PROGRAMMER INTERFACE GUIDE

10.1" Android Display | USB Serial Protocol

Bonrix Software Systems
www.bonrix.co.in

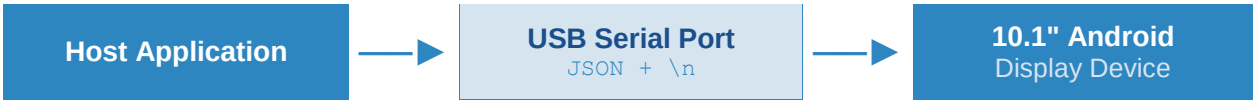
Document Type	API / Integration Reference
Device	10.1 Inch Android Customer Display
Protocol	USB Serial (Web Serial API)
Command Format	JSON over serial line
Version	1.0
Interface URL	https://www.bonrix.co.in/pos-device-controller.html

1. Overview

This document is the complete integration reference for connecting a host application (browser, desktop, or server) to the Bonrix 10.1-inch Android Customer Display via USB Serial communication. The device accepts JSON-formatted command packets that control the display content — including totals, item listings, QR codes, receipt printing, and Wi-Fi management.

1.1 Architecture

The host sends plain-text JSON commands terminated with a newline character (\n) over a USB serial port. The Android display parses each complete JSON line and renders the appropriate screen.



2. Device Connection

2.1 Hardware Setup

Connect the Bonrix 10.1-inch Android Display to the host machine using a standard USB-A to USB-B (or USB-C, depending on device variant) cable. The device will enumerate as a USB CDC (Communications Device Class) serial port — no additional driver installation is required on modern Windows 10+, macOS, or Linux systems.

USB Class	CDC-ACM (Abstract Control Model)
Connector	USB Type-A (Host) ↔ USB Type-C or Micro-USB (Device)
Power	Powered via its own AC adapter; USB is data-only
OS Support	Windows 10+, macOS 12+, Linux (kernel ≥ 3.x), ChromeOS
Driver	Built-in OS CDC driver (no manual install needed)

2.2 Serial Port Parameters

Configure the serial port with the following parameters before sending commands. The recommended baud rate is 115200 for lowest latency:

Baud Rate	9600 / 19200 / 38400 / 57600 / 115200 (default: 115200)
Data Bits	8
Stop Bits	1
Parity	None
Flow Control	None (RTS/CTS disabled)
Line Ending	LF (\n, 0x0A) — each JSON command terminated with newline
Encoding	UTF-8

Important: Every JSON command MUST end with a newline character (\n / 0x0A). The device uses line-by-line parsing; commands without a newline will be buffered indefinitely.

2.3 Web Serial API (Browser)

The Bonrix POS Controller page (pos-device-controller.html) uses the browser's Web Serial API. If you are integrating directly in a browser application, use the following pattern:

```
// 1. Request port (triggers browser permission prompt)
const port = await navigator.serial.requestPort();

// 2. Open with desired baud rate
await port.open({ baudRate: 115200 });
```

```
// 3. Get a writer
const writer = port.writable.getWriter();

// 4. Encode & send a command
const encoder = new TextEncoder();
const payload = JSON.stringify(command) + '\n';
await writer.write(encoder.encode(payload));

// 5. Release writer when done
writer.releaseLock();
```

 Browser Support: Web Serial API is supported in Chrome 89+, Edge 89+, and Opera 76+. It is NOT available in Firefox or Safari. HTTPS or localhost is required.

2.4 Native / Desktop Integration

For Python, Java, Node.js, or any native application, use any standard serial library. The device appears as a COM port (Windows) or /dev/ttyUSB* or /dev/ttyACM* (Linux/macOS).

```
# Python example using pyserial
import serial, json

ser = serial.Serial('/dev/ttyUSB0', baudrate=115200,
                    bytesize=8, parity='N', stopbits=1,
                    timeout=1)

def send_command(cmd_dict):
    payload = json.dumps(cmd_dict) + '\n'
    ser.write(payload.encode('utf-8'))
```

2.5 Auto-Connect & Reconnect

The Bonrix POS Controller page supports two convenience modes:

- Auto-connect on page load — automatically opens the last-used serial port when the page is opened.
- Auto-reconnect on disconnect — if the USB cable is unplugged and re-plugged, the application automatically attempts to reopen the port.

