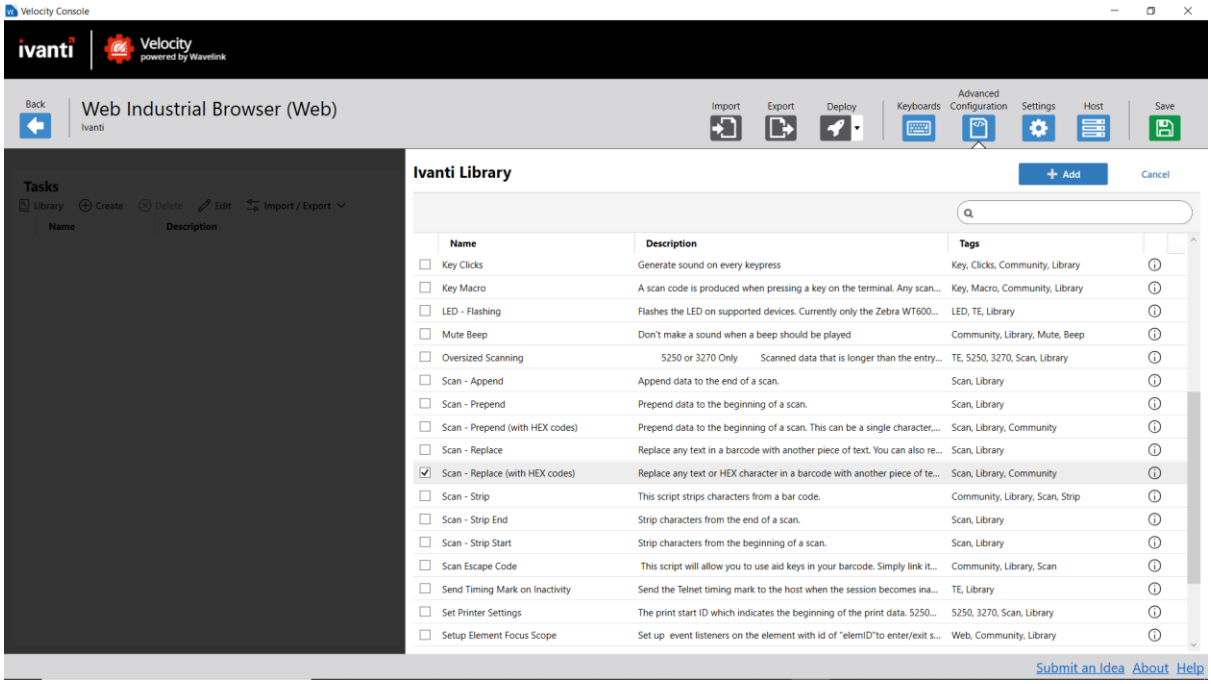


Scan Handlers

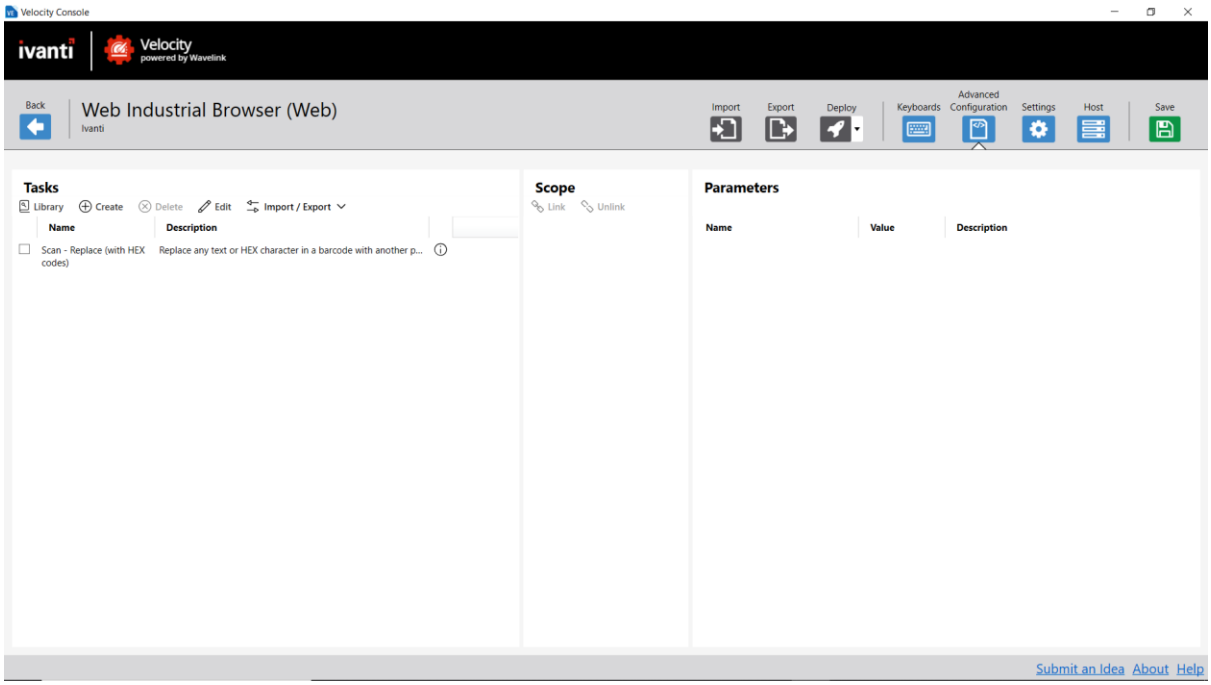
Addition scan handling can be done in the Advanced Configuration section. Advanced scan handler options are available as options from the library where you can pick and choose the options as needed:



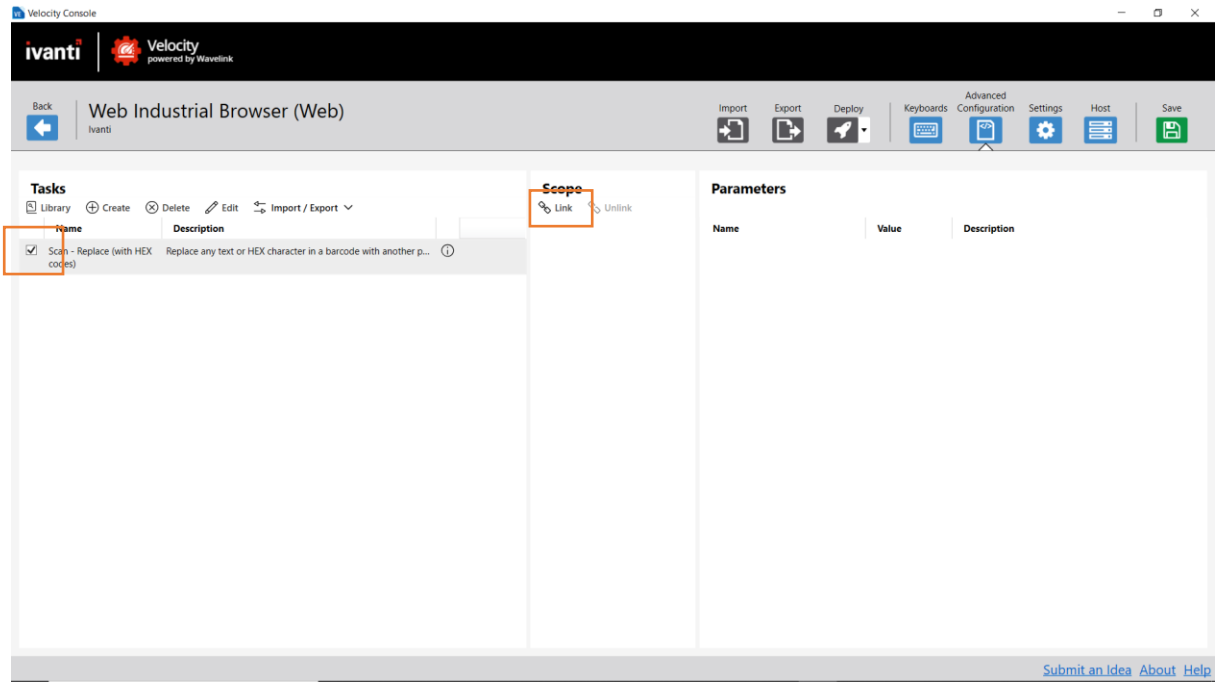
Multiple scan handler options are available to modify the scanned data prepend, append, replace or and delete characters. For example, to replace the FNC1 character in a GS-1/EAN128 barcode select ‘Scan - Replace (with HEX codes)’ from the list and press Add:

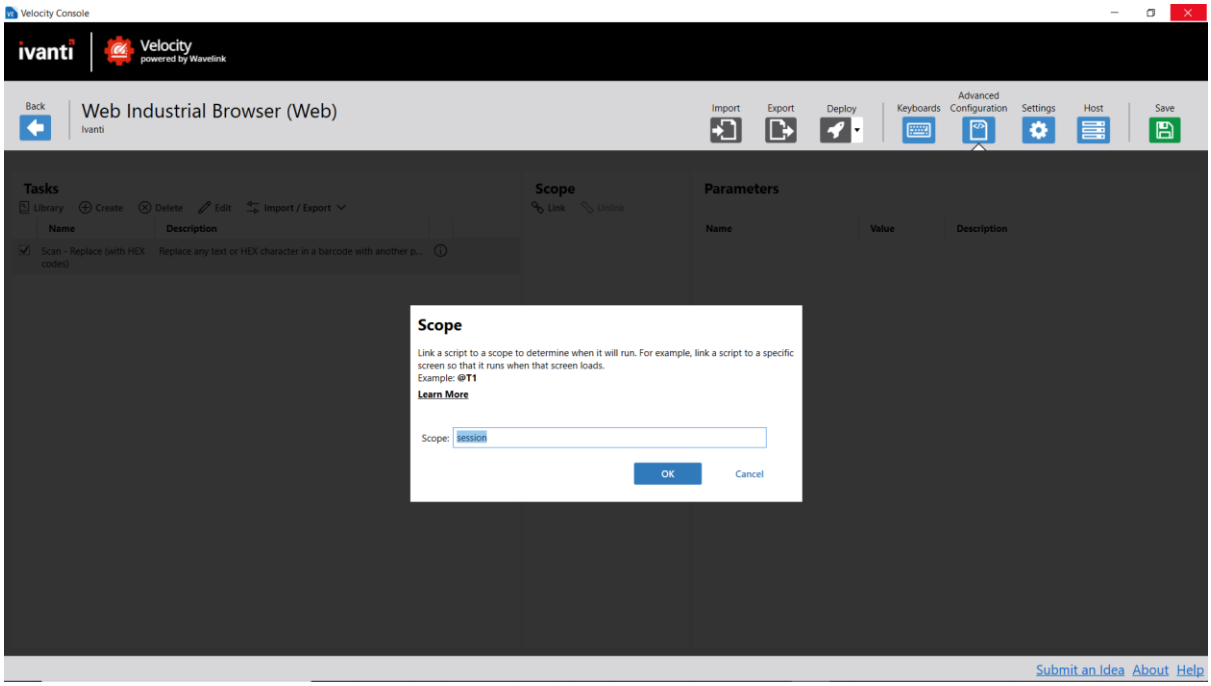


The option now becomes available in the Tasks list:

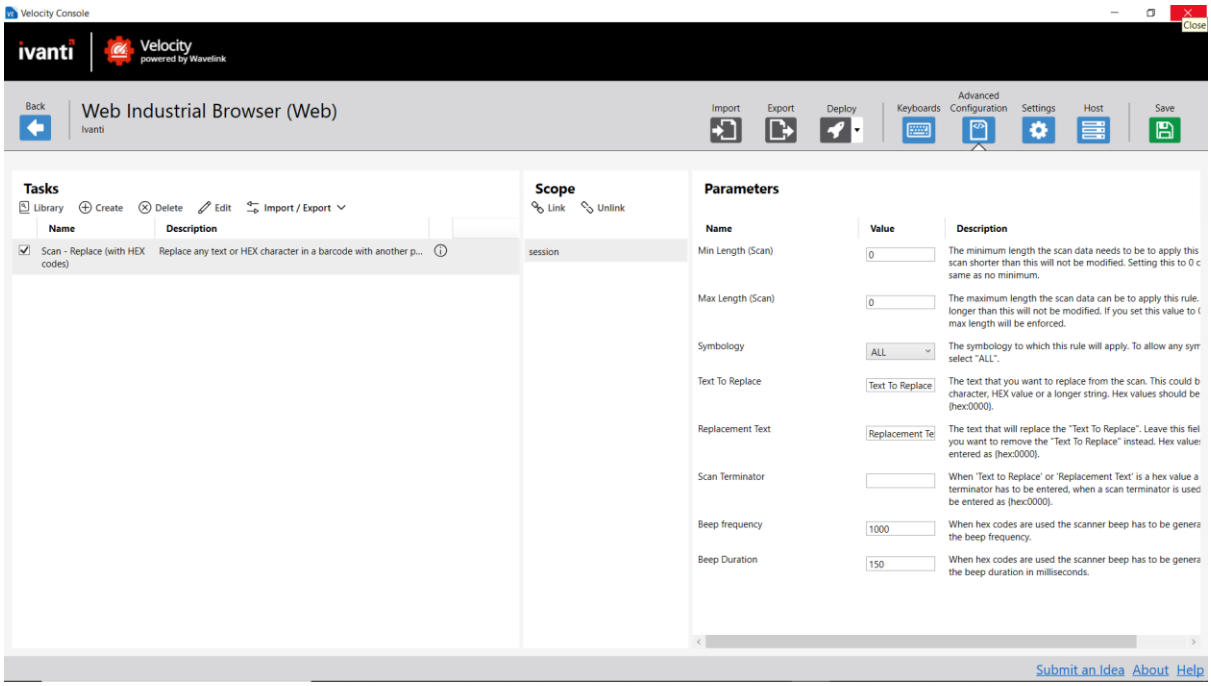


Only adding this option to the configuration does make the setting active. That means that the functionality will not work yet. Also, it is not possible to set any parameters or variables. To activate a script it has to be linked to the session scope. Select the script, click ‘Link’ and click ‘OK’:


Velocity Console
ivanti
Velocity
powered by Wavelink
Back
Web Industrial Browser (Web)
Ivanti
Import
Export
Deploy
Keyboards
Advanced Configuration
Settings
Host
Save
Tasks
Library
Create
Delete
Edit
Import / Export
Name
Description
Scan - Replace (with HEX codes)
Replace any text or HEX character in a barcode with another p...
Scope
Link
Unlink
Parameters
Name
Value
Description
Submit an Idea
About
Help


Velocity Console
ivanti
Velocity
powered by Wavelink
Back
Web Industrial Browser (Web)
Ivanti
Import
Export
Deploy
Keyboards
Advanced Configuration
Settings
Host
Save
Tasks
Scope
Parameters
Scope
Link a script to a scope to determine when it will run. For example, link a script to a specific screen so that it runs when that screen loads.
Example: @T1
Learn More
Scope:
session
OK
Cancel
Submit an Idea
About
Help

Now the parameters become available:



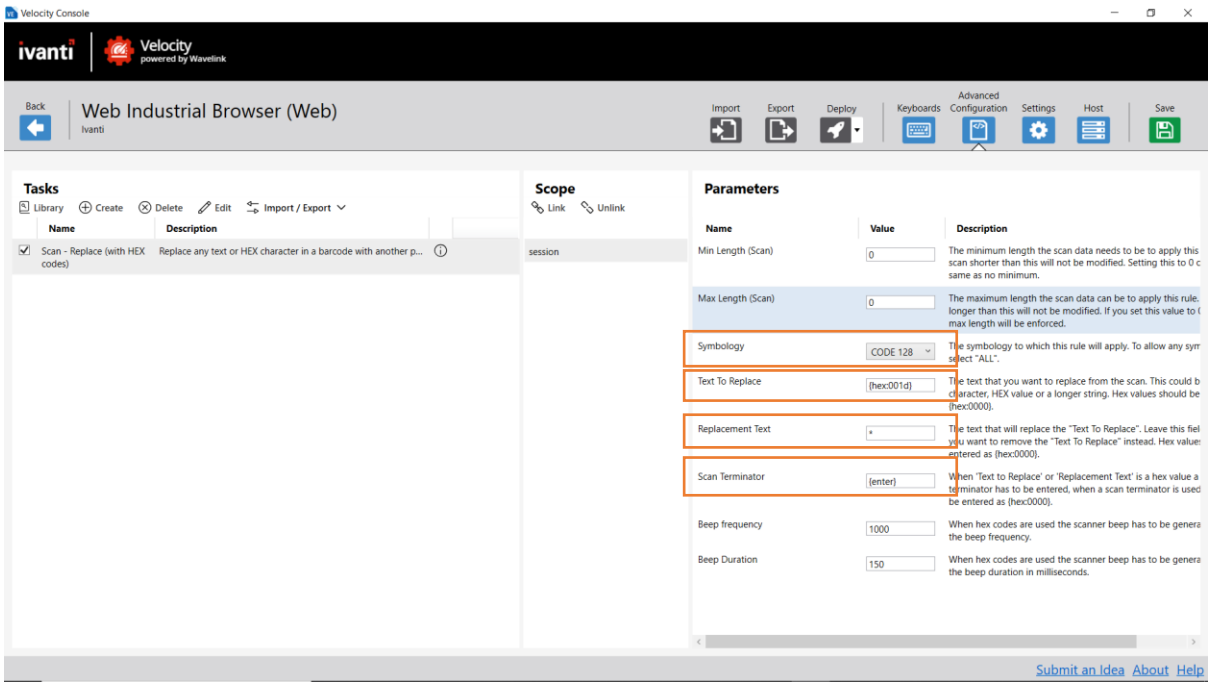
The parameters to replace any FNC1 character in a scanned EAN128 barcode by a '*' are:

Symbology: `CODE 128` (optional: if not selected any barcode will be accepted)

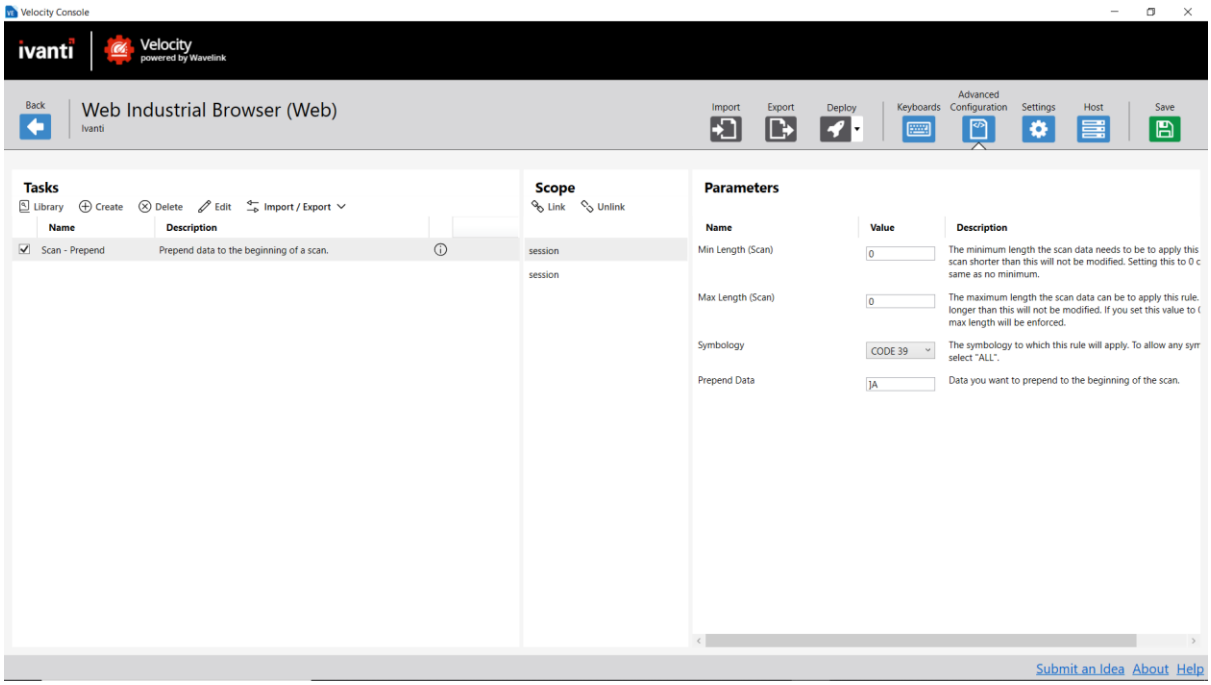
Text To Replace: `{hex:001d}` (hexadecimal value of the FNC1 character)

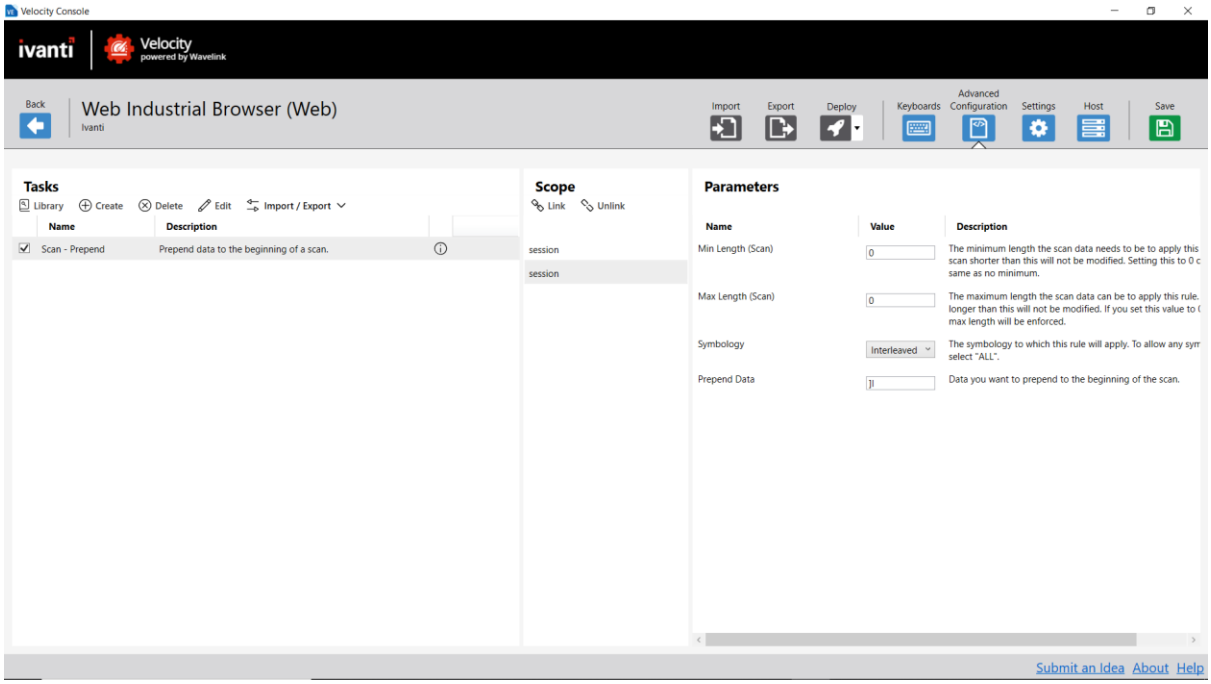
Replacement Text: *

Scan Terminator: `{enter}` (needed for this script)



