ធនាគារជាតិ នៃ កម្ពុជា
ប្រាក់រៀល. ស្ថិរភាព. អភិវឌ្ឍន៍.

Implementation Guideline

KHQR SDK Documentation

Version 2.3

December 2020

DOCUMENT HISTORY

Version	Date	Description
1.0	14.12.2020	
1.0.1	04.01.2021	- Alignment content. - Remove the Error Handling part. - Make all contents in Requirement, Installation and Usage parts more consistent.
1.1	25.01.2021	- Add a timestamp to the QR code string. - Add md5 hash string to response data.
1.1.1	01.02.2021	- Add Bill Number, Store Label, Terminal Label in QR - Add the request parameter section.
1.2	17.02.2021	Adding decode KHQR information function.
1.3	15.03.2021	Add Javascript SDK
1.4	20.04.2021	- Add function to generate KHQR deeplink. - Rename decode function name from **decodeQR** to **decode**. (iOS only)
1.5	08.07.2021	Add deeplink routing diagram
1.6	11.07.2021	- Add merchant Id for merchant type. - Add Acquiring Bank merchant name for merchant type. - Add a mobile number to additional data. - Capture value of merchant city. - Update generate KHQR function. - Update decode response value. - Add deeplink routing diagram.
1.7	01.09.2021	- Update generate individual by adding new tag for account information and acquiring bank name. - Update decode function without validate on required fields.
1.8	12.11.2021	- Make improvements on the verify function for iOS, Java and javascript.
1.9	23.11.2021	- Fix an issue on generating KHQR of an amount with leading zero.
2.0	04.01.2021	Moving library for stable API request on javascript SDK and improve verify KHQR content function for all SDK.

2.1	13.01.2022	Fix bug on decode function in javascript SDK
2.2	28.03.2022	Fix bug on decode function in javascript SDK
2.3	20.04.2022	Fix an issue on the verify function from get invalid status to valid status when QR contains any optional subtags that are not defined in standard KHQR document.

TABLE CONTENT

Introduction

Overview

The standardization of KHQR code specification will help promote wider use of mobile retail payments in Cambodia and provide consistent user experience for merchants and consumers. It can enable interoperability in the payment industry. A common QR code would facilitate payments among different schemes, e-wallets and banks and would encourage small merchants to adopt KHQR code as payment method.

KHQR is created for retail or remittance in Cambodia and Cross-Border. It only requires a single QR for receiving transactions from any payment provider through Bakong including Bakong App. For more detail please refer to **Prakas KHQR Code Specification** in Cambodia.

Purpose

This document describes the detailed specification of how to use KHQR SDK offered by **National Bank of Cambodia**. The expected readers are NBC technical team and third-party technical team. This can be used as reference for any interest related to the KHQR SDK.

Scope

This document contains the complete description of the KHQR SDK specification including: Features, Requirement, Installation Guide, Usage and FAQ.

Control Version

SDK	Version	Description
Jscavaript	1.0.0	Having generate decode and verify KHQR feature
	1.0.1	Add generate deeplink feature
	1.0.2	- Ghost Version
	1.0.3	- Add merchant Id for merchant type. - Add Acquiring Bank merchant name for merchant type. - Add a mobile number to additional data. - Capture value of merchant city. - Update generate KHQR function. - Update decode response value.
	1.0.4	- Ghost version
	1.0.5	- Update generate individual by adding new tag for account information and acquiring bank name. - Update decode function without validate on required fields - Fix amount formatting
	1.0.6	- Adding a function to check bakong account
	1.0.7	Make improvements on the verify function.
	1.0.8	Moving library for stable API request
	1.0.9	Improve verify KHQR content function.
	1.0.10	Fixed on decode function (Return null data)
	1.0.11	Fixed on decode cache issue
	1.0.12	- Ghost version
	1.0.13 **(Latest)**	- Fixed on wrong decode information and issue with timestamp without subtag on decode function. - Fix null value on optional data does not throw exceptions anymore.