For custom integrations, implement a connection-check loop (poll port.readable for errors) and re-call port.open() on failure.

2.6 General Command Structure

All commands follow a common JSON envelope. The cmd field acts as a command discriminator, and all other fields are command-specific parameters.

```
{
  "cmd": "<COMMAND_NAME>",
```

```
  // ... command-specific fields
}
```

Field	Type	Required	Description
cmd	string	Required	Command identifier — must exactly match the values listed in Section 3.

 Tip: Field names and cmd values are case-sensitive. Send the JSON as a single line with no embedded newlines — pretty-printed JSON (multi-line) will NOT be parsed correctly.

3. Command Reference

The following table summarises all supported commands. Detailed payloads and field descriptions follow for each command.

#	cmd Value	Purpose
01	`display_total`	Show running cart total and amount summary on the display
02	`submit_bill`	Finalise the bill; show grand total and thank-you message
03	`single_item`	Display a single scanned or selected line item with price
04	`welcome`	Show the idle / welcome screen with branding
05	`qr_code`	Display a QR code for UPI / payment link scanning
06	`qr_success`	Show payment success confirmation with amount
07	`qr_fail`	Show payment failure screen with retry prompt
08	`qr_pending`	Show payment pending / processing animation
09	`qr_cancel`	Show payment cancelled status screen
10	`print_receipt`	Trigger receipt printing on an attached receipt printer
11	`reset_wifi`	Reset / reconfigure the device Wi-Fi network settings
12	`custom`	Send arbitrary display content for custom use cases

3.1 Display Total

CMD-01 display_total

Updates the display with the current cart subtotal, discount, tax, and grand total. Call this command each time an item is added or removed from the cart to keep the customer display in sync.

JSON Payload:

```
{
  "cmd": "display_total",
  "subtotal": "₹1,250.00",
  "discount": "₹50.00",
  "tax": "₹112.50",
  "total": "₹1,312.50",
  "items_count": 5,
  "currency": "INR"
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "display_total"
subtotal	string	Required	Formatted subtotal before discount/tax (e.g., "₹1,250.00")
discount	string	Optional	Formatted discount amount; omit or pass "" if none
tax	string	Optional	Formatted tax / GST amount; omit or pass "" if none
total	string	Required	Final payable amount after all adjustments
items_count	integer	Optional	Number of line items in the cart (displayed as badge)
currency	string	Optional	ISO 4217 currency code, e.g. "INR", "USD" (default: INR)

Note: Format currency values as human-readable strings with the symbol; the device does not perform arithmetic.

3.2 Submit Bill

CMD-02 submit_bill

Presents the finalised bill screen. This should be sent after the cashier completes item entry and clicks 'Submit'. The display shows a thank-you message, grand total, and payment mode.

JSON Payload:

```
{
  "cmd": "submit_bill",
  "bill_no": "INV-20240318-0042",
  "total": "₹1,312.50",
  "payment_mode": "UPI",
  "cashier": "Ravi Kumar",
  "items_count": 5,
  "thank_you_msg": "Thank you for shopping with us!"
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "submit_bill"
bill_no	string	Optional	Invoice / bill reference number for display
total	string	Required	Grand total amount payable
payment_mode	string	Optional	Payment method label: "Cash", "UPI", "Card", etc.
cashier	string	Optional	Cashier / operator name for the bill footer
items_count	integer	Optional	Total number of items in the bill
thank_you_msg	string	Optional	Custom thank-you message (max ~80 chars)

Note: After submit_bill is displayed, follow up with either a qr_code command (if UPI) or print_receipt to complete the transaction flow.

3.3 Single Item

CMD-03 single_item

Displays a single product or scanned item with name, quantity, unit price, and line total. This is typically triggered immediately after a barcode scan or product selection.

