Push Payments - Android - QR Parser Technical Documentation

MVisaQRParser-[version].zip

The MVisaQRParser zip file contains MVisaQRParser-[version].jar along with change log and documentation

mVisa QR Parser is a jar parses the input tlv string and returns the values in the form of a POJO or HashMap or JSON.

Steps to integrate MVisaQRParser

1. Add the MVisaQRParser.jar to the libs folder (app/libs) of the app module.
2. Add the following block to app module's **build.gradle** file to define the location of the Jars.

```
repositories{
   flatDir{
      dirs 'libs'
   }
}
```

3. Add the following line under dependencies block in the app module's **build.gradle** file to add the visa dependencies.

```
compile fileTree(dir: 'libs', include: ['*.jar'])
```

4. Add following snippet in proguard.

```
-keep class com.visa.mvisa.** {
    *;

}
```

5. Gradle sync & build and start using the SDK APIs.

```
String tlvStringNew = "000201010211020847613617520460105303356580 2IN5912Corner
Store6006Mumbai61064011046219010 6billid0505refid6304CA4C";


String tlvStringOld = "08476136171LGrace Hopper Merchant24581237Kolkata42IN53356";

//Initializing the parser
 QrCodeParser qrCodeParser = new QrCodeParser();



//Method 1 :Calling the sdk to get the respnse as json.
String qrCodeParserResponse = new QrCodeParser().parseQrDataAsJson(tlvString);
Log.i(QrParser.class.getSimpleName(), "JSON RESPONSE : " + qrCodeParserResponse);


//Method 2: passing the tlv input to the parser and getting the response as object
QrCodeParserResponse qrCodeParserResponse = qrCodeParser.parseQrData(tlvStringNew);
if (qrCodeParserResponse != null) {

 //Checking if error codes list is not empty
     if(qrCodeParserResponse.getQrCodeError() != null && !qrCodeParserResponse.getQrCodeError().
isEmpty()) {

        Log.i(QrParser.class.getSimpleName(), "Error Codes : " + qrCodeParserResponse.
getQrCodeError());

    } else if (qrCodeParserResponse.getQrCodeData() != null) { //Use QRCodeData pojo to access
each item from tlv - deprecated


 //if error code list is empty, then getting the value from the parsed response and printing
         Log.i(QrParser.class.getSimpleName(), "mVisaMerchant Pan : "qrCodeParserResponse.
getQrCodeData().getmVisaMerchantPan());

    } else if (qrCodeParserResponse.getQrCodeDataMap() != null { //Use qrCodeDataMap to access
each item from tlv - recommended

         //getQrCodeDataMap return a HashMap(Key value pair) containing all values of tlv in
hashmap.

         //Use QRCodeTag enum's method QRCodeTag.tagCode() to get the key for accessing data
items.

         //For subtags, the hashmap will have nested HashMap. Use QRCodeTag.SubTag.tagCode()
method for subtag keys.

         Log.i(QrParser.class.getSimpleName(), "mVisaMerchant Pan : "qrCodeParserResponse.
getQrCodeDataMap().get(QRCodeTag.MERCHANT.tagCode()));

}

)

}
```

MVisaQRParser API details

The following describes the methods used in the SDK with their use cases, input & output parameters.

API Methods

USE CASE	ENTRY POINT METHOD	INPUT PARAMETERS	OUTPUT PARAMETERS
Parse the QR code data string	QrCodeParserResponse parseQrData(String)	The raw tlvString read from the QR Code	QrCodeParserResponse object – Model object contaning all the data present in the input tlv string/list of error codes
Parse the QR code data string into a JSON String	String parseQrDataAsJson(String qrCodeData)	The raw tlvString read from QR Code	String qrCodeData – Qr code data containing all the present in the input tlv/list of error codes.

API Objects

The following describes the objects used by the methods in the SDK with their type, purposes and where they are used.

QrCodeParserResponse

ITEM	DESCRIPTION	FORMAT	METHODS
QrCodeData qrCodedata	Model object containing all the data present in the input tlv	CONDITIONAL Will be null in case of error in parsing	Getter: QrCodeData getQrCodeData()
List<Integer> qrCodeError	List of error codes found while parsing the QR Code tlv string	CONDITIONAL Will be null in case of no errors found.	Getter List<Integer> getQrCodeError()
HashMap<String, Object>	Key value pair containing all data present in input tlv	CONDITIONAL Will be null in case of error in parsing	Getter: HashMap<String, Object> getQrCodeDataMap()

*Note – Both qrCodeData and qrCodeError cannot be null at the same time.