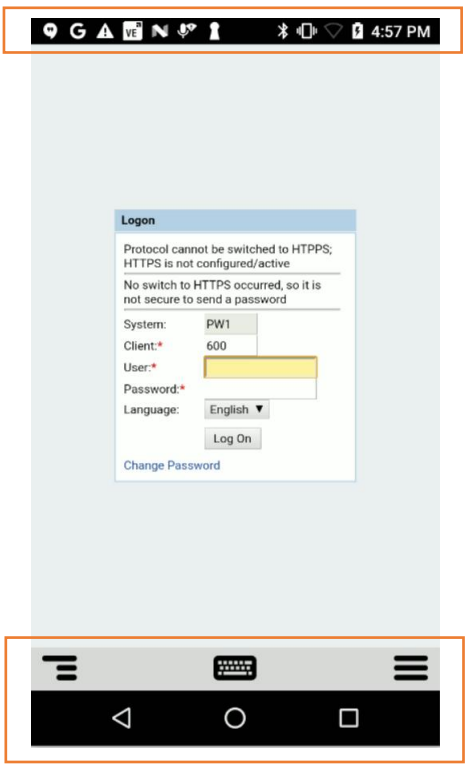
A script can be linked multiple times to with different values. Prepending ‘]A’ to any scanned CODE39 barcode and ‘]I’ to any Interleaved2of5 barcode would look like:





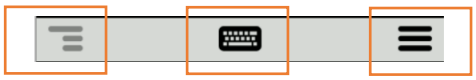
Full Screen Mode

It is not possible to lock down Velocity and have it always in the foreground. To lock down an application in Android a third-party kiosk mode application is necessary, for instance Enterprise Home Screen from Zebra or iLauncher from Honeywell. It is, however, possible to make Velocity full screen by removing the menu bars on screen.



Hide menu/Hide Toolbar

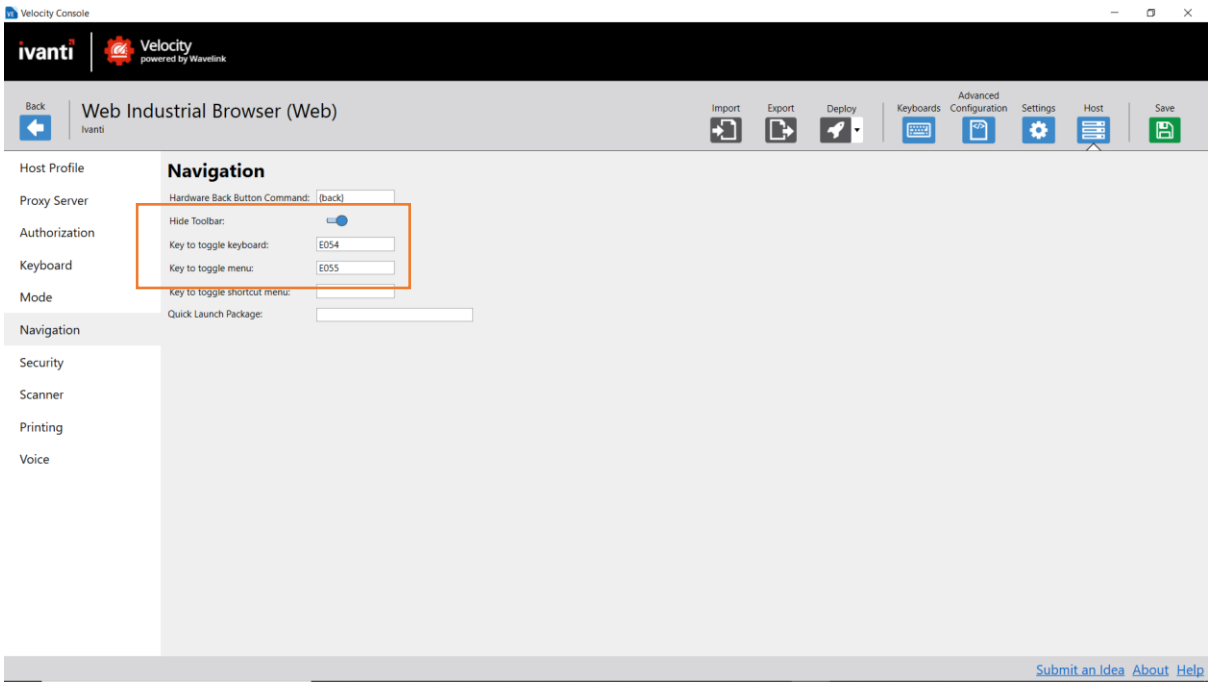
It is very likely that, if the device has a fixed keyboard, there is no need to have the grey menu bar (or toolbar) available for the user, other than to be able to open the menu or shortcut menu or toggle the on-screen keyboard.

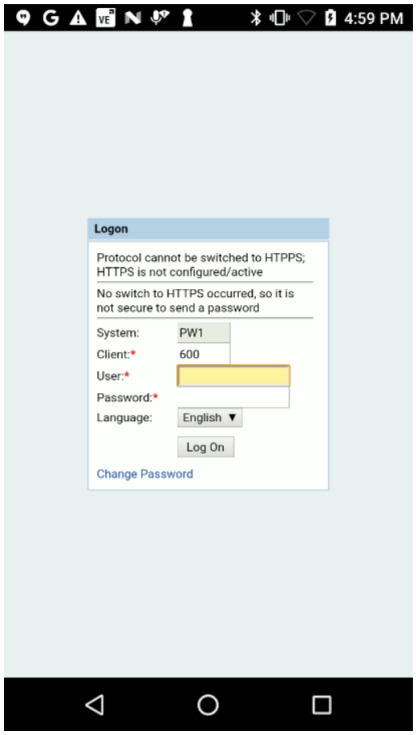


Shortcut Menu Toggle Keyboard Menu

This menu bar can be removed by selecting Hide Toolbar from the Navigation section. Hardware keys can then be defined to open the Shortcut Menu or Menu or Toggle the on-screen keyboard. This is configured with the four-digit hex notation of the specific key. These key codes can be found here: https://help.ivanti.com/wl/help/en_US/Velocity/2.1/admin/keyboardCodes.htm

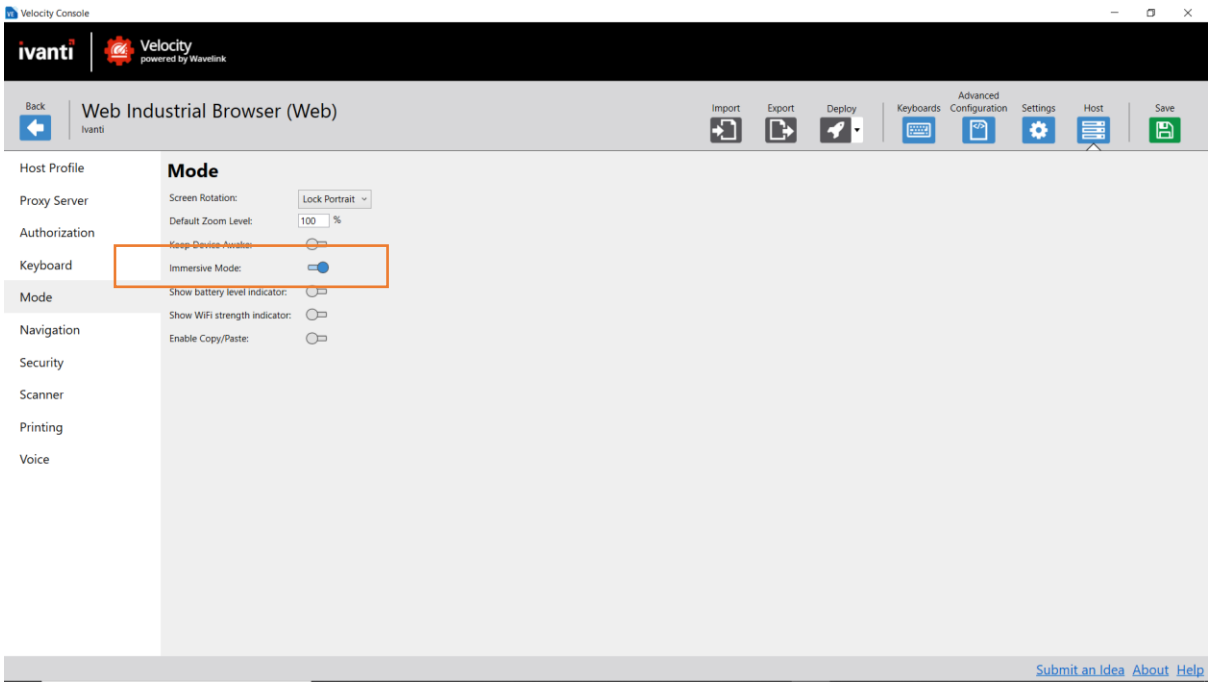
The example below uses F11 to toggle the on-screen keyboard (E054) and F12 to toggle the Menu (E055):





Immersive Mode

The black menu bars at the top and button on the screen can be hidden by enabling Immersive Mode in the Mode section:



These bars are not completely disabled. They will appear when the user swipes down but then automatically blend out again.

Distributing Settings to Devices

After a project is created with the desired host profile and settings, distribute it to your devices. Distributing a project has two steps:

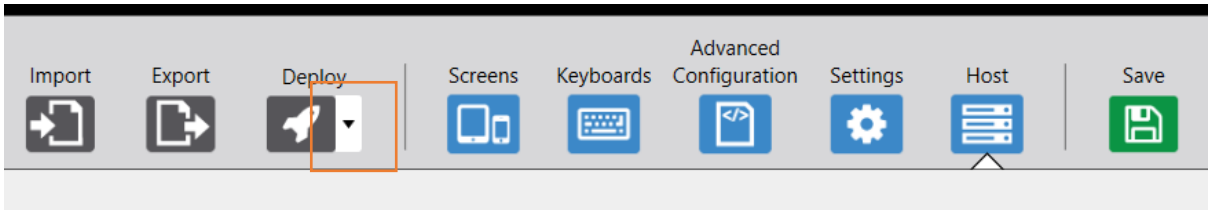
- Deploy the project from the Console. This takes all the settings, files, and scripts and uses them to create a .wldep file.
- Copy the file (or files) to the devices. You can do it manually over USB, or use mobile device management (MDM) software like Ivanti Avalanche.

