Image Generator SDK

Integration Guide

Version: 4.22

Copyright Notice

Copyright © 2013–2022 OneSpan North America, Inc. All rights reserved.

Trademarks

OneSpan™, DIGIPASS® and CRONTO® are registered or unregistered trademarks of OneSpan North America Inc., OneSpan NV and/or OneSpan International GmbH (collectively "OneSpan") in the U.S. and other countries.

OneSpan reserves all rights to the trademarks, service marks and logos of OneSpan and its subsidiaries.

All other trademarks or trade names are the property of their respective owners.

Intellectual Property

OneSpan Software, documents and related materials (“Materials”) contain proprietary and confidential information. All title, rights and interest in OneSpan Software and Materials, updates and upgrades thereof, including software rights, copyrights, patent rights, industrial design rights, trade secret rights, sui generis database rights, and all other intellectual and industrial property rights, vest exclusively in OneSpan or its licensors. No OneSpan Software or Materials may be downloaded, copied, transferred, disclosed, reproduced, redistributed, or transmitted in any form or by any means, electronic, mechanical or otherwise, for any commercial or production purpose, except as otherwise marked or when expressly permitted by OneSpan in writing.

Disclaimer

OneSpan accepts no liability for the accuracy, completeness, or timeliness of content, or for the reliability of links to and content of external or third party websites.

OneSpan shall have no liability under any circumstances for any loss, damage, or expense incurred by you, your company, or any third party arising from the use or inability to use OneSpan Software or Materials, or any third party material made available or downloadable. OneSpan will not be liable in relation to any loss/damage caused by modification of these Legal Notices or content.

Reservation

OneSpan reserves the right to modify these Notices and the content at any time. OneSpan likewise reserves the right to withdraw or revoke consent or otherwise prohibit use of the OneSpan Software or Materials if such use does not conform to the terms of any written agreement between OneSpan and you, or other applicable terms that OneSpan publishes from time to time.

Contact us

Visit our website: https://www.onespan.com
Resource center: https://www.onespan.com/resource-center
Technical support and knowledge base: https://www.onespan.com/support

If there is no solution in the knowledge base, contact the company that supplied you with the OneSpan product.

Date: 2022-02-03

Contents

1 Introduction 1

1.1 OneSpan Mobile Security Suite Documentation 2

2 Usage of the Image Generator SDK 4

2.1 Overview 5

2.2 GenerateBWQRCode 7

2.3 GenerateCrontoSign and GenerateDynamicCrontoImage 8

3 Integrate the Image Generator SDK 11

3.1 Integrate the OneSpan Image Generator SDK in a Java Project 12

3.2 Integrate the OneSpan Image Generator SDK in a .NET Project 13

Figures

Figure 1: QR Code 5

Figure 2: Cronto Image 5

Tables

Table 1: GenerateBWQRCode parameters 7

Table 2: Number of squares in generated images 8

Table 3: GenerateCrontoSign / GenerateDynamicCrontoImage parameters 9

Procedures

To use the OneSpan Image Generator SDK in your Java project 12

To use the OneSpan Image Generator SDK in your .NET project 13

1 Introduction

Welcome to the Image Generator SDK Integration Guide! This guide describes how to integrate the Image Generator SDK into any application.

This guide provides information about:

- Functions of the Image Generator SDK, with parameter and method descriptions
- Integration steps

1.1 OneSpan Mobile Security Suite Documentation 2

1.1 OneSpan Mobile Security Suite Documentation

The OneSpan Mobile Security Suite product documentation comprises the following guides:

- *OneSpan Mobile Security Suite Product Guide*: Provides an overview of OneSpan Mobile Security Suite and its components.
- *Biometric Sensor SDK Integration Guide*: Provides instructions to integrate the Biometric Sensor SDK.
- *Client Device Data Collector SDK Integration Guide*: Provides instructions to integrate the Client Device Data Collector SDK.
- *Device Binding SDK Integration Guide*: Provides instructions to integrate the Device Binding SDK.
- *DIGIPASS SDK Integration Guide*: Provides instructions to integrate the DIGIPASS SDK.
- *Digipass Software Advanced Provisioning Protocol SDK Integration Guide*: Provides instructions to integrate the Digipass Software Advanced Provisioning Protocol SDK.
- *FIDO UAF SDK Developer Guide*: Provides instructions to integrate the FIDO UAF SDK.
- *Image Generator SDK Integration Guide*: Provides instructions to integrate the Image Generator SDK.
- *Image Scanner SDK Integration Guide*: Provides instructions to integrate the Image Scanner SDK.
- *Notification SDK Integration Guide*: Provides instructions to integrate the Notification SDK.
- *Orchestration SDK Integration Guide*: Provides instructions to integrate the Orchestration SDK.
- *Root Detection SDK Integration Guide*: Provides instructions to integrate the Root Detection SDK.