iOS	0.1.1	
	1.0.0.1	- Add a timestamp to the QR code string. - Add md5 hash string to response data.
	1.0.0.2	- Add Bill number, Store label and terminal label to QR String.
	1.0.0.3	Add decode KHQR information function.
	1.0.0.4	Changes on function decode khqr - Validate length and CRC - Validate each tag according to KHQR specification
	1.0.0.5	Add generate deep link function
	1.0.0.6	- Add merchant Id for merchant type. - Add Acquiring Bank merchant name for merchant type. - Add a mobile number to additional data.. - Capture value of merchant city. - Update generate KHQR function. - Update decode response value.
	1.0.0.7	- Update generate individual by adding new tag for account information and acquiring bank name. - Update decode function without validate on required fields
	1.0.0.8	Make improvements on the verify function.
	1.0.0.9	Improve verify KHQR content function.
	1.0.0.10 **(Latest)**	Fix an issue on the verify function from get invalid status to valid status when QR contains any optional subtags that are not defined in standard KHQR document.
Java	1.0.0.4	Add decode function
	1.0.0.5	Add generate deep link function
	1.0.0.6	- Add merchant Id for merchant type.

		- Add Acquiring Bank merchant name for merchant type. - Add a mobile number to additional data.. - Capture value of merchant city. - Update generate KHQR function. - Update decode response value.
	1.0.0.7	- Update generate individual by adding new tag for account information and acquiring bank name. - Update decode function without validate on required fields
	1.0.0.8	Make improvements on the verify function.
	1.0.0.9 **(Latest)**	Improve verify KHQR content function.
C#	0.1.1	
	1.0.0.1	- Add a timestamp to the QR code string. - Add md5 hash string to response data.
	1.0.0.2	- Add Bill number, Store label and terminal label to QR String.
	1.0.0.3	Add decode KHQR information function.
	1.0.0.4	Changes on function decode khqr - Validate length and CRC - Validate each tag according to KHQR specification
	1.0.0.5	Add generate deep link function
	1.0.0.6	- Add merchant Id for merchant type. - Add Acquiring Bank merchant name for merchant type. - Add a mobile number to additional data.. - Capture value of merchant city. - Update generate KHQR function. - Update decode response value.
	1.0.0.7	- Update generate individual by adding new tag for account information and acquiring bank name. - Update decode function without validate on required fields

	1.0.0.8	Fix an issue on generating KHQR of an amount with leading zero.
	1.0.0.9 **(Latest)**	Improve verify KHQR content function.

Features

- Generate KHQR.
- Verification (Valid or Invalid).
- Decode KHQR Information.
- Generate KHQR Deeplink.

Requirement

iOS

- Development target 11.0 +
- Dependency management using Cocoapod

Java

- Source compatibility Java 8
- Target compatibility Java 8

C#

- NetStandard 2.0
- .Net Framework 4.0 and later
- .NET Core 2.0 and later
- Xamarin.iOS
- Xamarin.Android

Javascript

- NPM Javascript package management
- script raw file

Installation

iOS

CocoaPods

- **How to**

    1. Add source to podfile

```
source "https://sambo:ycfXmxxRbyzEmozY9z6n@gitlab.nbc.org.kh/khqr/khqr-ios-pod.git"
```

    2. Add pod name to podfile

```
pod "BakongKHQR"
```

    3. Run command

```
pod install
```

Java

Maven

- **How to**

    1. Add dependency to maven **pom.xml**

```
<dependency>
      <groupId>kh.org.nbc.bakong_khqr</groupId>
      <artifactId>sdk-java</artifactId>
       <version>current_version</version>
</dependency>
```

Gradle

- **How to**

    1. Add dependency to **build.gradle**

```
dependencies {
          implementation 'kh.org.nbc.bakong_khqr:sdk-java:{current_version}'
}
```

C#

Nuget

- **How to**

1. Using Visual Studio Package Manager

```
Install-Package Kh.Org.Nbc.BakongKHQR
```

2. Using Dotnet Cli

```
dotnet add package Kh.Org.Nbc.BakongKHQR
```

3. Include in csproj

```
<ItemGroup>
    <PackageReference Include="Kh.Org.Nbc.BakongKHQR" Version=current_version />
    <!-- ... other packages goes here -->
</ItemGroup>
```

Install Manually

If you prefer not to use nuget, you can integrate this SDK into your project manually via dll file.