QRCodeTag enum

This enum represent the key for HashMap response which you receive. It has two methods tagCode() and tagName()

ITEM	DESCRIPTION	Getter Method
String tagCode	Returns the numerical tagCode which should be used to access an item from the HashMap response	tagCode()
String tagName	Returns the tagName which can be use to access an item from the JSON response	tagName()

Enum Values

```
//ENUM("tagName", "tagCode");

PAYLOAD_FORMAT_INDICATOR("payloadFormatIndicator", "00"),
POINT_OF_INITIATION("pointOfInitiation", "01"),

MERCHANT_ID("mVisaMerchantId", "02"),
MERCHANT_PAN("mVisaMerchantPan", "02"),

MASTER_CARD_PAN_1("masterCardPan1", "04"),
MASTER_CARD_PAN_2("masterCardPan2", "05"),
NPCIID1("npciid1", "06"),
NPCIID2("npciid2", "07"),
TAG_08("tag08", "08"),
DISCOVER_1("discover1", "09"),
DISCOVER_2("discover2", "10"),
AMEX_1("amex1", "11"),
AMEX_2("amex2", "12"),
JCB_1("jcb1", "13"),
JCB_2("jcb2", "14"),
UNION_PAY_1("unionPay1", "15"),
UNION_PAY_2("unionPay2", "16"),
TAG_17("tag17", "17"),
TAG_18("tag18", "18"),
TAG_19("tag19", "19"),
TAG_20("tag20", "20"),
TAG_21("tag21", "21"),
TAG_22("tag22", "22"),
TAG_23("tag23", "23"),
TAG_24("tag24", "24"),
TAG_25("tag25", "25"),
TAG_26("tag26", "26"),
TAG_27("tag27", "27"),
TAG_28("tag28", "28"),
TAG_29("tag29", "29"),
TAG_30("tag30", "30"),
TAG_31("tag31", "31"),
TAG_32("tag32", "32"),
```

```
TAG_33("tag33", "33"),
TAG_34("tag34", "34"),
TAG_35("tag35", "35"),
TAG_36("tag36", "36"),
TAG_37("tag37", "37"),
TAG_38("tag38", "38"),
TAG_39("tag39", "39"),
TAG_40("tag40", "40"),
TAG_41("tag41", "41"),
TAG_42("tag42", "42"),
TAG_43("tag43", "43"),
TAG_44("tag44", "44"),
TAG_45("tag45", "45"),
TAG_46("tag46", "46"),
TAG_47("tag47", "47"),
TAG_48("tag48", "48"),
TAG_49("tag49", "49"),
TAG_50("tag50", "50"),
TAG_51("tag51", "51"),
MERCHANT_CATEGORY_CODE("merchantCategoryCode", "52"),
CURRENCY_CODE("currencyCode", "53"),
TRANSACTION_AMOUNT("transactionAmount", "54"),
CONVENIENCE_FEE_INDICATOR("convenienceFeeIndicator", "55"),
CONVENIENCE_FEE_AMOUNT("convenienceFeeAmount", "56"),
CONVENIENCE_FEE_PERCENTAGE("convenienceFeePercentage", "57"),
COUNTRY_CODE("countryCode", "58"),
MERCHANT_NAME("merchantName", "59"),
CITY_NAME("cityName", "60"),
POSTAL_CODE("postalCode", "61"),
ADDITIONAL_DATA_FIELD_TEMPLATE("additionalDataFieldTemplate", "62"),
CRC("crc", "63"),
MERCHANT_INFORMATION_LANGUAGE_TEMPLATE("merchantInformationLanguageTemplate", "64"),
TAG_65("tag65", "65"),
TAG_66("tag66", "66"),
TAG_67("tag67", "67"),
TAG_68("tag68", "68"),
TAG_69("tag69", "69"),
TAG_70("tag70", "70"),
TAG_71("tag71", "71"),
TAG_72("tag72", "72"),
TAG_73("tag73", "73"),
TAG_74("tag74", "74"),
TAG_75("tag75", "75"),
TAG_76("tag76", "76"),
TAG_77("tag77", "77"),
TAG_78("tag78", "78"),
TAG_79("tag79", "79"),
TAG_80("tag80", "80"),
TAG_81("tag81", "81"),
TAG_82("tag82", "82"),
TAG_83("tag83", "83"),
TAG_84("tag84", "84"),
TAG_85("tag85", "85"),
TAG_86("tag86", "86"),
TAG_87("tag87", "87"),
TAG_88("tag88", "88"),
TAG_89("tag89", "89"),
TAG_90("tag90", "90"),
TAG_91("tag91", "91"),
TAG_92("tag92", "92"),
TAG_93("tag93", "93"),
TAG_94("tag94", "94"),
TAG_95("tag95", "95"),
TAG_96("tag96", "96"),
TAG_97("tag97", "97"),
TAG_98("tag98", "98"),
TAG_99("tag99", "99");
```