Deploying a project from the Console

Prior to using a project in the Velocity Client, projects must first be deployed to mobile devices. When a project is deployed, it is stored as a .wldep file containing all the scripts, images, keyboards, and other resources needed to use the project.

To deploy a project from the Velocity Console to a local Windows Client

From the Velocity Console, click the arrow next to the Deploy button and select Deploy locally.



If there is already a deployment file with the same name in the directory, you are prompted to replace it. The project deployment is automatically copied into the %Programdata%\Wavelink\Velocity directory, and the local Client will load the project.

To deploy a project from the Velocity Console

- From the Velocity Console, click Deploy.
- Click the ... button to browse for a save location.
- Navigate to the desired location and enter a file name to save the project as. Click Save to close the dialog.

The project is deployed as a .wldep to the specified destination. It can now be distributed to devices manually or using a mobile device management system like Avalanche.

Distributing a project over USB

If you don't use an MDM system, you'll need to distribute the file over USB.

On Android, the Velocity Client looks for the deployment files in the first external storage partition. It is the storage you see when you connect the device over USB. It is sometimes named 'SD card' or 'external', even though it is not an SD card or external to the device. In the first external storage partition, the files need to be in the following location:

Pre-Android 10: /com.wavelink.velocity

Android 10 and later: Android/data/com.wavelink.velocity/files

If you have an SD card in the device, make sure you create the directory in the storage on the device and not the removable SD card. The Velocity Client will not recognize the deployment file if it is not in the correct location on the device storage.

-Or-

For Windows devices, copy the deployment file (.wldep) or files into the %Programdata%\Wavelink\Velocity directory.

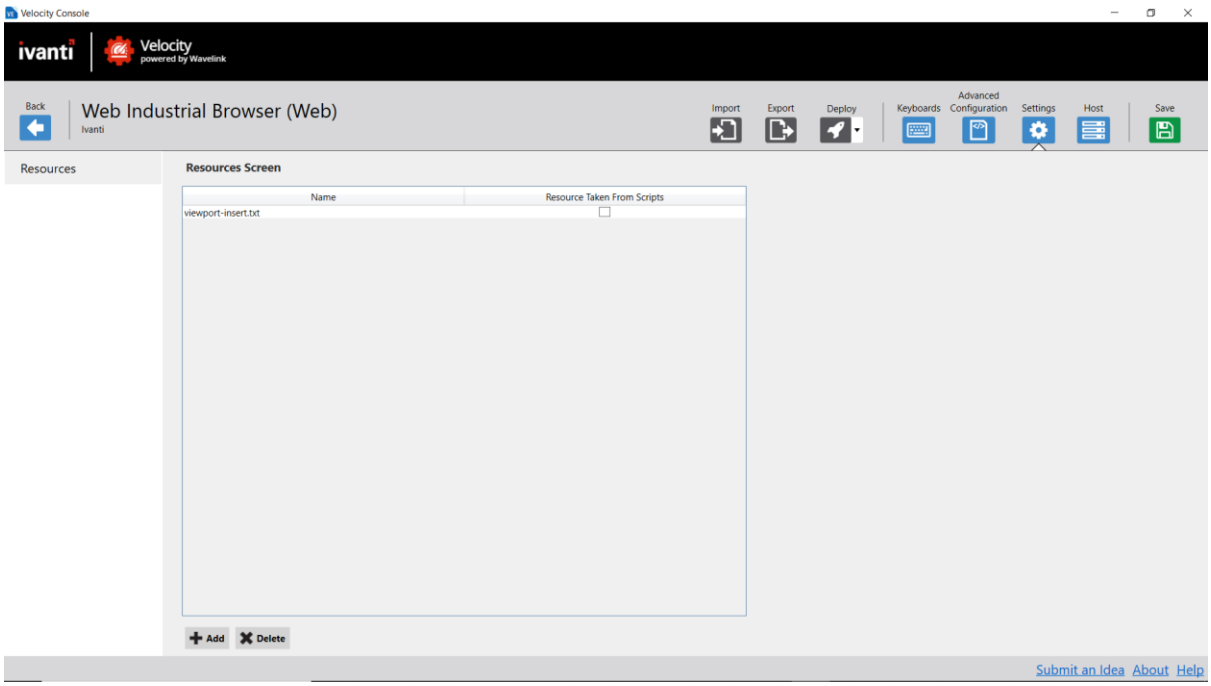
Launch the Velocity Client on the device. The app automatically extracts the project details and displays the new host profile in the list of available profiles. Tap the name of the host profile to connect. When connecting to the host profile associated with a deployed project, all of the keyboards, project settings, and scripts associated with the project are applied to the session.

Web settings – changing application behaviour

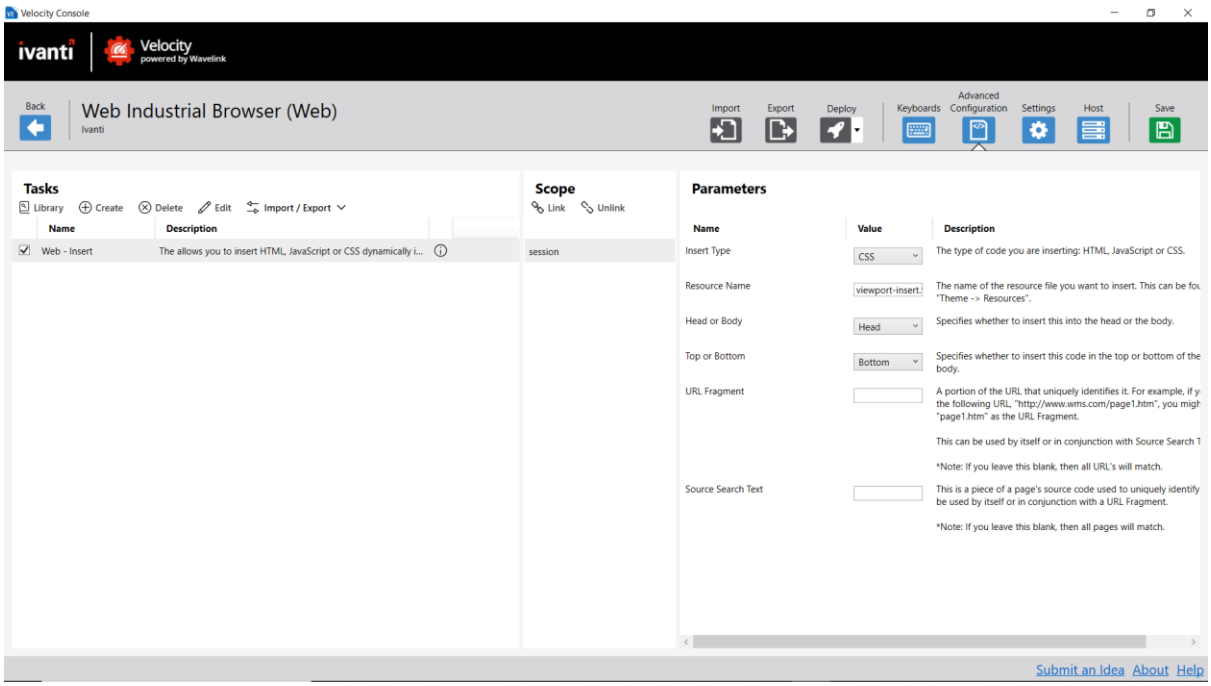
Velocity allows for finetuning of the (behaviour) of the web application by inserting HTML, CSS or javascript code to a web page the moment the page is loaded on the device. These code inserts can create a great impact in the acceptance of the mobile application and can even be the defining factor in making the application work on an Android device.

Web – Insert script

The code inserts are done with the Web – Insert script from the scripting library. The code to insert is stored as a text file and added to the project as a resource in the Settings section of the console, for instance the viewport insert that is described in paragraph 'Set the display with, disable zoom – viewport':

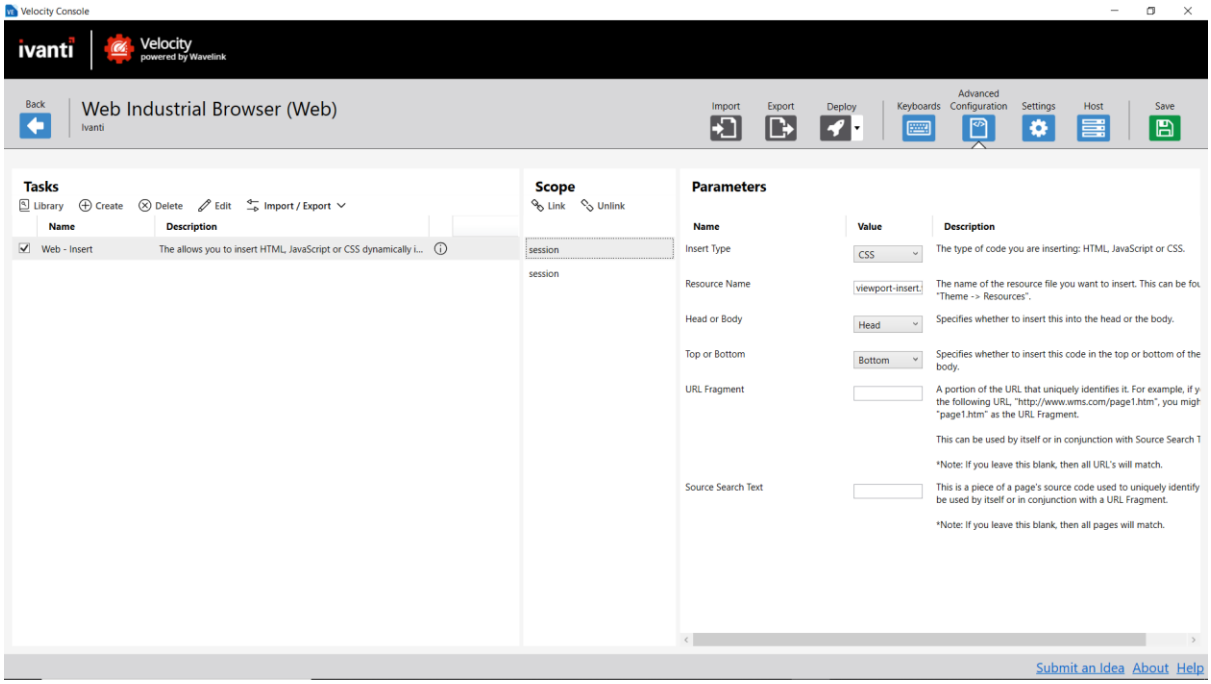


The name of that file is used as a parameter in the Web – Insert script:



The other parameters define what type of code it concerns (HTML, CSS, javascript), there the code a should be added to page (Header Top, Header Bottom, Body Top, Body Bottom) and a definition of the page where to insert the code. If the fields URL Fragment and Source Search Text are left empty the code will be added to every page.