- **How to**

1. Go to Nuget page of Kh.Org.Nbc.BakongKHQR
   https://www.nuget.org/packages/Kh.Org.Nbc.BakongKHQR
2. Download package (nupkg file).
3. Extract the package using any unzip tool.
4. Copy the dll file based on your platform from the following folder.

For .NetFramework 4.0 copy dll from this path

```
lib/net40/kh.org.nbc.bakong_khqr.dll
```

For .NetFramework 4.5 copy dll from this path

```
lib/net45/kh.org.nbc.bakong_khqr.dll
```

For .NetFramework 4.6 copy dll from this path

```
lib/net46/kh.org.nbc.bakong_khqr.dll
```

Other Platforms (.NetStandard 2.0) copy from this path

```
lib/netstandard2.0/kh.org.nbc.bakong_khqr.dll
```

Please consult the NetStandard support table for further explanation. https://docs.microsoft.com/en-us/dotnet/standard/net-standard#net-implementation-support

Javascript

NMP

- **How to**

    1. Init project with NPM

    ```
    npm init -y
    ```

    2. Install KHQR SDK

    ```
    npm install bakong-khqr
    ```

    3. Use package

    ```
    const {BakongKHQR, khqrData, IndividualInfo, MerchantInfo, SourceInfo} =
    require(“bakong-khqr”);
    ```

    Or

    ```
    import {BakongKHQR, khqrData, IndividualInfo, MerchantInfo, SourceInfo} from
    “bakong-khqr”;
    ```

Raw Script

- **How to**

    1. Link script to HTML

    ```
    <script
    src="https://drive.google.com/uc?export=view&id=12emdSpcYBx4NyOm-siW-OnDy0HEpdR-4">\
    </script>
    ```

    2. Use it at .js file

    ```
    let KHQR = BakongKHQR
    // This is the object consist of BakongKHQR, khqrData, IndividualInfo, MerchantInfo, SourceInfo
    ```

Usage

Request Parameter

- **Generate KHQR (Individual)**

Parameter	Type	Mandatory	Length	Description
BakongAccountID	String	Y	32	
AccountInformation	String	N	32	Account number or phone number.
AcquiringBank	String	N	32	Acquiring Bank name.
Currency	KHQRCurrency	N		Default: KHR
Amount	Double	N	13	
MerchantName	String	Y	25	
MerchantCity	String	N	15	Default: Phnom Penh
BillNumber	String	N	25	
MobileNumber	String	N	12	example: 855967854321
StoreLabel	String	N	25	
TerminalLabel	String	N	25	

- **Generate KHQR (Merchant)**

Parameter	Type	Mandatory	Length	Description
BakongAccountID	String	Y	32	
MerchantID	String	Y	32	
AcquiringBank	String	Y	32	Acquiring Bank name.
Currency	KHQRCurrency	N		Default: KHR

Amount	Double	N	13	
MerchantName	String	Y	25	
MerchantCity	String	N	15	Default: Phnom Penh
BillNumber	String	N	25	
MobileNumber	String	N	12	example: 855967854321
StoreLabel	String	N	25	
TerminalLabel	String	N	25	

- **Verification KHQR**

Parameter	Type	Mandatory	Length	Description
qrCode	String	Y		

- **Decode KHQR**

Parameter	Type	Mandatory	Length	Description
qrCode	String	Y		

- **KHQR Deeplink**

Parameter	Type	Mandatory	Length	Description
qr	String	Y		
url	String	Y		URL for API generate Deeplink. Provided by NBC team.
sourceInfo		N		
appIconUrl	String	Y		Your app icon
appName	String	Y		Your app name

appDeepLinkCallBack	String	Y		Deeplink url for opening your app after payment is completed.