JSON Payload:

```
{
  "cmd": "single_item",
  "item_name": "Amul Full Cream Milk 1L",
  "item_code": "8901063021013",
  "qty": 2,
  "unit": "Pcs",
  "unit_price": "₹68.00",
  "line_total": "₹136.00",
  "mrp": "₹72.00",
  "discount_pct": "5.6%"
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "single_item"
item_name	string	Required	Product / item display name
item_code	string	Optional	Barcode or SKU code; shown in smaller text below name
qty	number	Required	Quantity (integer or decimal, e.g. 1.5 for weight items)
unit	string	Optional	Unit label: "Pcs", "Kg", "L", etc.
unit_price	string	Required	Formatted price per unit
line_total	string	Required	Formatted total for this line (qty × unit_price)
mrp	string	Optional	Maximum Retail Price; shown crossed-out if provided
discount_pct	string	Optional	Discount percentage label, e.g. "5.6%"

Note: For weight-based items, pass qty as a decimal (e.g., 0.850) and set unit to "Kg".

3.4 Welcome

CMD-04 welcome

Returns the display to its idle/welcome state. This should be called after a transaction completes or when the POS is idle, so the customer-facing screen shows the store branding.

JSON Payload:

```
{
  "cmd": "welcome",
  "store_name": "Bonrix Retail",
  "tagline": "Quality you can trust",
  "logo_url": "http://192.168.1.10/logo.png",
  "bg_color": "#1B4F8A",
  "text_color": "#FFFFFF"
}
```

Field Reference:

Field	Type	Required	Description
`cmd`	string	Required	Must be "welcome"
`store_name`	string	Optional	Store / brand name shown prominently
`tagline`	string	Optional	Short tagline or slogan below the store name
`logo_url`	string	Optional	HTTP URL of logo image accessible on local network; omit for no logo
`bg_color`	string	Optional	Background hex color (default: device theme color)
`text_color`	string	Optional	Text hex color (default: white)

Note: The welcome screen typically auto-displays after a configurable idle timeout on the device itself. Sending this command forces an immediate return to welcome.

3.5 QR Code

CMD-05 qr_code

Renders a scannable QR code on the display. Primarily used for UPI payment requests. The customer scans the QR code using their payment app (GPay, PhonePe, BHIM, etc.).

JSON Payload:

```
{
  "cmd": "qr_code",
  "qr_data": "upi://pay?pa=bonrix@ybl&pn=Bonrix+Retail&am=1312.50&cu=INR",
  "amount": "₹1,312.50",
  "label": "Scan to Pay via UPI",
  "upi_id": "bonrix@ybl",
  "timeout_sec": 120
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "qr_code"
qr_data	string	Required	Full content to encode in the QR (UPI URI, URL, text, etc.)
amount	string	Optional	Formatted amount shown below the QR code
label	string	Optional	Instruction text shown above the QR code
upi_id	string	Optional	UPI VPA shown as text for manual entry fallback
timeout_sec	integer	Optional	Seconds before the QR auto-expires; 0 = no timeout

Note: Construct the qr_data value using the standard UPI deep-link format: upi://pay?pa=VPA&pn=NAME&am=AMOUNT&cu=INR

3.6 QR Success

CMD-06 qr_success

Displays a payment success confirmation screen (green check mark animation). Send this command after your backend payment gateway confirms a successful UPI transaction.

JSON Payload:

```
{
  "cmd": "qr_success",
  "amount": "₹1,312.50",
  "transaction_id": "TXN4293847265",
  "upi_ref": "UPI-REF-2024-83726",
  "message": "Payment Received Successfully",
  "paid_by": "GPay"
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "qr_success"
amount	string	Required	Confirmed paid amount
transaction_id	string	Optional	Internal transaction / order ID
upi_ref	string	Optional	UPI Reference number from payment gateway
message	string	Optional	Custom success message (default: "Payment Successful")
paid_by	string	Optional	Payment app or method used: "GPay", "PhonePe", etc.

Note: After displaying success, send a welcome command after a short delay (3–5 seconds) to reset the screen for the next customer.

3.7 QR Fail

CMD-07 qr_fail

Shows a payment failure screen (red X animation) with an error reason. Send when the payment gateway returns a failure or timeout.

JSON Payload:

```
{
  "cmd": "qr_fail",
  "amount": "₹1,312.50",
  "reason": "Payment declined by bank",
  "error_code": "U30",
  "retry": true
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "qr_fail"
amount	string	Optional	Amount that was attempted
reason	string	Optional	Human-readable failure reason shown to customer
error_code	string	Optional	Gateway error code for reference
retry	boolean	Optional	If true, device shows a "Please try again" prompt

 Note: After qr_fail, either re-send qr_code to allow a retry, or send qr_cancel followed by welcome to abandon the payment.