QRCodeTag.SubTag enum

This enum is used to access subtag elements in the HashMap repsonse

ITEM	DESCRIPTION	Getter Method

String tagCode	Returns the numerical tagCode which should be used to access an item from the HashMap response	tagCode()
String tagName	Returns the tagName which can be use to access an item from the JSON response	tagName()

Enum Values

```
SUB_TAG_00("subTag00", "00"),
ALTERNATE_LANGUAGE("alternateLanguage", "00"),

SUB_TAG_01("subTag01", "01"),
MERCHANT_NAME_ALTERNATE_LANGUAGE("merchantNameAlternateLanguage", "01"),
BILL_ID("billID", "01"),

SUB_TAG_02("subTag01", "02"),
MERCHANT_CITY_ALTERNATE_LANGUAGE("cityAlternateLanguage", "02"),
MOBILE_NUMBER("mobileNumber", "02"),

SUB_TAG_03("subTag03", "03"),
STORE_ID("storeID", "03"),

SUB_TAG_04("subTag04", "04"),
LOYALTY_NUMBER("loyaltyNumber", "04"),

SUB_TAG_05("subTag05", "05"),
REFERENCE_ID("referenceID", "05"),

SUB_TAG_06("subTag06", "06"),
CONSUMER_ID("consumerID", "06"),

SUB_TAG_07("subTag07", "07"),
TERMINAL_ID("terminalID", "07"),

SUB_TAG_08("subTag08", "08"),
PURPOSE("purpose", "08"),

SUB_TAG_09("subTag09", "09"),
ADDITIONAL_CONSUMER_DATA_REQUEST("additionalConsumerDataRequest", "09"),

SUB_TAG_10("subTag10", "10"),

SUB_TAG_11("subTag11", "11"),
ADD_DATA_MASTER_CARD_1("addDataMasterCard1", "11"),

SUB_TAG_12("subTag12", "12"),
ADD_DATA_MASTER_CARD_2("addDataMasterCard2", "12"),

SUB_TAG_13("subTag13", "13"),
ADD_DATA_NPCI_1("addDataNpci1", "13"),

SUB_TAG_14("subTag14", "14"),
ADD_DATA_NPCI_2("addDataNpci2", "14"),

SUB_TAG_15("subTag15", "15"),
SUB_TAG_16("subTag16", "16"),
SUB_TAG_17("subTag17", "17"),
SUB_TAG_18("subTag18", "18"),
SUB_TAG_19("subTag19", "19"),
SUB_TAG_20("subTag20", "20"),
SUB_TAG_21("subTag21", "21"),
SUB_TAG_22("subTag22", "22"),
SUB_TAG_23("subTag23", "23"),
SUB_TAG_24("subTag24", "24"),
SUB_TAG_25("subTag25", "25"),
SUB_TAG_26("subTag26", "26"),
SUB_TAG_27("subTag27", "27"),
SUB_TAG_28("subTag28", "28"),
SUB_TAG_29("subTag29", "29"),
SUB_TAG_30("subTag30", "30"),
SUB_TAG_31("subTag31", "31"),