iOS

- **Generate KHQR for Individual**

  Swift

```
let info = IndividualInfo(accountId: "john_smith@devb",
                              merchantName: "John Smith",
                          accountInformation: "85512233455",
                               acquiringBank: "Dev Bank",
                               currency: .Usd,
                                amount: 100)
info?.billNumber = "#12345"
info?.mobileNumber = "85512233455"
info?.storeLabel = "Coffee Shop"
info?.terminalLabel = "Cashier_1"
let khqrResponse = BakongKHQR.generateIndividual(info!)
if khqrResponse.status?.errorCode == 0 {
    let khqrData = khqrResponse.data as? KHQRData
    print("data: \(khqrData?.qr)")
    print("md5: \(khqrData?.md5)")
}
```

Objective C

```
IndividualInfo * info = [[IndividualInfo alloc] initWithAccountId: @"john_smith@devb"
                                                          merchantName: @"John Smith"
                                                         accountInformation: @"85512233455"
                                                         acquiringBank: @"Dev Bank"
                                                          currency: Usd
                                                          amount: 100];
[info setBillNumber: @"#12345"];
[info setMobileNumber: @"85512233455"];
[info setStoreLabel: @"Coffee Shop"];
[info setTerminalLabel: @"Cashier_1"];

KHQRResponse * khqrResponse = [BakongKHQR generateIndividual: info];
if (khqrResponse.status.errorCode == 0) {
    KHQRData * khqrData = (KHQRData *) khqrResponse.data;
    NSLog(@"data: %@", khqrData.qr);
    NSLog(@"md5: %@", khqrData.md5);
}
```

Result

```
data: 00020101021229460015john_smith@devb0111855122334550208Dev
Bank5204599953038405403100580２KH5910John Smith6010Phnom
Penh62500106#123450211855122334550311Coffee
Shop070601234599170013163117038554963049８0C

md5: 2d9afcceaa03c8685cf663b059d44534
```

- **Generate KHQR for Merchant**

  Swift

```swift
let merchantInfo = MerchantInfo(accountId: "devbkhppxxx@devb",
                               merchantId: "123456",
                            merchantName: "John Smith",
                               acquiringBank: "Dev Bank",
                               currency: .Usd,
                               amount: 100)
merchantInfo?.billNumber = "#12345"
merchantInfo?.mobileNumber = "85512233455"
merchantInfo?.storeLabel = "Coffee Shop"
merchantInfo?.terminalLabel = "Cashier_1"
let khqrResponse = BakongKHQR.generateMerchant(merchantInfo!)
if khqrResponse.status?.errorCode == 0 {
    let khqrData = khqrResponse.data as? KHQRData
    print(khqrData?.qr)
    print(khqrData?.md5)
}
```

Objective C

```
MerchantInfo * merchantInfo = [[MerchantInfo alloc] initWithAccountId:
@"devbkhppxxx@devb"
                                                        merchantId: @"123456"
                                                        merchantName: @"John Smith"
                                                        acquiringBank: @"Dev Bank"
                                                        currency: Usd
                                                        amount: 100];
[merchantInfo setBillNumber: @"#12345"];
[merchantInfo setMobileNumber: @"85512233455"];
[merchantInfo setStoreLabel: @"Coffee Shop"];
[merchantInfo setTerminalLabel: @"Cashier_1"];

KHQRResponse * khqrResponse = [BakongKHQR generateMerchant: merchantInfo];
if (khqrResponse.status.errorCode == 0) {
    KHQRData * khqrData2 = (KHQRData *) khqrResponse.data;
    NSLog(@"data: %@", khqrData2.qr);
    NSLog(@"md5: %@", khqrData2.md5);
}
```

Result

```
data: 00020101021230420016devbkhppxxx@devb01061234560208Dev
Bank52045999530384054031005802KH5910john Smith6010Phnom
Penh62550113Invoice#069030206#123450311Coffee
md5: 147d5ba80ec67969b1ec148648746281
```

