Datalogic Mobile Devices

CUSTOM OCR TEMPLATE

Configuration Manual

DATALOGIC

Datalogic S.r.l.
Via S. Vitalino, 13
40012 Calderara di Reno (BO)
Italy
Tel. +39 051 3147011
Fax +39 051 3147205

Electronic versions of this document may be downloaded from the Datalogic website (www.datalogic.com). If you visit our website and would like to make comments or suggestions about this or other Datalogic publications, please let us know via the "Contact" page.

Disclaimer

Datalogic has taken reasonable measures to provide information in this manual that is complete and accurate, however, Datalogic shall not be liable for technical or editorial errors or omissions contained herein, nor for incidental or consequential damages resulting from the use of this material. Datalogic reserves the right to change any specification at any time without prior notice.

Trademarks

Datalogic and the Datalogic logo are registered trademarks of Datalogic S.p.A. in many countries, including the U.S.A. and the E.U.

TABLE OF CONTENTS

NOTES

INTRODUCTION

OCR SYMBOLOGY OVERVIEW

Datalogic mobile devices include a built-in **OCR (Optical Character Recognition)** symbology designed to read machine-readable text printed in standardized fonts. This functionality is primarily intended for reading official documents such as passports, ID cards, and other travel documents that use OCR-A, OCR-B, or MICR E13B fonts.

The OCR symbology was specifically developed to decode structured documents with predictable layouts and character sequences, making machine reading reliable and accurate for official document processing.

SUPPORTED STANDARD FORMATS

- **TD3 Format (Passport)**: 2 rows of 44 characters each
- **TD1 Format (ID Card)**: 3 rows of 30 characters each
- Standard OCR fonts: OCR-A, OCR-B, and MICR E13B

OCR SYMBOLOGY SETTINGS

The standard OCR symbology (DLSettings → Scanner → Symbology → OCR) provides the following configurable parameters:

Enable: Enables or disables the OCR symbology for reading machine-readable text in standardized fonts.

User ID: Specifies the symbology identifier transmitted with decoded data when code ID transmission is enabled.

Passport Format: Enables reading of passport booklets in TD3 size format (Machine Readable Zone with 2 rows of 44 characters each).

ID Card Format: Enables reading of official travel documents in TD1 size format (Machine Readable Zone with 3 rows of 30 characters each).

Confidence: Sets the threshold in a [0, 100] interval (default 50) used to decide when to accept single character classification results.

This parameter balances reading accuracy versus success rate.

A high confidence value reduces the risk of misreading but increases the probability of a discarded reading. Vice versa, a low value allows to read characters in poor quality images but may result in misreads.

Multiframe: Specifies how many consecutive frames (images) must produce the same OCR result before accepting the decoded data (default 1).

This parameter improves reading stability. For example, when set to 3 frames, the device will only return a decoded paragraph after three consecutive images produce the same result. If a reading differs from the previous one, the device will reset the internal

state, and 3 more images will be needed to decode a paragraph. Once a paragraph is successfully decoded, the internal state resets again and the process restarts to decode the next paragraph.

PARAMETER	DESCRIPTION	TYPE	DEFAULT VALUE
Enable	Enables OCR symbology	Boolean	false
User ID	Symbology identifier	Character	'O'
Passport Format	TD3 format (2 rows × 44 chars)	Boolean	false
ID Card Format	TD1 format (3 rows × 30 chars)	Boolean	false
Confidence	Minimum confidence level (0-100)	Integer (0-100)	50
Multiframe	Matching frames required (1-5)	Integer (1-5)	1

Additional technical documentation is available in the Android SDK Reference.

To support the reading of documents, labels, and forms that use OCR fonts but follow different layouts than the standard TD1/TD3 formats, it's possible to create **Custom OCR Templates**. This feature extends the standard OCR functionality by allowing the definition of custom document reading patterns.

By creating a JSON configuration file, users can specify exact character patterns, document layouts, and validation rules tailored to their specific use cases, including passport variations (e.g., starting with "PMxxx"), industrial labels, custom ID cards, and specialized forms.

CUSTOM OCR TEMPLATES

DEVICE AND SOFTWARE REQUIREMENTS

Custom OCR Templates are supported on Datalogic Android mobile devices including Memor series (Memor 11, Memor 12/17, Memor 30/35 Memor K20), Skorpio X5 series, running firmware that includes Android SDK 1.29 or higher.

For current device support and SDK versions, refer to the Supported Devices documentation.

The Custom OCR Templates configuration file must be named exactly *OCRCustomConfiguration.json* and placed in the /data/media/0/ directory on the device.