SUB_TAG_32("subTag32", "32"),
SUB_TAG_33("subTag33", "33"),
SUB_TAG_34("subTag34", "34"),
```

```
SUB_TAG_35("subTag35", "35"),
SUB_TAG_36("subTag36", "36"),
SUB_TAG_37("subTag37", "37"),
SUB_TAG_38("subTag38", "38"),
SUB_TAG_39("subTag39", "39"),
SUB_TAG_40("subTag40", "40"),
SUB_TAG_41("subTag41", "41"),
SUB_TAG_42("subTag42", "42"),
SUB_TAG_43("subTag43", "43"),
SUB_TAG_44("subTag44", "44"),
SUB_TAG_45("subTag45", "45"),
SUB_TAG_46("subTag46", "46"),
SUB_TAG_47("subTag47", "47"),
SUB_TAG_48("subTag48", "48"),
SUB_TAG_49("subTag49", "49"),
SUB_TAG_50("subTag50", "50"),
SUB_TAG_51("subTag51", "51"),
SUB_TAG_52("subTag52", "52"),
SUB_TAG_53("subTag53", "53"),
SUB_TAG_54("subTag54", "54"),
SUB_TAG_55("subTag55", "55"),
SUB_TAG_56("subTag56", "56"),
SUB_TAG_57("subTag57", "57"),
SUB_TAG_58("subTag58", "58"),
SUB_TAG_59("subTag59", "59"),
SUB_TAG_60("subTag60", "60"),
SUB_TAG_61("subTag61", "61"),
SUB_TAG_62("subTag62", "62"),
SUB_TAG_63("subTag63", "63"),
SUB_TAG_64("subTag64", "64"),
SUB_TAG_65("subTag65", "65"),
SUB_TAG_66("subTag66", "66"),
SUB_TAG_67("subTag67", "67"),
SUB_TAG_68("subTag68", "68"),
SUB_TAG_69("subTag69", "69"),
SUB_TAG_70("subTag70", "70"),
SUB_TAG_71("subTag71", "71"),
SUB_TAG_72("subTag72", "72"),
SUB_TAG_73("subTag73", "73"),
SUB_TAG_74("subTag74", "74"),
SUB_TAG_75("subTag75", "75"),
SUB_TAG_76("subTag76", "76"),
SUB_TAG_77("subTag77", "77"),
SUB_TAG_78("subTag78", "78"),
SUB_TAG_79("subTag79", "79"),
SUB_TAG_80("subTag80", "80"),
SUB_TAG_81("subTag81", "81"),
SUB_TAG_82("subTag82", "82"),
SUB_TAG_83("subTag83", "83"),
SUB_TAG_84("subTag84", "84"),
SUB_TAG_85("subTag85", "85"),
SUB_TAG_86("subTag86", "86"),
SUB_TAG_87("subTag87", "87"),
SUB_TAG_88("subTag88", "88"),
SUB_TAG_89("subTag89", "89"),
SUB_TAG_90("subTag90", "90"),
SUB_TAG_91("subTag91", "91"),
SUB_TAG_92("subTag92", "92"),
SUB_TAG_93("subTag93", "93"),
SUB_TAG_94("subTag94", "94"),
SUB_TAG_95("subTag95", "95"),
SUB_TAG_96("subTag96", "96"),
SUB_TAG_97("subTag97", "97"),
SUB_TAG_98("subTag98", "98"),
SUB_TAG_99("subTag99", "99");
```