- **Verification KHQR**

Swift

```swift
let qrCode = "00020101021229190015john_smith@devb5204599953038405405100.05802KH5910John Smith6010Phnom Penh62530106#123450211855122334550311Coffee Shop0709Cashier_19917001316262229737646304D940"

let response = khqr?.verify(qrCode)

let crcValidation = response?.data as? CRCValidation

print("valid : \(crcValidation?.valid)")
```

Objective C

```objc
NSString *qrCode = @"00020101021229190015john_smith@devb5204599953038405405100.05802KH5910John Smith6010Phnom Penh62530106#123450211855122334550311Coffee Shop0709Cashier_19917001316262229737646304D940";

KHQRResponse * response = [khqr verify: qrCode];

CRCValidation* crcValidation = (CRCValidation *) response.data;

NSLog(@"valid: %d", crcValidation.valid);
```

Result

```
valid: 1
```

- **Decode KHQR**

Swift

```
let decodeResponse =

BakongKHQR.decode("00020101021229190015john_smith@devb52045999530384054051
00.05802KH5910John Smith6010Phnom
Penh62530106#123450211855122334550311Coffee
Shop0709Cashier_199170013162622297376463O4D940")

let decodeData = decodeResponse?.data as? KHQRDecodeData

decodeData?.printAll()
```

Objective C

```
KHQRResponse* decodeResponse = [BakongKHQR decode:
@"00020101021229190015john_smith@devb5204599953038405405100.05802KH5910Jo
hn Smith6010Phnom Penh62530106#123450211855122334550311Coffee
Shop0709Cashier_19917001316262229737646304D940"];

KHQRDecodeData* decodeData = (KHQRDecodeData *) decodeResponse.data;

[decodeData printAll];
```

Result

```
payloadFormatIndicator = 01
pointOfInitiationMethod = 12
merchantType = 29
bakongAccountID = john_smith@devb
merchantAccountId = null
acquiringBank = null
merchantCategoryCode = 5999
countryCode = KH
merchantName = John Smith
merchantCity = Phnom Penh
transactionCurrency = 840
transactionAmount = 100.0
billNumber = #12345
mobileNumber = 85512233455
storeLabel = Coffee Shop
terminalLabel = Cashier_1
timestamp = 1626222973764
crc = D940
```

- **Generate KHQR deeplink**

Swift

```
let deeplinkResponse =
BakongKHQR.generateDeepLink("http://api.example.com/v1/generate_deeplink_by_qr",

qr:”00020101021229180014jonhsmith@nbcq52045999530384054031.05802KH5910Jonh Smith6010Phnom Penh6304C297”,
                sourceInfo: SourceInfo(appIconURL:"http://cdn.example.com/icons.logo.png",
                                    appName: "Example App",
                                    appDeepLinkCallBack: "http://app.example.com")
         )
if  deeplinkResponse.status.code == 0 {
        let deeplinkData = deeplinkResponse?.data as? KHQRDeepLinkData
        print(deeplinkData?.shortLink)
} else {
       print(deeplinkResponse.status.message)
}
```

Objective C

```
KHQRResponse* deeplinkResponse = [BakongKHQR generateDeepLink:
@"http://api.example.com/v1/generate_deeplink_by_qr"

qr:
@”00020101021229180014jonhsmith@nbcq52045999530384054031.05802KH5910Jonh
Smith6010Phnom Penh6304C297”

sourceInfo: [[SourceInfo alloc]

                              initWithAppIconURL:@"http://cdn.example.com/icons.logo.png"

                               appName:@"Example App"

                             appDeepLinkCallBack:@"http://app.example.com"]

                      ];

if (deeplinkResponse.status.code == 0) {

        KHQRDeepLinkData* deeplinkData = (KHQRDeepLinkData *) deeplinkResponse.data;

        NSLog(@"%@", deeplinkData.shortLink);

} else {

        NSLog(@"%@",deeplinkResponse.status.message);

}
```

Result

```
https://bakongsit.page.link/pBJzebGwUuQfMfhf9
```

*** **Note:** Generated short links will be different every time method is called and also depend on which api url being used.