After file placement, a device restart is required to load the new configuration templates.

Note: Alternative deployment paths (/enterprise/ and /enterprise/factory/) are reserved for enterprise deployments and require coordination with Datalogic Technical Support.

JSON CONFIGURATION FILE STRUCTURE

The JSON configuration file can contain one or more custom templates, allowing multiple document types to be supported simultaneously. Each template defines the character patterns and validation rules for a specific document format.

Basic Template Structure

json

```
[
  {
    "Name": "Template_Name",
    "Font": 0,
    "Rows": [
      {
        "Content": "PATTERN",
        "Must": 1
      }
    ]
  }
]
```

Multi-Template Configuration

json

```
[
  {
    "Name": " Template_1",
    "Font": 1,
    "Rows": [
      {
        "Content": "BBBBBBBBBB",
        "Must": 1
      }
    ]
  },
  {
    "Name": " Template_2",
    "Font": 0,
    "Rows": [
      {
        "Content": "AAAANNNN",
        "Must": 1
      },
      {
        "Content": "NNNNAAAA",
        "Must": 0
      }
    ]
  }
]
```

CONFIGURATION PARAMETERS REFERENCE

Template-Level Parameters

"Name"

- **Type**: String
- **Required**: Yes
- **Description**: Unique identifier for the template. Appears in device settings under DLSettings → Scanner → Symbology → OCR
- **Example**: "Passport_PMxxx", "Radio_Labels"

"Font"

- **Type**: Integer
- **Required**: Yes
- **Values**:
  - 0: OCR-B font (standard bank/financial documents)
  - 1: OCR-A font (passport/ID standard)
  - 2: MICR E13B font (banking magnetic characters)
- **Performance Note**: OCR-B performs better than OCR-A

Row-Level Parameters

"Content"

- **Type**: String
- **Required**: Yes
- **Description**: Defines the character pattern for each position in the row

CHARACTER	DESCRIPTION	VALID CHARACTERS
N	Number	0-9
A	Letter or <	A-Z, <
B	Number, letter, or <	0-9, A-Z, <
E	Number or <	0-9, <
Y	Special symbols	<, >, +, =
S	Gender symbols	M, F, <
M	Dash	-
<	Literal < character	<
>	Literal > character	>
+	Literal + character	+
X	Number or dash	0-9, -
=	Any character	Any
_	Whitespace	(space character)

"Must"

- **Type**: Integer (0 or 1)
- **Required**: Yes
- **Description**:
  - 1: Row is mandatory for successful decode
  - 0: Row is optional

Advanced Parameters (Optional)

"Mapping"

Used for character-to-number conversion in check digit calculations:

json

```
"Mapping": [
  {
    "OCRValue": "A",
    "MappedValue": 10
  },
  {
    "OCRValue": "B",
    "MappedValue": 11
  }
]
```

"Operations"

Mathematical operations for check digit validation:

json

```
"Operations": ["mod5", "+", "mod2"]
```

Available Operations:

- "+": Sum all input values
  Example: [1,2,3,4,5] → Result: 15
- "*": Multiply all input values
  Example: [1,2,3,4,5] → Result: 120
- "explain_num": Concatenate values
  Example: [1,2,3,4,5] → Result: 12345
- "*X" (where X is integer): Multiply each value by corresponding multiplier
  **Example: "731" with [1,2,3,4,5] → Result: [7,6,3,21,15]*
- "modX" (where X is integer): Apply modulo operation
  Example: "mod3" with [1,2,3,4,5] → Result: [1,2,0,1,2]

"CheckDigit"

Defines check digit location and calculation range:

json

```
"CheckDigit": [
  {
    "Range": [
      {
        "Line": 1,
        "Values": [1, 3]
      }
    ],
    "AffectedIndex": {
      "Line": 1,
      "Values": [9, 10]
    }
  }
]
```

STEP-BY-STEP IMPLEMENTATION GUIDE

STEP 1: ANALYZE YOUR DOCUMENT FORMAT

1. **Identify the font type**: OCR-A, OCR-B, or MICR E13B
2. **Map character positions**: Determine what type of character appears in each position
3. **Identify patterns**: Look for consistent formatting across similar documents
4. **Note special requirements**: Check digits, mandatory vs optional rows.

STEP 2: CREATE THE CONFIGURATION FILE

Example 1: Simple Industrial Label Template

json

```
[
  {
    "Name": "Simple_Label",
    "Font": 1,
    "Rows": [
      {
        "Content": "NNNNNN",
        "Must": 1
      }
    ]
  }
]
```

Purpose: Demonstrates the basic template structure for reading 6-digit numeric labels.