QrCodeData

(Deprecated - Use HashMap based response and use QRCodeTag.[Item].tagCode() method to get keys for accessing individual items

ITEM	DESCRIPTION	FORMAT	METHODS
String payloadFormatIndicator	Defines the format of the merchant data payload.	CONDITIONAL Will be null if Qr Code version is 00	Getter: String getPayloadFormatIndicator()

String pointOfInitiation	Indicates the method by which the data is presented by the merchant. Indicates whether the data is static or dynamic. Refer to version 01 QR Spec	Optional	Getter: String getPointOfInitiation()
String mVisaMerchantId	Merchant Id	MANDATORY	Getter: String getmVisaMerchantId()
String mVisaMerchantPan	Merchant Id converted to merchant Pan(16 digit) – should be used in mVisa transaction as merchant id	MANDATORY	Getter: String getmVisaMerchantPan()
String masterCardPan1	Master card pan 1	OPTIONAL	Getter: String getMasterCardPan1()
String masterCardPan2	Master card pan 2	OPTIONAL	Getter: String getMasterCardPan2()
String npciid1	NPCI ID 1	OPTIONAL	Getter: String getNpciid1()
String npciid2	NPCI ID 2	OPTIONAL	Getter: String getNpciid2()
String merchantCategoryCode	Merchant category code as defined in As defined by ISO 8583:1993 for Card Acceptor Business Code.	MANDATORY	Getter: String getMerchantCategoryCode()
String currencyCode	Transaction Currency code as defined by ISO 4217	MANDATORY	Getter : String getCurrencyCode()
String transactionAmount	Transaction Amount	Optional	Getter: String getTransactionAmount()
String tipAndFeeIndicator	Tip and Fee Indicator "01": Indicates consumer should be prompted to enter tip "02": Indicates that merchant would mandatorily charge a flat convenience fee "03": Indicates that merchant would charge a percentage convenience fee	Optional	Getter: String getTipAndFeeIndicator()
String convenienceFeeAmount	Convenience fee amount	CONDITIONAL	Getter: String getConvenienceFeeAmount()
String primaryId	Primary Id value – populated only for Merchant Data version 00	OPTIONAL	Getter: String getPrimaryId()
String secondaryId	Secondary Id value – populated only for Merchant Data version 00	OPTIONAL	Getter: String getSecondaryId()
String convenienceFeePercentage	Convenience Fee Percentage	OPTIONAL	Getter: String getConvenienceFeePercentage()
String countryCode	Country code. As defined by ISO 3166.	OPTIONAL	Getter: String getCountryCode()
String merchantName	Merchant Name	OPTIONAL	Getter: String getMerchantName()
String cityName	City Name	OPTIONAL	Getter: String getCityName()

String postalCode	Postal code	OPTIONAL	Getter: String getPostalCode()
String billId	Bill number (Part of Additional Data)	OPTIONAL	Getter: String getAdditionalData()
String mobileNumber	Mobile Number	OPTIONAL	Getter: String getMobileNumber()
String storeId	Store ID	OPTIONAL	Getter: String getStoreId()
String loyaltyNumber	Loyalty number1	OPTIONAL	Getter: String getLoyaltyNumber()
String referenceId	Reference ID	OPTIONAL	Getter: String getReferenceId()
String consumerId	Consumer ID	OPTIONAL	Getter: String getConsumerId()
String terminalId	Terminal ID	OPTIONAL	Getter: String getTerminalId()
String purpose	Purpose	OPTIONAL	Getter: String getPurpose()
String additionalConsumerDataRequest	Additional Consumer Data Request	OPTIONAL	Getter: String getAdditionalConsumerDataRequest()
String addDataMasterCard1	Additional data for Master card 1	OPTIONAL	Getter: String getAddDataMasterCard1()
String addDataMasterCard2	Additional data for Master card 2	OPTIONAL	Getter: String getAdditionalDataMasterCard2
String addDataNpci1	Additional data for Npci 1	OPTIONAL	Getter: String getAddDataNpci1()
String addDataNpci2	Additional data for Npci 2	OPTIONAL	Getter: String getAddDataNpci2()
String crc	Cyclic Redundant Check value	MANDATORY	Getter: String getCrc()
boolean isPrimaryIdMandatory	Flag which defines whether primary ID is mandatory for consumer to enter or not	OPTIONAL	Getter; boolean isPrimaryIdMandatory()
String primaryIdLength	Mandatory length for primary ID	OPTIONAL	Getter: String getPrimaryIdLength()
boolean isSecondaryIdMandatory	Flag which defines whether Secondary ID is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isSecondaryIdMandatory()
String secondaryIdLength	Mandatory length for secondary ID	OPTIONAL	Getter: Boolean getSecondaryIdLength()
boolean isBilldMandatory	Flag which defines whether Bill ID is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isBillIdMandatory()
boolean isMobileNumberMandatory	Flag which defines whether Mobile number is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isMobileNumberMandatory()

boolean isStoreIdMandatory	Flag which defines whether Store ID is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isStoreIdMandatory()
boolean isLoyaltyNumberMandatory	Flag which defines whether Loyalty Number mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isLoyaltyNumberMandatory()
boolean isReferenceIdMandatory	Flag which defines whether Reference ID is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isReferenceIdMandatory()
boolean isConsumerIdMandatory	Flag which defines whether Consumer ID is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isConsumerIdMandatory()
boolean isTerminalIdMandatory	Flag which defines whether Store ID is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isTerminalIdMandatory()
boolean isPurposeMandatory	Flag which defines whether Store ID is mandatory for consumer to enter or not	OPTIONAL	Getter : Boolean isPurposeMandatory()

Sample JSON Response


```
{
        “qrCodeData” : {
                “payloadFormatIndicator” : “01”
                ……
                //all keys are same as
                }
        “qrCodeError” : [33, 40]
}
```