Java

- **Generate KHQR for Individual**

```java
IndividualInfo individualInfo = new IndividualInfo();
individualInfo.setBakongAccountId("john_smith@devb");
individualInfo.setAccountInformation("85512233455");
individualInfo.setAcquiringBank("Dev Bank");
individualInfo.setCurrency(KHQRCurrency.USD);
individualInfo.setAmount(100.0);
individualInfo.setMerchantName("John Smith");
individualInfo.setMerchantCity("PHNOM PENH");
individualInfo.setBillNumber("#12345");
individualInfo.setMobileNumber("85512233455");
individualInfo.setStoreLabel("Coffee Shop");
individualInfo.setTerminalLabel("Cashier_1");
KHQRResponse<KHQRData> response = BakongKHQR.generateIndividual(individualInfo);
if (response.getKHQRStatus().getCode() == 0) {
    System.out.println("data: " + response.getData().getQr());
    System.out.println("md5: " + response.getData().getMd5());
 }
```

Result

```
data: 00020101021229460015john_smith@devb0111855122334550208Dev
Bank52045999530384054031005802KH5910John Smith6010PHNOM
PENH62530106#12345021185512233455 0311Coffee
Shop0709Cashier_199170013163115455739063041AFF

md5: 5bcfe307134683f68bd387abd1d4a804
```

- **Generate KHQR for Merchant**

```
 MerchantInfo merchantInfo = new MerchantInfo();
 merchantInfo.setBakongAccountId("devbkhppxxx@devb");
merchantInfo.setMerchantId("123456");
merchantInfo.setAcquiringBank("Dev Bank");
merchantInfo.setCurrency(KHQRCurrency.USD);
merchantInfo.setAmount(100.0);
 merchantInfo.setMerchantName("John Smith");
 merchantInfo.setMerchantCity("PHNOM PENH");
merchantInfo.setBillNumber("#12345");
merchantInfo.setMobileNumber("85512233455");
merchantInfo.setStoreLabel("Coffee Shop");
merchantInfo.setTerminalLabel("Cashier_1");
KHQRResponse<KHQRData> response = BakongKHQR.generateMerchant(merchantInfo);
if (response.getKHQRStatus().getCode() == 0) {
    System.out.println("data: " + response.getData().getQr());
     System.out.println("md5: " + response.getData().getMd5());
 }
```

Result

```
data: 00020101021230420016devbkhppxxx@devb01061234560208Dev
Bank5204599953038405403100580 2KH5910John Smith6010PHNOM
PENH62530106#12345021185512233455031 1Coffee
Shop0709Cashier_199170013163115269902563 04C6C4

md5: 02f4f94d30be3f031da7b5c952444064
```

- **Verification KHQR**

```java
String  qrCode = "00020101021230420016devbkhppxxx@devb01061234560208Dev Bank5204599953038405403100 5802KH5910John Smith6010PHNOM PENH62530106#12345021185512233455 0311Coffee Shop0709Cashier_19917001316311526990256304C6C4";

KHQRResponse<CRCValidation> response = BakongKHQR.verify(qrCode);

System.out.println("valid:  " + response.getData().isValid());
```

Result

```
valid: true
```

- **Decode KHQR**

```java
String  qrCode = "00020101021230420016devbkhppxxx@devb01061234560208Dev Bank5204599953038405403100 5802KH5910John Smith6010PHNOM PENH62530106#12345021185512233455 0311Coffee Shop0709Cashier_19917001316311526990256304C6C4";

KHQRResponse<KHQRDecodeData> response = BakongKHQR.decode(qrCode);

System.out.println(response);
```