To insert multiple files the Web – Insert script can be linked multiple times:



Insert CSS or javascript as an HTML insert

In the source code of a web page the CSS or javascript code is placed between tags. CSS code is placed between these tags:

```
<style type="text/css">
```

```
</style>
```

Javascript code between these:

```
<script language="JavaScript" type="text/javascript">
```

```
</script>
```

These tags are not necessary in the code insert files that are added to the project. These files will be added as a reference of the corresponding type to the page, for instance:

```
▼<html>
  ▼<head>
    <link rel="stylesheet" type="text/css" href="//velocity/resources/scaleHB.txt">
    <title>ITSmobile login</title>
    <meta http-equiv="OnKey0x0D" content="javascript:MobileSubmitLogin('onLogin')">
    <meta http-equiv="IBrowse_OnKey0x0D" content="javascript:MobileSubmitLogin('onLogin')">
   ▶<style type="text/css">…</style>
   ▶<script language="JavaScript" type="text/javascript">…</script>
    <link rel="stylesheet" type="text/css" href="//velocity/resources/viewport-insert.txt">
  </head>
 ▶<body class="MobileLoginBody" onload="MobileHtmLogin()" onkeydown="return MobileKeyEvent(event, 'onLogin');">…</body>
</html>
```

Actually, if the tags are added the insert will fail and have no effect on the page.

It sometimes occurs that a CSS or javascript insert has no effect although added the correct way. When this happens, it seems that the timing for the page load and inserting the file is wrong. To overcome this problem, add the tags to the insert file and chose HTML as Insert Type:

The code is then added to the source code of the web page:

```
▼<html>
  ▼<head>
   ▼<style>
        body {
            -webkit-transform: matrix(1.900, -0.000, 0.000, 2.500, 0.000, 0.000);
            -webkit-transform-origin: 0.0px 0.0px;
        }
    </style>
    <title>ITSmobile login</title>
    <meta http-equiv="OnKey0x0D" content="javascript:MobileSubmitLogin('onLogin')">
    <meta http-equiv="IBrowse_OnKey0x0D" content="javascript:MobileSubmitLogin('onLogin')">
   ▶<style type="text/css">…</style>
   ▶<script language="JavaScript" type="text/javascript">…</script>
    <link rel="stylesheet" type="text/css" href="//velocity/resources/viewport-insert.txt">
  </head>
 ▶<body class="MobileLoginBody" onload="MobileHtmLogin()" onkeydown="return MobileKeyEvent(event, 'onLogin');">…</body>
</html>
```

This is also usefully if the inserts are done in one single file and not in separate files for HTML, CSS and/or javascript.

Set the display with, disable zoom – viewport

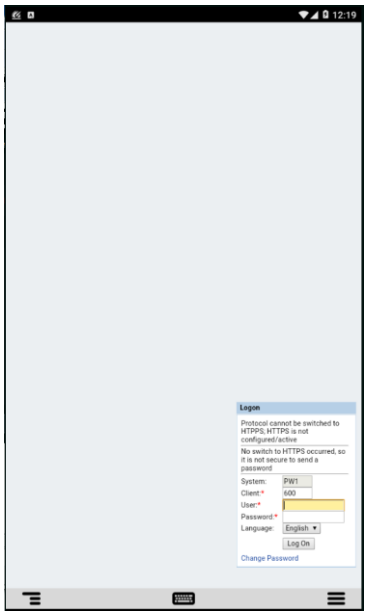
When replacing devices running on a version of Windows to devices running Android the existing web application is most likely developed for QVGA or VGA displays. Because the replacement device will have a display with a higher resolution the application needs to be adapted for the resolution. Also, the new device will support multitouch and allows the user to zoom in and out the application. To define the screen width or disable zooming and HTML insert is used with the viewport metatag:

```
<meta name="viewport" content="width=320, initial-scale=1, user-scalable=no">
```

`width=320` means that the window is 320 pixels wide and that the application should use that width. Often `width=device-width` is used but because of the resolution of the device that usually has no effect on the display size of the application. `user-scalable=no` disables zooming of the display.

Resizing, scaling of the application (SAP Logon)

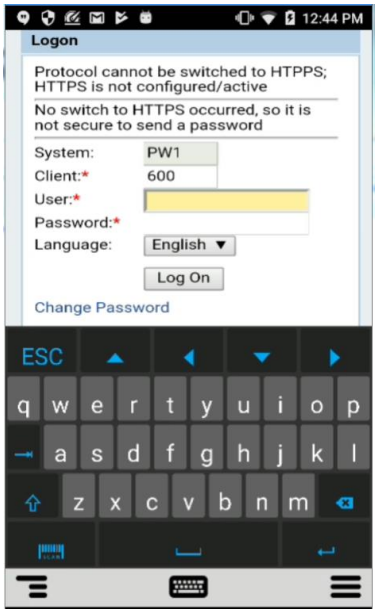
Besides setting the viewport it is sometimes necessary to resize or scale the content on the screen, for instance the login page of an SAP ITS Mobile session.



To make it fit the -webkit-transform: matrix() function can be used:

```
body {
	-webkit-transform: matrix(1.650, -0.000, 0.000, 1.350, -115.000, -50.000);
	-webkit-transform-origin: 0.0px 0.0px;
}
```

This code scales 1.650 times in the x-direction, 1.350 times in the y -direction and moves the screen 115.000 pixels to the left and 50.00 pixels up.



Resizing SAP ITS Mobile

With SAP ITS Mobile, pages might need some further tuning besides adding a viewport metatag and scaling the page content. The style of every element that is on the page is defined in the include file mobile mobile.css and without any further adjustment an element will take up 100% of the available width. This width is not defined by the width of the application but by that of the display canvas that can be used.

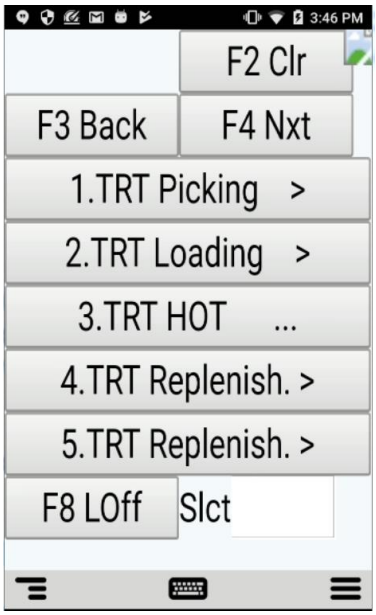
This menu is drawn in two areas on the screen: the MobileHML and MobileBody. Other implementations use MobileScreen as well:

This example redefines these three classes to make the page fit, as well as setting the viewport and scale factor. The insert is done in one text file that is inserted as HTML code. Note, as part of the example MobileScreen is added to the insert too:

```
<meta name="viewport" content="width=190, initial-scale=1, user-scalable=no">


<style>
body {
        -webkit-transform: matrix(1.900, -0.000, 0.000, 2.500, 0.000, 0.000);
        -webkit-transform-origin: 0.0px 0.0px;
}

.MobileHtml {
  width:190px;
}

.MobileScreen {
  width:190px;
}
.MobileBody {
  width:190px;
}
```

```
</style>
```

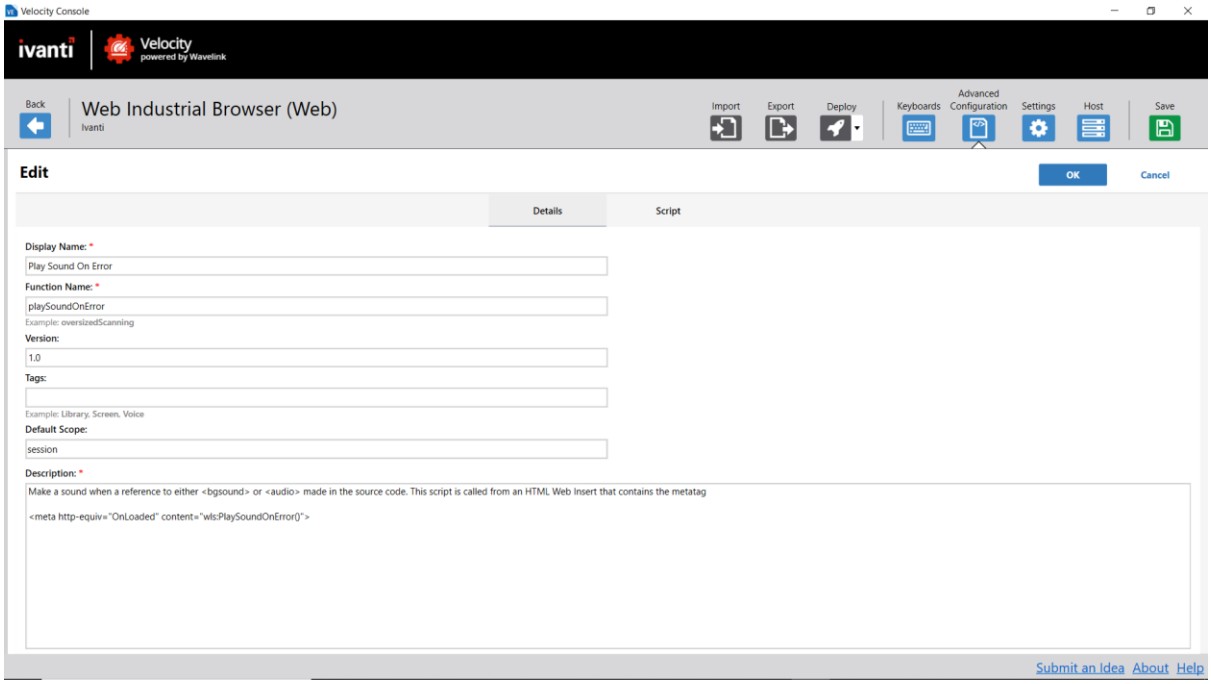
The result with the insert loaded:

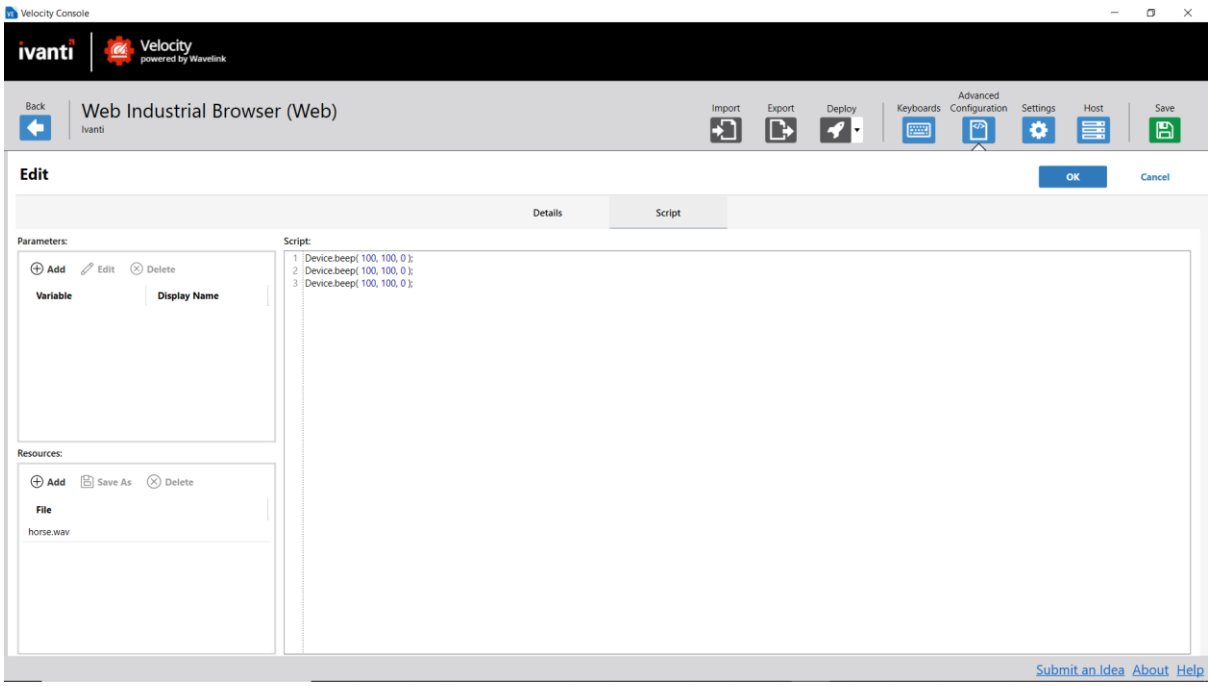


SAP ITS Mobile Error sound

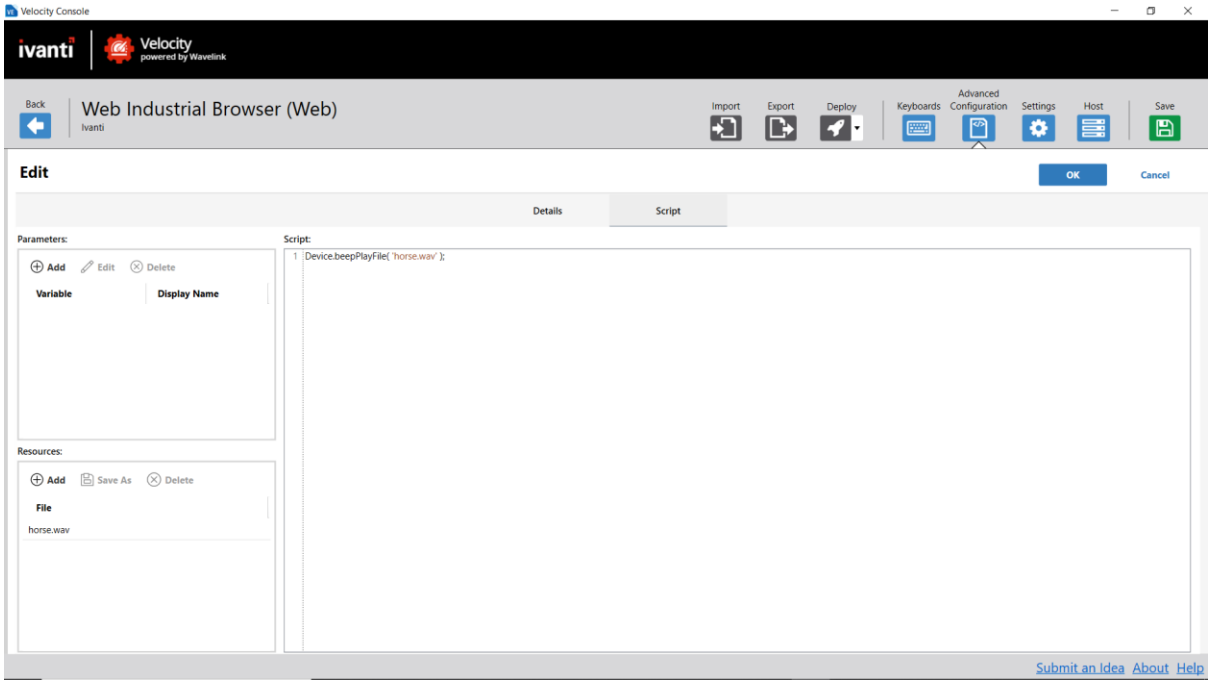
Applications in SAP ITS Mobile are usually designed to be used with Internet Explorer or Edge as the browser. To generate a sound either the `<bgsound>` or `<audio>` tags are used in the source code. These tags are not supported by the Webview on an Android device and are therefore also not supported in Velocity.

The solution is to call a script every time a reference to one of the tags is made. In this example will be called playSoundonError and can either generate beeps:





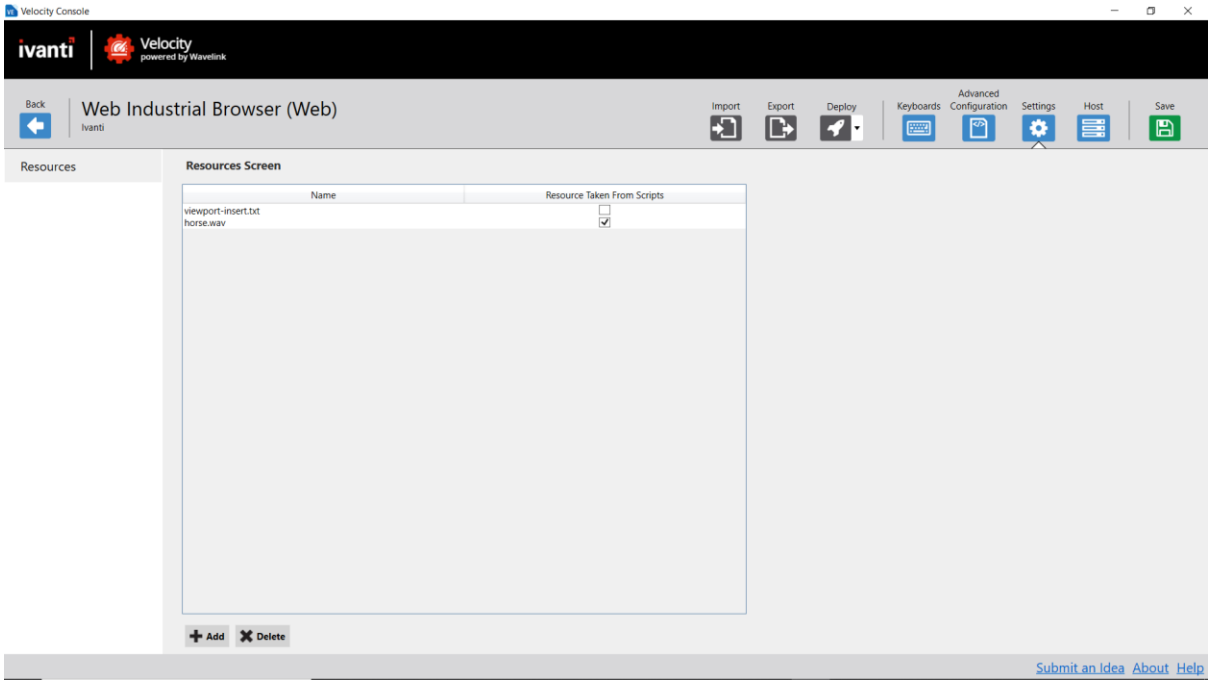
Or use an audio file:



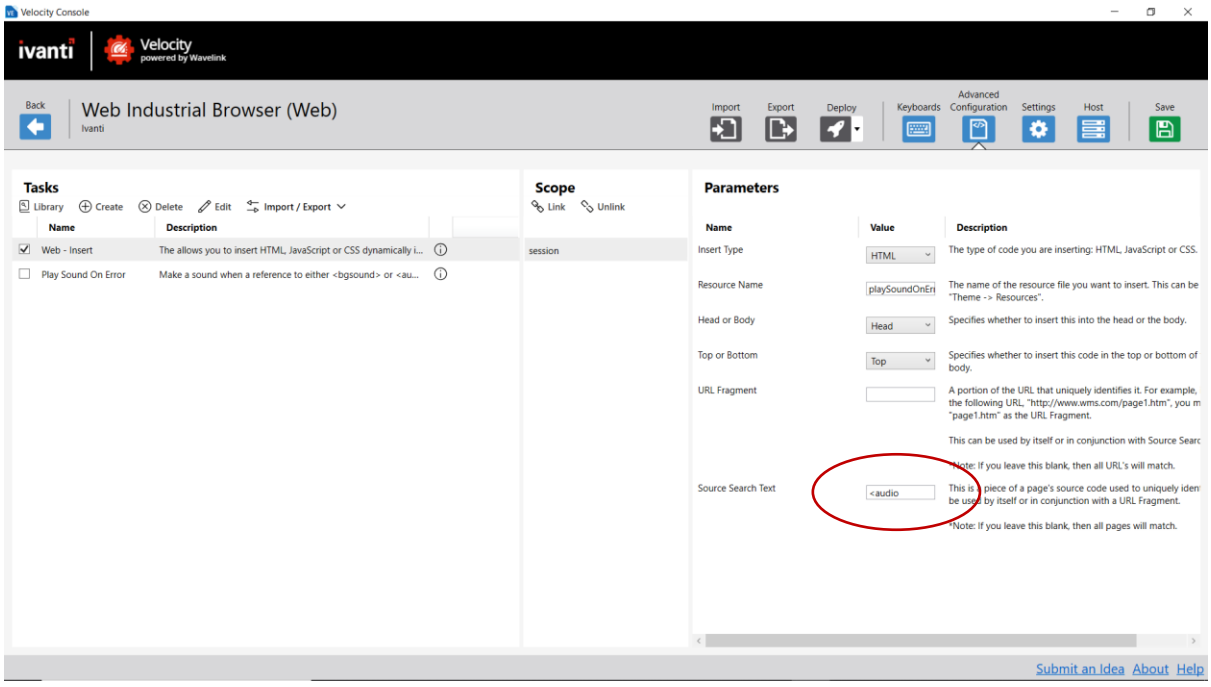
The script does not have to be linked to any scope. It will be called from an HTML insert that uses the code `<meta http-equiv="OnLoaded" content="wls:playSoundOnError()">`

The prefix `wls:` in the metatag refers to the script function that is defined in the Velocity project. Without it, it would refer to a function that is inside the web page source code.