Pattern explanation: "NNNNNN" matches exactly 6 consecutive numbers (000000-999999).

Example 2: Mixed Alphanumeric Pattern

json

```
[
  {
    "Name": "Mixed_Format",
    "Font": 0,
    "Rows": [
      {
        "Content": "AAA_NNN",
        "Must": 1
      }
    ]
  }
]
```

Purpose: Shows how to handle labels with mixed character types and whitespace.

Pattern explanation: "AAA_NNN" matches 3 letters, followed by a space, followed by 3 numbers (e.g., "ABC 123").

Step 3: Deploy the Configuration

1. **Connect device to computer** via USB or use file manager app
2. **Copy file**: Place OCRCustomConfiguration.json in the /data/media/0/ directory
3. **Restart device**: Power cycle or restart to load new configuration
4. **Verify installation**: Check DLSettings → Scanner → Symbology → OCR for new template.

Step 4: Configure Device Settings

1. Open **DLSettings**
2. Navigate to **Scanner → Symbology → OCR**
3. **Enable OCR**: Set to true
4. **Select custom template**: Choose your template from the dropdown
5. **Adjust confidence**: Start with default (50), increase if needed
6. **Set multiframe**: Use 1 for single-frame, increase for increasing redundancy check.

Step 5: Test and Optimize

1. **Test with sample documents**: Verify reading accuracy
2. **Monitor performance**: Check decode speed and success rate
3. **Refine patterns**: Make patterns more specific if possible
4. **Adjust confidence**: Increase for difficult-to-read documents.

OTHER EXAMPLES

Example 1: Custom Passport Variants, excluding check digit calculation

Problem: Documents with non-standard encoding, not reading with standard OCR settings.

Solution - Flexible Approach:

json

```
[
  {
    "Name": "Custom_Passport_Flexible",
    "Font": 1,
    "Rows": [
      {
        "Content": "BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB",
        "Must": 1
      },
      {
        "Content": "BBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB",
        "Must": 1
      }
    ]
  }
]
```

Purpose: Provides maximum flexibility for passport variants by accepting any alphanumeric characters, without check digit calculation.

Pattern explanation: Two rows of 44 "B" characters each, matching any combination of letters, numbers, and '<' symbols typical of passport MRZ format.

Example 2: Equipment Identification Labels

Format: 6 letter prefix + underscore + 1 letter + 3 numbers

- **Real examples**:
- TracPw_N001,
- RailOp_E001,
- FacElc_C001

json

```
[
  {
    "Name": "Equipment_Labels",
    "Font": 1,
    "Rows": [
      {
        "Content": "AAAAAA_ANNN",
        "Must": 1
      }
    ]
  }
]
```

Purpose: Demonstrates specific pattern matching for structured equipment labeling systems.

Pattern explanation: "AAAAAA_ANNN" matches exactly 6 letters, underscore, 1 letter, and 3 numbers - optimized for better recognition than using generic "B" characters.

Example 3: Multi-Row Document with Optional Fields

Use case: Forms with mandatory header and optional footer information.

json

```
[
  {
    "Name": "Form_Processing",
    "Font": 0,
    "Rows": [
      {
        "Content": "NNNN-AAAA-NNNN",
        "Must": 1
      },
      {
        "Content": "AAAAAAAA",
        "Must": 0
      }
    ]
  }
]
```

Purpose: Shows how to handle documents with both mandatory and optional sections.

Pattern explanation: First row "NNNN-AAAA-NNNN" is required and matches format like "1234-ABCD-5678", second row is optional and matches any 8-letter sequence.

Example 4: Advanced Template with Check Digit Validation

Use case: Documents requiring data integrity verification through check digit algorithms.

json

```
[
  {
    "Name": "Validated_Document",
    "Font": 0,
    "Rows": [
      {
        "Content": "AAANNNNNNN",
        "Must": 1
      }
    ],
    "Mapping": [
      {
        "OCRValue": "A",
        "MappedValue": 10
      },
      {
        "OCRValue": "B",
        "MappedValue": 11
      }
    ],
    "Operations": ["+", "mod10"],
    "CheckDigit": [
      {
        "Range": [
          {
            "Line": 1,
            "Values": [1, 9]
          }
        ],
        "AffectedIndex": {
          "Line": 1,
          "Values": [10, 10]
        }
      }
    ]
  }
]
```

Purpose: Demonstrates advanced validation capabilities with character mapping and mathematical operations.

Pattern explanation: "AAANNNNNNN" matches 3 letters + 7 numbers, with positions 1-9 used to calculate and verify the check digit at position 10 using sum and Mod10 operations.