3.8 QR Pending

CMD-08 qr_pending

Displays a "Payment Processing" animation while the backend is waiting for payment gateway confirmation. Keep showing this until a success or failure response is received.

JSON Payload:

```
{
  "cmd": "qr_pending",
  "amount": "₹1,312.50",
  "message": "Processing your payment...",
  "timeout_sec": 30
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "qr_pending"
amount	string	Optional	Amount being processed
message	string	Optional	Animated status message (default: "Processing...")
timeout_sec	integer	Optional	Auto-transition to qr_fail after this many seconds if no update received

3.9 QR Cancel

CMD-09 qr_cancel

Displays a payment cancelled status screen. Send when the cashier cancels the transaction, the QR code expires, or the customer requests cancellation.

JSON Payload:

```
{
  "cmd": "qr_cancel",
  "amount": "₹1,312.50",
  "reason": "Cancelled by cashier",
  "message": "Transaction Cancelled"
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "qr_cancel"
amount	string	Optional	Amount that was being processed
reason	string	Optional	Reason code or description
message	string	Optional	Display message (default: "Transaction Cancelled")

Note: After displaying qr_cancel, send welcome to return to idle state.

3.10 Print Receipt

CMD-10 print_receipt

Triggers receipt printing on a receipt printer connected to the Android display device (via USB OTG or Bluetooth). The receipt content is passed as a structured array of line items.

JSON Payload:

```
{
  "cmd": "print_receipt",
  "store_name": "Bonrix Retail",
  "store_address": "123 MG Road, Surat, Gujarat 395001",
  "bill_no": "INV-20240318-0042",
  "date": "18-03-2024 14:32:05",
  "cashier": "Ravi Kumar",
  "items": [
    { "name": "Amul Milk 1L",    "qty": 2, "rate": "68.00",  "total":
"136.00" },
    { "name": "Parle-G Biscuit", "qty": 3, "rate": "10.00",  "total": "30.00"
},
    { "name": "Colgate 200g",    "qty": 1, "rate": "120.00", "total":
"120.00" }
  ],
  "subtotal": "286.00",
  "discount": "14.00",
  "tax_lines": [
    { "label": "GST 5%", "amount": "13.60" }
  ],
  "grand_total": "285.60",
  "payment_mode": "UPI",
  "footer_msg": "Thank you! Visit again.",
  "print_copies": 1
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "print_receipt"
store_name	string	Optional	Printed in header of receipt
store_address	string	Optional	Store address line(s)
bill_no	string	Required	Bill / invoice number
date	string	Required	Transaction date-time string
cashier	string	Optional	Cashier / operator name
items	array	Required	Array of line item objects (see sub-fields below)
items[].name	string	Required	Product name
items[].qty	number	Required	Quantity
items[].rate	string	Required	Unit rate (no symbol)
items[].total	string	Required	Line total (no symbol)

Field	Type	Required	Description
`subtotal`	string	Required	Pre-tax, pre-discount subtotal
`discount`	string	Optional	Discount amount
`tax_lines`	array	Optional	Array of {label, amount} tax breakdown objects
`grand_total`	string	Required	Final payable amount
`payment_mode`	string	Optional	"Cash", "UPI", "Card", etc.
`footer_msg`	string	Optional	Custom footer message at bottom of receipt
`print_copies`	integer	Optional	Number of receipt copies (default: 1)

 Note: Amounts in the items array and totals should be plain numeric strings without currency symbols (e.g., "68.00" not "₹68.00") to allow the printer formatter to align columns correctly.

3.11 Reset Wi-Fi

CMD-11 reset_wifi

Triggers a Wi-Fi reset or reconfiguration on the Android display device. Optionally provides new SSID and password credentials for the device to connect to a different network.

JSON Payload:

```
{
  "cmd": "reset_wifi",
  "action": "reconnect",
  "ssid": "BonrixStore_5G",
  "password": "StorePass@2024",
  "security": "WPA2"
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "reset_wifi"
action	string	Required	"reconnect" – re-join current network; "configure" – apply new credentials; "forget" – clear saved network
ssid	string	Optional	Wi-Fi network name (required when action is "configure")
password	string	Optional	Wi-Fi password (required when action is "configure")
security	string	Optional	Security type: "WPA2" (default), "WPA3", "WEP", "OPEN"

Note: Use action: "reconnect" for a soft Wi-Fi restart without changing credentials. Use action: "configure" only when deploying the device to a new network. The device will reboot Wi-Fi and display its new IP address on screen.