The text file that contains the metatag is then added to the resources of the project (note that in the resources list the name of the audio file that is added to the script is also displayed):



Then, a Web – Insert script is used to call the Play Sound On Error script when the text <audio is found in the source code of a web page:



SAP ITS Mobile Function Keys

When a function key is pressed in SAP ITS Mobile a specific javascript function should be fired. Usually, this function is called setFKey() for function keys or setOKCodeEnter() for the enter key. The

definition of these functions is done in the web page. setfKey() takes the number of the function key as a parameter. So, 1 for F1, 2 for F2, etc. When the application does not respond to any of these keys the following script will solve that:

```
// This is the callback function to replace the keys
// The key code in event.data is the decimal key code as a string
function KeyMacro(event)
{
        switch( event.keyCode )
        {
                // F1 - F12
                case '112':
                case '113':
                case '114':
                case '115':
                case '116':
                case '117':
                case '118':
                case '119':
                case '120':
                case '121':
                case '122':
                case '123':
                        View.evaluateJavascript("(function(){" +
                                "setFKey(event.keyCode - 111);" +
                                "})()"
                        , "");
                        break;
                // Enter
                case '13':
                        View.evaluateJavascript("(function(){" +
                                "setOkCodeEnter();" +
                                "})()"
                        , "");
                        break;
             default:
                        break;
        }
}

// Register the key macro function to be called at keypress
WLEvent.on( "Key", KeyMacro );
```

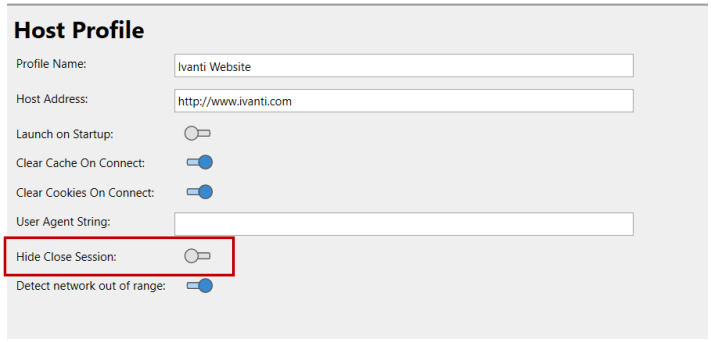
Note: in some installations the functions to be called have different names. These can then be changed in this script.

Automatically end session (after SAP logoff)

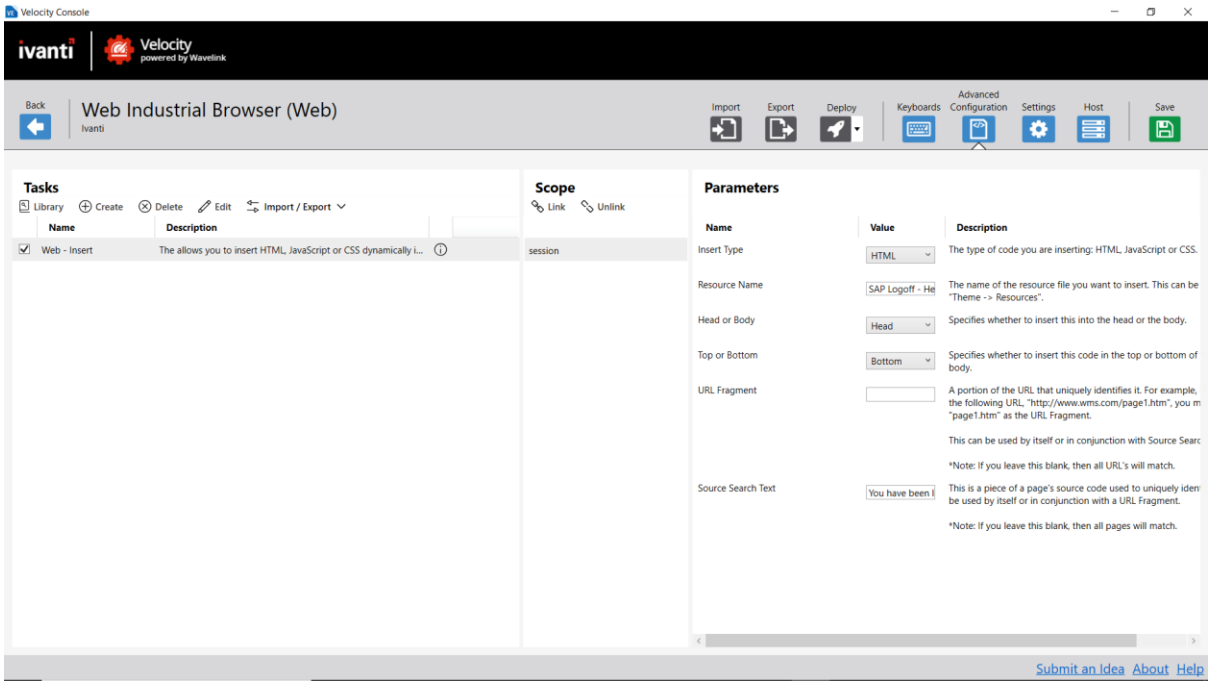
Like the Wavelink Industrial Browser, Naurtech CETerm and Velocity CE several IDA codes can be used with Velocity. One of these IDA codes will disconnect a websession:

```
<meta http-equiv="OnLoaded" content="ida:IDA_SESSION_DISCONNECT">
```

This code can be used to automatically end a web session when the user logs off in SAP. Logging off in SAP usually means that the session in the back end is closed but not the corresponding web session. If done right, the user will land on the login page again. If done wrong, a message is displayed to confirm that the user is logged of and the session needs to be closed manually. In a locked-down environment where the option 'Hide Close Session' is enabled in the host profile the IDA code is necessary:

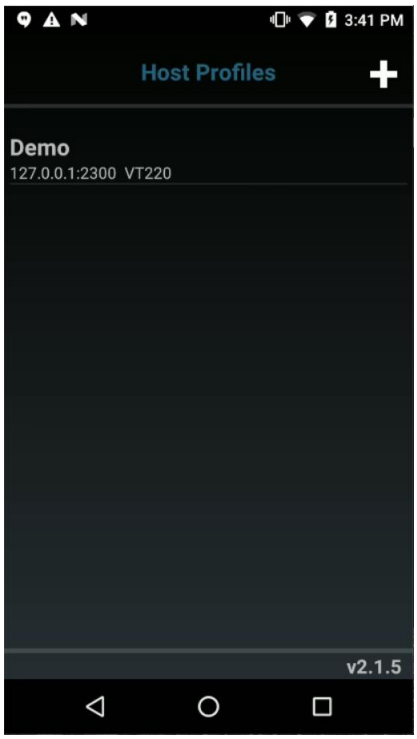


Call the insert file when the logged off message is displayed:



Delete Demo project from main menu

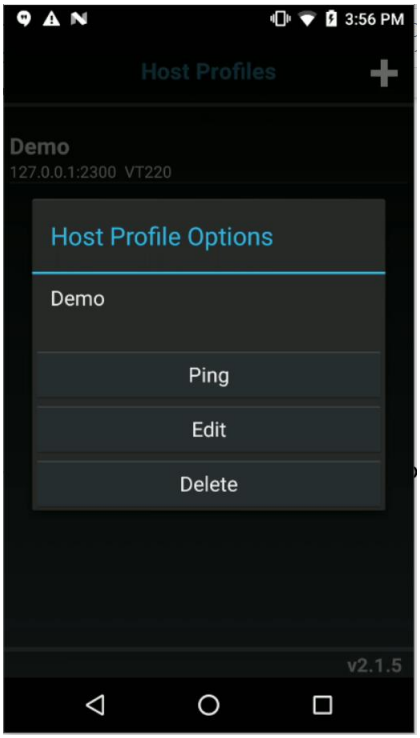
After installation, Velocity will have one entry – called Demo – in its main menu:



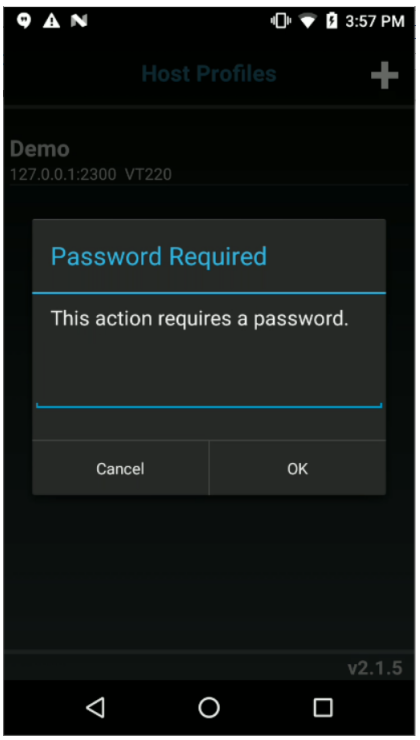
This Demo project can be deleted, either manually or by copying a global settings project file to the device.

Delete manually

Tap and hold the entry until the Host Profile Options menu is displayed:



Select ‘Delete’. To complete this action a password must be entered. This password is ‘system’:



Delete in global settings

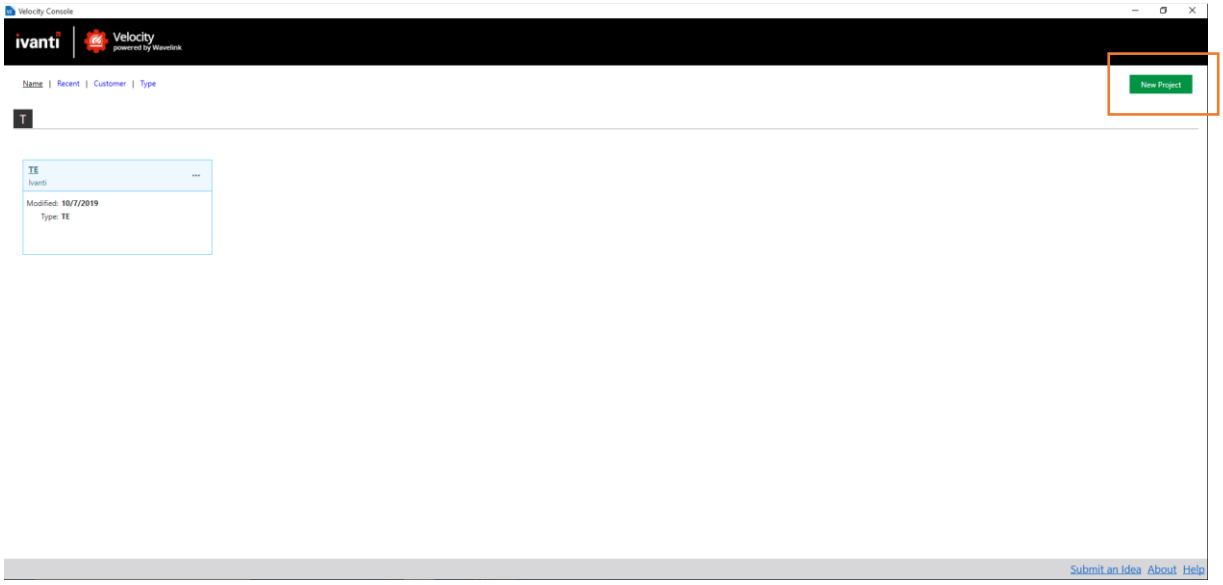
To use the same settings with different projects, create a Global project with settings that can be distributed with multiple projects. A Global project has the following options:

- Configuration password. Provide a custom password that device users must type before they can access the settings on the device. Use the Confirm Password text box to make sure you typed the password correctly. The default password for editing the settings on the device is system.
- Max sessions. The maximum number of concurrent sessions allowed.
- Hide Exit. Hides the Exit button from the menu.
- Remove Demo Profile. Removes the demo profile from the device, so it no longer shows in the list of host profiles.
- Lock task mode. The Velocity Client opens full screen, hides the taskbar and Start menu, prevents task switching and minimizing, and blocks Alt+Tab, Alt+F4, Alt+Space, and Ctrl+Esc. In order to exit the Client, the user must provide the program exit password. Only available for Velocity Clients installed on Windows 10.
- Program exit password. A password required to exit the Client on Windows 10 machines when Lock task mode is enabled. By default, the program exit password is blank. Even if you do not specify a program exit password, the user is prompted to provide a password if they attempt to exit the Client.
- Key to switch to next session. The hex value for a key to switch to the next open session. For example: E040
- Key to switch to previous session. The hex value for a key to switch to the previous open session. For example: E041