TROUBLESHOOTING & BEST PRACTICES

COMMON ISSUES AND SOLUTIONS

Template Not Appearing in Settings

- **Check file name**: Must be exactly OCRCustomConfiguration.json
- **Verify file location**: Use /data/media/0/ as primary location
- **Restart device**: Power cycle required after file placement
- **Check JSON syntax**: Validate JSON formatting using online tools

Poor Reading Performance

- **Use specific patterns**: Replace generic "B" with specific "N" or "A" where possible
- **Optimize font selection**: OCR-B generally performs better than OCR-A
- **Adjust confidence threshold**: Lower values for difficult documents
- **Increase multiframe**: Use 2-3 for better stability

Inconsistent Reading Results

- **Pattern too restrictive**: Use more flexible patterns if documents vary
- **Pattern too generic**: Make patterns more specific to reduce false positives
- **Check document quality**: Ensure clear, well-printed OCR fonts
- **Verify lighting conditions**: Adequate illumination required

BEST PRACTICES

Pattern Design

1. **Be as specific as possible**: Use exact character types when known
2. **Start broad, then narrow**: Begin with flexible patterns, refine based on testing
3. **Consider variations**: Account for different document versions or wear
4. **Use meaningful names**: Template names should clearly identify the document type

Performance Optimization

1. **Choose appropriate font**:
   - OCR-B for optimal performance
   - OCR-A for passport compatibility
   - MICR E13B only when required
2. **Minimize optional rows**: Too many optional patterns can slow processing
3. **Order templates strategically**: Most common templates should appear first in array
4. **Test thoroughly**: Validate with representative document samples

Maintenance

1. **Version control**: Keep track of configuration changes
2. **Document templates**: Maintain clear documentation of each template's purpose
3. **Regular testing**: Verify continued functionality after firmware updates
4. **Backup configurations**: Keep copies of working configurations

CHARACTER PATTERN GUIDELINES

SCENARIO	RECOMMENDATION	EXAMPLE
Known to be numbers	Use "N"	"NNNNNNNN"
Known to be letters	Use "A"	"AAAAAAAA"
Could be either	Use "B" sparingly	"BBBB" → "AANN" if pattern known
Whitespace	Use "_"	"AAAA_NNNN"
Variable content	Use "=" as last resort	"NNNN===="

FAQ

GENERAL QUESTIONS

Q: Can I have multiple templates in one file?

A: Yes, the JSON file supports an array of templates. Each will appear as a separate option in device settings.

Q: What's the maximum number of templates supported?

A: There's no documented hard limit, but performance may degrade with too many templates. Recommend staying under 20 templates per file.

IMPORTANT NOTE:

Currently, although the decoding libraries allow you to define an unlimited number of templates, you can only activate up to three at a time.

Q: Can templates be used alongside standard passport/ID card formats?

A: Yes, custom templates supplement rather than replace standard OCR formats.

TECHNICAL QUESTIONS

Q: Why choose OCR-B over OCR-A?

A: OCR-B is optimized for machine reading and generally provides better performance. Use OCR-A only when required for specific application compatibility.

Q: How do I handle documents with varying lengths?

A: Use the "=" character for variable-length sections or create multiple templates for different document variations.

DEPLOYMENT QUESTIONS

Q: Can I update templates without restarting?

A: No, the device must be restarted for configuration changes to take effect.

Q: What happens if the JSON file is corrupted?

A: Corrupted files are ignored. No custom templates will be available, but standard OCR functionality remains unaffected.

PERFORMANCE QUESTIONS

Q: How do I improve reading speed?

A: Use specific character patterns, minimize optional rows, choose OCR-B font when possible, and optimize confidence settings.

Q: What if documents have poor print quality?

A: Lower the confidence threshold, increase multiframe count, and ensure adequate lighting during scanning.

Q: Can I combine Custom OCR with other scanning modes?

A: Yes, Custom OCR works alongside barcode scanning and SnapOCR functionality.

SUPPORT AND RESOURCES

For additional assistance:

- **Technical Documentation**: developer.datalogic.com
- **Discussion Forum**: discussion.datalogic.com
- **OEMConfig Reference**: OEMConfig Scanner Settings

This manual was prepared by the Datalogic Mobile Products L3 Specialist team. For updates and additional technical resources, visit the Datalogic Developer Portal.

DATALOGIC

www.datalogic.com

Datalogic S.r.l.
Via S. Vitalino, 13
40012 Calderara di Reno (BO)
Italy
Tel. +39 051 3147011
Fax +39 051 3147205

October 2025