WebScanner  
Implementation guide

Version 1.3.1

2025-03-21

Contents

[About the scanner 3](#_Toc73956490)

[Installation 4](#_Toc73956491)

[Https 4](#_Toc73956492)

[Usage 4](#_Toc73956493)

[Configuration 5](#_Toc73956494)

[Processing data 5](#_Toc73956495)

# About the scanner

The cWebScanner is a standalone component that provides a way of scanning a vast number of barcode formats. It is based on Google’s ZXing (ZebraCrossing) library and wrapped for easy use in DataFlex WebApps.

Supported formats are:

1-Dimensional

* code\_128\_reader (default)
* ean\_reader
* ean\_8\_reader
* code\_39\_reader
* code\_39\_vin\_reader
* codabar\_reader
* upc\_reader
* upc\_e\_reader
* i2of5\_reader
* 2of5\_reader
* code\_93\_reader

2-Dimensional

* QR

# Installation

Copy the WebScanner folder from cWebScanner/AppHtml to your application’s AppHtml folder

Add the required references to these files in your index.html:

     <!-- WEBSCANNER COMPONENTS -->

        <script src="WebScanner/quagga.min.js"></script>

        <script src="WebScanner/core-estimator/core-estimator.min.js"></script>

<script src="WebScanner/qr-scanner/paulmillr-qr.js"></script>

        <script src="WebScanner/WebScanner.js"></script>

        <link rel="stylesheet" href="WebScanner/WebScanner.css">

Add the cWebScanner as a library to your DataFlex workspace, select the correct version. For this, in the studio go to “Tools” 🡪 “Maintain Libraries” and add the workspace as a library.

You can now use the cWebScanner component in your application.

# Https

Note that outside of testing on your **localhost** the scanner will only work in **https** environments as it requires access to device hardware.

# Usage

The scanner detects and uses your device’s integrated cameras and allows for switching between cameras if your device has multiple (e.g. front and rear facing camera on a smartphone). To begin using the cWebScanner, simply create an object instance and set the format you want to scan for.

Object oScanner is a cWebScanner

Set piColumnSpan to 0

Set peCodeFormat to CODE\_FORMAT\_QR

// Set psCodeFormatMulti to "ean\_reader, ean\_8\_reader"

// Set psCodeFormatMulti to "codabar\_reader, code\_128\_reader, code\_39\_reader"

Procedure OnProcessCodeResult tCodeResult CodeResult

Forward Send OnProcessCodeResult CodeResult

WebSet psValue of oWfResult to CodeResult.sCode

End\_Procedure

End\_Object

The scanner above reads a QR code and outputs the result in a separate webform.

## Configuration

All properties are extensively documented inside the cWebScanner class, but the main properties to use are:

**peCodeFormat (integer)** – defines to type of code to scan for, uses a set of constants for ease of access (CODE\_FORMAT\_128, CODE\_FORMAT\_EAN, etc.)

**psCodeFormatMulti (String)** – Allows for defining a set of multiple codes to scan for using a comma separated string. The format name must be exactly correct or this instruction will fail. A list of accepted code formats can be found in the comments inside the class.

**piScannerWidth and piScannerHeight (Integer)** – Sets the dimensions for the scanner’s video feed in the WebApp.

## Processing data

After each successful scan, **OnProcessCodeResult** is called. This contains a **tCodeResult** containing the data inside the code, as well as information about the scan and the code itself. The data can then be used for further processing inside your application.