- *Secure Messaging SDK Client Integration Guide*: Provides instructions to integrate the Secure Messaging SDK Client.
- *Secure Messaging SDK Server Integration Guide*: Provides instructions to integrate the Secure Messaging SDK Server.
- *Secure Storage SDK Integration Guide*: Provides instructions to integrate the Secure Storage SDK.
- *White-Box Cryptography SDK Integration Guide*: Provides instructions to integrate the White-Box Cryptography SDK.

2 Usage of the Image Generator SDK

This section contains an overview of the Image Generator SDK and information about the functions of the SDK.

2.1 Overview 5

2.2 GenerateBWQRCode 7

2.3 GenerateCrontoSign and GenerateDynamicCrontoImage 8

2.1 Overview

The OneSpan Image Generator SDK generates standard black and white QR codes and **Cronto images**[1] from a string value.



Figure 1: QR Code

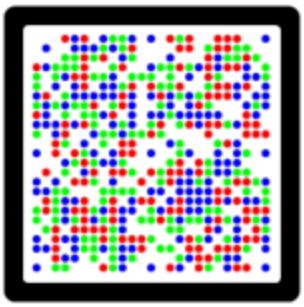


Figure 2: Cronto Image

The Image Generator SDK package includes the following:

- `ImageGeneratorSDK.jar` for Java platforms
- `ImageGeneratorSDK.dll` for .NET platforms

The Image Generator SDK can be used on a variety of devices and supports the following platforms:

- Image Generator SDK Java edition: Java-enabled platforms (JDK 1.8 and later)
- Image Generator SDK .NET edition: ..NET Framework 2.0 and later

2.1.1 Supported image formats

QR code

A **QR code** is a two-dimensional bar code. It is composed of black modules (square dots) arranged in a square grid on a white background. The QR code is defined as an ISO (ISO/IEC18004) standard.

[1] Colorful cryptogram, similar to a QR code; used for visual transaction signing.

Cronto image

A **Cronto image** is a specific visual cryptogram which is composed of four components:

- An image that consists of several squares (e.g. 25 x 25 squares), and each square has a specific color. The colors used are white, red, green, or blue.
- A black border with a width of two squares.
- A *quiet zone* outside the black border with a width of at least two squares.
- A *quiet zone* inside the black border with a width of one square.

The data starts from the top-left corner downwards and proceeds by columns; a **Cyclic redundancy check**[1] checksum is added for error detection.

[1] See Cyclic redundancy check.

2.2 GenerateBWQRCode

The `GenerateBWQRCode` method is used to generate QR code images according to ISO No.18004.

2.2.1 Parameters

Table 1: `GenerateBWQRCode` **parameters**

Parameter name	Data type	Description
`imageSize`	int	Width of a QR code image in pixels. The minimum image size is 25 pixels (`ImageGeneratorSDKConstants.QRCODE_MIN_IMAGE_SIZE`), the maximum image size is 700 pixels (`ImageGeneratorSDKConstants.QRCODE_MAX_IMAGE_SIZE`).
`inputValue`	String	The string to be encoded.
`errorCorrectionLevel`	int	The error correction level. Possible values can be retrieved from the `ImageGeneratorSDKConstants` class; they are: • `QRCODE_ERROR_CORRECTION_LEVEL_LOW` • `QRCODE_ERROR_CORRECTION_LEVEL_MIDDLE` • `QRCODE_ERROR_CORRECTION_LEVEL_QUALITY` • `QRCODE_ERROR_CORRECTION_LEVEL_HIGH`

2.2.2 Return value

The returned value is a `BWQRCodeResponse` object. It contains the QR code image (which is a buffered image object in the Java edition and a bitmap object in the .NET edition), as well as the associated Boolean matrix.

For more information about this method, refer to the technical documentation delivered in the Image Generator SDK package in addition to the *OneSpan Image Generator SDK Integration Guide*.

2.3 GenerateCrontoSign and GenerateDynamicCrontoImage

The Image Generator SDK provides two APIs to generate Cronto images:

- `GenerateCrontoSign`: This method is used to generate Cronto images with 25 x 25 squares. This type of Cronto image is limited to data with a size of max. 200 hexadecimal characters (i.e. 100 bytes).