3.12 Custom

CMD-12 custom

Sends a fully custom payload to the display. Use this for promotional messages, advertisements, announcements, or any screen not covered by the built-in command set. Supports text, background color, and optionally an image URL.

JSON Payload:

```
{
  "cmd": "custom",
  "title": "□ Weekend Sale!",
  "body": "Flat 20% OFF on all Electronics\nOffer valid till Sunday
midnight",
  "image_url": "http://192.168.1.10/promo.png",
  "bg_color": "#F39C12",
  "text_color": "#FFFFFF",
  "title_size": "large",
  "duration_sec": 10
}
```

Field Reference:

Field	Type	Required	Description
cmd	string	Required	Must be "custom"
title	string	Optional	Large headline text (supports emoji)
body	string	Optional	Body text; use \n for line breaks
image_url	string	Optional	HTTP URL to display image (must be LAN-accessible)
bg_color	string	Optional	Background hex color (e.g., "#F39C12")
text_color	string	Optional	Text hex color (e.g., "#FFFFFF")
title_size	string	Optional	Title font size: "small", "medium" (default), "large"
duration_sec	integer	Optional	Auto-dismiss after N seconds; 0 = stays until next command

□ **Note:** Image URLs must be reachable from the device's network. Use a local web server or file server on the same LAN. External internet URLs may not be accessible depending on the store network configuration.

4. Typical Transaction Flow

The following sequence illustrates a standard UPI checkout transaction from start to finish:

Step	Command Sent	Display Result
1	`welcome`	Idle screen with store branding
2	`single_item`	Item scan: name, qty, price displayed instantly
3	`display_total`	Running cart total updated on each scan
4	`submit_bill`	Bill finalised, grand total shown
5	`qr_code`	UPI QR code shown for customer to scan
6	`qr_pending`	"Processing..." animation while waiting for gateway
7a	`qr_success`	Green confirmation with amount and transaction ID
7b	`qr_fail`	Red failure screen with reason; cashier retries
8	`print_receipt`	Receipt printed on attached printer
9	`welcome`	Reset to idle for next customer

5. Error Handling & Best Practices

5.1 Common Issues

Device not found	Ensure the USB cable is data-capable (not charge-only). Try a different USB port. On Linux, add user to 'dialout' group: sudo usermod -aG dialout $USER
Permission denied (Linux)	Run: sudo chmod 666 /dev/ttyUSB0 OR add udev rule for the device's USB VID:PID
Command not displayed	Verify the JSON ends with \n. Validate JSON syntax (no trailing commas, all keys quoted). Check baud rate matches device setting.
Garbled characters	Baud rate mismatch. Confirm both host and device are at the same rate (default 115200).
QR code not readable	Ensure qr_data is a valid URI and not URL-encoded twice. Test the QR string directly in a QR generator.
Image not loading	image_url and logo_url must be reachable from the device's own network IP. Avoid 'localhost' — use the host machine's LAN IP.

5.2 Best Practices

1. Always terminate JSON with \n (0x0A). Never send multi-line formatted JSON.

2. Validate JSON on the host side before sending to avoid the device silently ignoring malformed packets.
3. Send display_total after every cart modification to keep the customer display in sync with the POS.
4. Implement a 3–5 second delay after qr_success or qr_fail before sending welcome, to allow the customer to read the confirmation screen.
5. Use the lowest adequate baud rate if the USB cable run is long or you experience noise; 115200 is preferred for short (< 3m) cables.
6. For production deployments, log every command sent with a timestamp for debugging transaction disputes.
7. Do not send commands faster than 50ms apart; give the device time to render each screen.

6. Quick Reference Card

cmd Value	Minimum Required Fields
display_total	cmd, subtotal, total
submit_bill	cmd, total
single_item	cmd, item_name, qty, unit_price, line_total
welcome	cmd
qr_code	cmd, qr_data
qr_success	cmd, amount
qr_fail	cmd
qr_pending	cmd
qr_cancel	cmd
print_receipt	cmd, bill_no, date, items[ ], subtotal, grand_total
reset_wifi	cmd, action
custom	cmd (plus at least one of: title, body, image_url)

Support: For integration support, visit www.bonrix.co.in or contact the Bonrix development team.