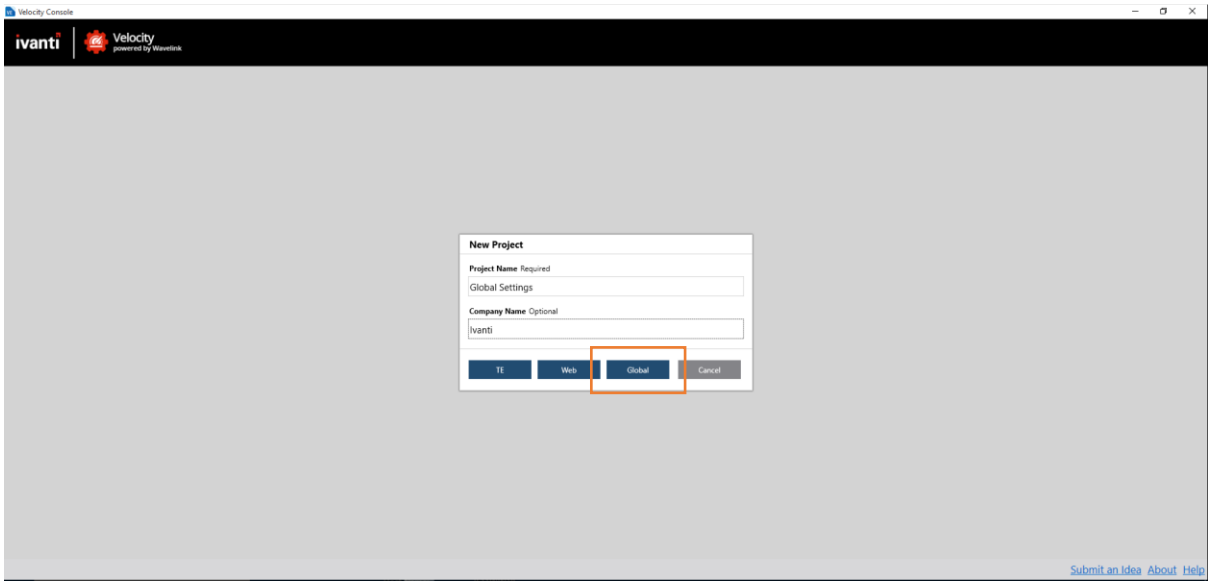
When you deploy a global settings project, those settings are applied for all host profiles used by the Velocity Client.

To create a global settings project

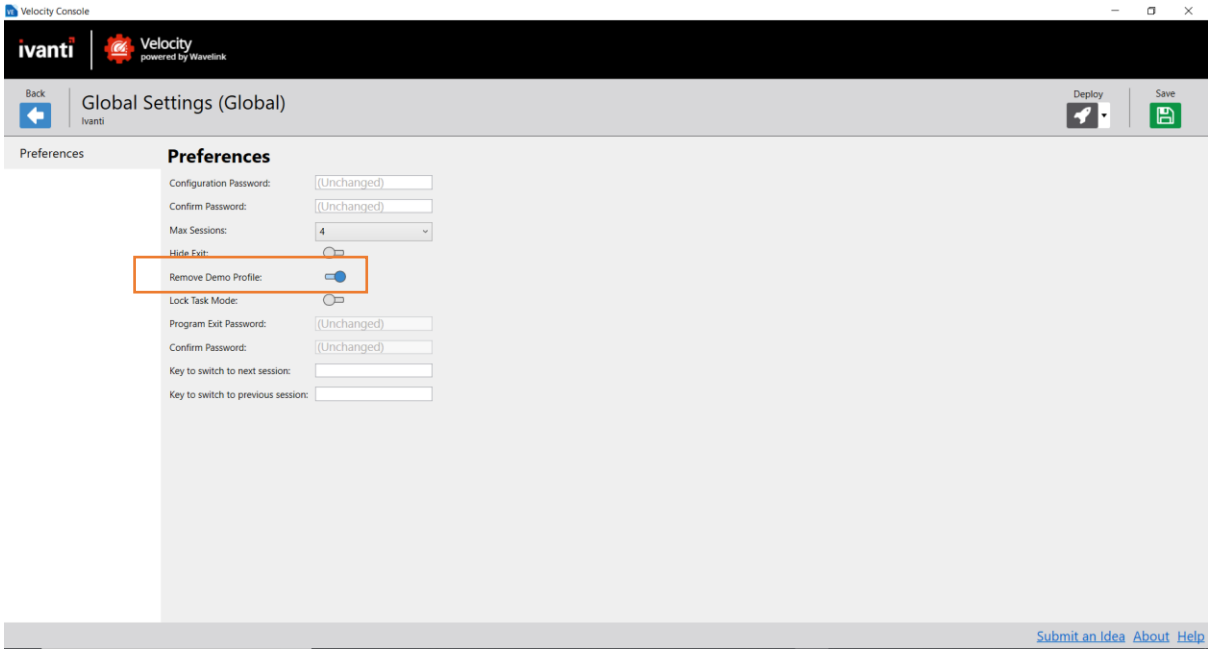
Launch the Velocity Console application and click ‘New Project’ in the top-right corner of the screen.



Enter a Project Name. You can also provide a Company Name, which is used solely for sorting projects. Click Global.

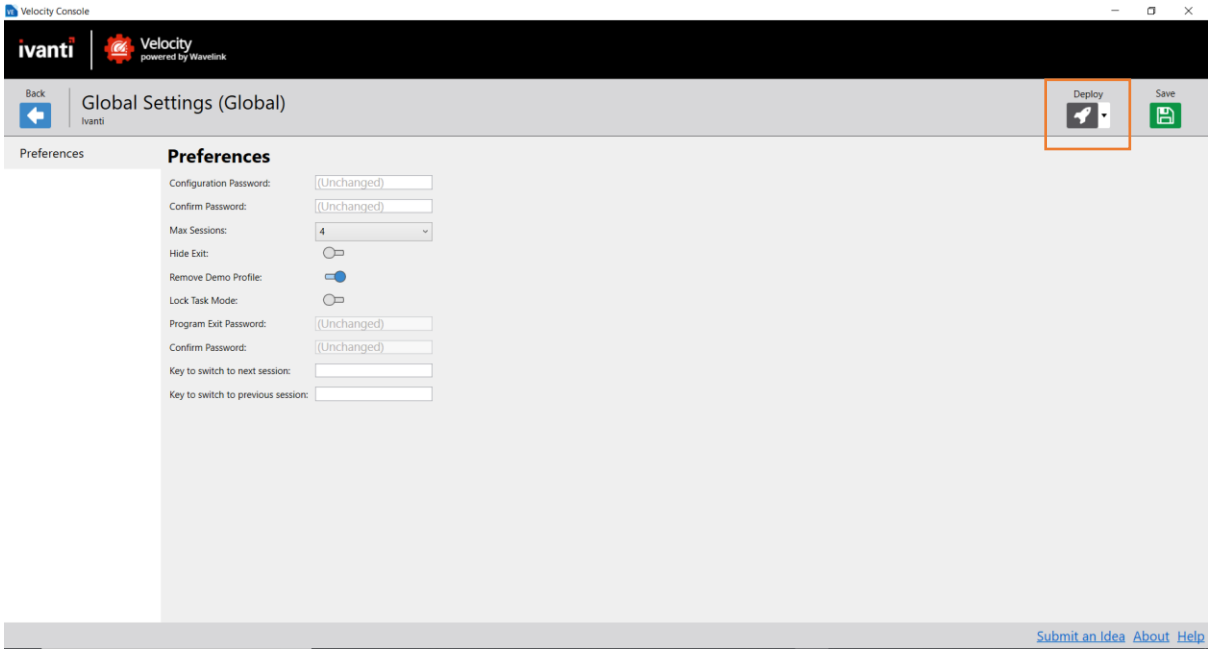


The project opens and you can edit global settings to be distributed with other projects. Enable ‘Remove Demo Profile’.



To deploy a global settings project

Click the Deploy button in the top right corner.



Navigate to the location where you want to save the file. The file name must be global.wlxgp. Click 'Save' to close the dialog. Copy the file to the device, in same location as where the .wldep files are stored.

On Android, the Velocity Client expects the deployment files in the first external storage partition. It is the storage you see when you connect the device over USB. It is sometimes named 'SD card' or

'external', even though it is not an SD card or external to the device. In the first external storage partition, the files need to be in the following location:

Pre-Android 10: /com.wavelink.velocity

Android 10 and later: Android/data/com.wavelink.velocity/files

If you have an SD card in the device, make sure you create the directory in the storage on the device and not the removable SD card. The Velocity Client will not recognize the deployment file if it is not in the correct location on the device storage.

-Or-

For Windows devices, copy the deployment files or files into the %Programdata%\Wavelink\Velocity directory.

Appendix A. Send data to Velocity using intents

Velocity listens for a broadcast intent to receive (barcode) data. This mechanism can be used to insert data in a telnet or web session. The format of the intent is:

Action:	"com.wavelink.intent.action.BARCODE"
Category:	"android.intent.category.DEFAULT"
Extra for (barode label) data:	String extra with a name ending in the format of "*.data_string" or "*.decode_string", e.g. "com.something.something.decode_string"
Extra for barcode label type:	String extra with a name ending in the format of "*.symbology_type" or "*.label_type" , e.g. "com.something.something.symbology_type"

A WLEvent.on('scan', function) event will be generated when data is received in this format. Further processing of the data can then be done in the callback function attached to that event.

If the Velocity client is used on a device that uses a scan engine that is not supported by Velocity, but has configurable wedge software installed to allow sending the scanned data as an intent, the following example might help setting up the scanner. This example describes configuring the scanner on a Panasonic device:

1. Ensure that the Velocity Client is installed on the device before starting.
2. Once installed, Open the "Barcode Reader"
3. Tap the Menu button(...) and choose "Select Profile"
4. Choose "Create Profile"
5. Enter "Velocity" as the profile name
6. Tap on the newly created profile, "Velocity", to change the settings
7. Tap on "Associated Apps"
8. Tap the Menu button (...) and choose "New app/activity"
9. Select "com.wavelink.velocity" as the app
10. Select "*" as the activity
11. Tap on the back button to return to the settings for the Velocity profile.
12. Tap to disable "keyboard Wedge"
13. Tap "Intent output settings"

14. Tap to enable "Intent output"
15. Tap to change "Intent action name"
16. Enter "com.wavelink.intent.action.BARCODE", Press "OK"
17. Tap to change the "Intent category name"
18. Enter "android.intent.category.DEFAULT", press "OK"
19. Tap to change "Intent delivery type"
20. Tap "Broadcast intent"
21. Hit the back button to return to the settings for the Velocity profile
22. Configure "Symbologies" and any other settings that you may need.