  Use the `GenerateCrontoSign` method to generate Cronto images compatible with OneSpan hardware authenticators (e.g. Digipass 760, Digipass 780).

> **NOTE:** This method also exists without the `onPaper` parameter; in that case this parameter is set to `true`.

- `GenerateDynamicCrontoImage`: This method is used to generate Cronto images with a dynamic number of squares. This type of Cronto image is limited to data with a size of max. 1070 hexadecimal characters (i.e. 535 bytes).

> **CAUTION:** The Cronto images generated with the `GenerateDynamicCrontoImage` method cannot be scanned by OneSpan hardware authenticators (e.g. Digipass 760, Digipass 780).

Table 2 lists the relation between the size of the data embedded in a generated image and the number of squares of the generated image.

Table 2: Number of squares in generated images

Data in bytes	Number of squares
0–100	25 x 25
101–199	35 x 35
200–351	45 x 45
352–535	55 x 55

2.3.1 Parameters

Table 3 lists the parameters of the `GenerateCrontoSign` and `GenerateDynamicCrontoImage`methods.

Table 3: `GenerateCrontoSign` / `GenerateDynamicCrontoImage` **parameters**

Parameter name	Data type	Description
`squareSize`	int	The size in pixel of a Cronto image color square. The minimum square size is 1 pixel (`ImageGeneratorSDKConstants.CRONTOSIGN_MIN_SQUARE_SIZE`), the maximum square size is 80 pixels (`ImageGeneratorSDKConstants.CRONTOSIGN_MAX_SQUARE_SIZE`).
`inputValue`	String	The hexadecimal string to be encoded.
`onPaper`	bool	`True` if the image is generated to be printed on paper; in that case, the scanning process is optimized and the detection speed is improved.

2.3.2 Return value

The returned value is a buffered `Image` object in the Java edition and a `bitmap` object in the .NET edition.

For more information about this method, refer to the technical documentation delivered in the Image Generator SDK package.

2.3.3 Additional considerations

To achieve faster results when the end-user scans the image from paper, your images must be printed according to these guidelines:

- Print on matte paper to avoid reflections on the image.
- Print colored images in high contrast to ensure clear and distinct colors. Common color problems arise due to an insufficient difference between blue and green, or blue and black—ensure to have a clear contrast between those colors.
- The closer the paper image colors are to the screen image colors, the better the scanning performance will be.
- Depending on your printer, colors on printed images can differ completely from those displayed on your monitor. This is due to predefined color profiles in your printer configuration which create images with mismatching colors. If your

printer allows it, change or disable this profile.

- The paper used to print the image must not be creased.
- If your printed images are too small, they may not be correctly detected for scanning. Optimal sizes of printed images to be scanned are:
  - 7 x 7 cm for 55 points
  - 6 x 6 cm for 45 points
  - 5 x 5 cm for 35 points, and
  - 4 x 4 cm for 25 points.

For a correct scan of an image, the end user must comply with these guidelines:

- To scan the image, sufficient light is required.
- During the scan, ensure that the camera is placed in front of the image.
- Do not scan too far or too close from the image; the scanning screen must contain the whole content of the image. When scanning the image, ensure that the rendered image on the scanning screen is at least half the size of the shorter side of the screen.

3 Integrate the Image Generator SDK

3.1 Integrate the OneSpan Image Generator SDK in a Java Project 12

3.2 Integrate the OneSpan Image Generator SDK in a .NET Project 13

3.1 Integrate the OneSpan Image Generator SDK in a Java Project

The following procedure allows you to integrate and use the OneSpan Image Generator SDK in your Java project.

▸ To use the OneSpan Image Generator SDK in your Java project

1. Link `ImageGeneratorSDK.jar` from the OneSpan Mobile Security Suite package to your project's `classpath`.
2. Link `UtilitiesSDK.jar` from the OneSpan Mobile Security Suite package to your project's `classpath`.

You are now ready to use the OneSpan Image Generator SDK. For more information how to integrate the functions, see **2 Usage of the Image Generator SDK**.

3.2 Integrate the OneSpan Image Generator SDK in a .NET Project

The following procedure allows you to integrate and use the OneSpan Image Generator SDK in your .NET project.

▸ To use the OneSpan Image Generator SDK in your .NET project

1. Add a reference to `ImageGeneratorSDK.dll` from the OneSpan Mobile Security Suite package to your project.
2. Add a reference to `zxing.dll` from the OneSpan Mobile Security Suite package to your project.

You are now ready to use the OneSpan Image Generator SDK. For more information how to integrate the functions, see **2 Usage of the Image Generator SDK**.