Result

```
KHQRResponse{
    status=KHQRStatus{
    code=0,
    errorCode=null,
    message='null’ }
  data=KHQRDecodeData{
    payloadFormatIndicator='01’,
    pointOfInitiationMethod='12’,
    merchantType='30’,
    bakongAccountID='devbkhppxxx@devb’,
    merchantId='123456’,
    accountInformation='nul’l,
    acquiringBank='Dev Bank’,
    merchantCategoryCode='5999’,
    countryCode='KH’,
    merchantName='John Smith’,
    merchantCity='PHNOM PENH’,
    transactionCurrency='840’,
    transactionAmount='100’,
    billNumber='#12345’,
    storeLabel='Coffee Shop’,
    terminalLabel='Cashier_1’,
    mobileNumber='85512233455’,
    timestamp='1631152699025’,
    crc='C6C4’}
}
```

- **Generate KHQR deeplink**

```
 String url = "http://api.example.com/v1/generate_deeplink_by_qr";
 String qr =
  "00020101021229180014jonhsmith@nbcq52045999530384054031.05802KH5910Jonh
  Smith6010Phnom Penh6304C297";
 SourceInfo sourceInfo = new SourceInfo();
 sourceInfo.setAppName("Example App");
 sourceInfo.setAppIconUrl("http://cdn.example.com/icons.logo.png");
sourceInfo.setAppDeepLinkCallback("http://app.example.com");
KHQRResponse<KHQRDeepLinkData> response = BakongKHQR.generateDeepLink(url,
qr, sourceInfo);
 if (response.getKHQRStatus().getCode() == 0) {
      System.out.println(response.getData().getShortLink());
 }else {
      System.out.println(response.getKHQRStatus().getMessage());
 }
```

Result

```
https://bakongsit.page.link/8dBnac2EQefMtmsM6
```

***** Note:** Generated short links will be different every time method is called and also depend on which api url being used.

C#

- **Generate KHQR For Individual**

```
var response = BakongKHQR.GenerateIndividual(
  new IndividualInfo {
    BakongAccountID = "john_smith@devb",
    Currency = KHQRCurrency.USD,
    AccountInformation = "85512233455",
    AcquiringBank = "Dev Bank",
    Amount = 100,
    MerchantName = "John Smith",
    MerchantCity = "PHNOM PENH",
    BillNumber = "#12345",
    MobileNumber = "85512233455",
    StoreLabel = "Coffee Shop",
    TerminalLabel = "Cashier_1",
  });
if (response.Status.Code == 0){
  Console.WriteLine("data: " + response.Data.QR);
  Console.WriteLine("md5: " + response.Data.MD5);
}
```

Result

```
data: 00020101021229460015john_smith@devb0111855122334550208Dev
Bank5204599953038405405100.05802KH5910John Smith6010PHNOM
PENH62530106#12345021185512233455031 1Coffee
Shop0709Cashier_199170013163055480984063045E5C

md5: 1fa8ba0d948f1d4629c14c1b57ac7869
```

- **Generate KHQR For Merchant**

```
var response = BakongKHQR.GenerateMerchant( new MerchantInfo
    {
        BakongAccountID = "devbkhppxxx@devb",
        MerchantID = "123456",
        AcquiringBank = "Dev Bank",
        Currency = KHQRCurrency.USD,
        Amount = 100,
        MerchantName = "John Smith",
        MerchantCity = "PHNOM PENH",
        BillNumber = "#12345",
        MobileNumber = "85512233455",
        StoreLabel = "Coffee Shop",
        TerminalLabel = "Cashier_1",
    });
if (response.Status.Code == 0){
    Console.WriteLine("data: " + response.Data.QR);
    Console.WriteLine("md5: " + response.Data.MD5);
}
```

Result

```
data: 00020101021230420016devbkhppxxx@devb01061234560208Dev
Bank52045999530384054031005802KH5910John Smith6010PHNOM
PENH62530106#123450211855122334550311Coffee
Shop0709Cashier_19917001316310870199636304A508

md5: bffb15cbad9d2c46f169c914aacf011c
```

- **Verification KHQR**

```
var response = BakongKHQR.Verify("00020101021229190015john_smith@devb5204599953038405405100.05802KH5910John Smith6010PHNOM PENH62530106#123450211855122334550311Coffee Shop0709Cashier_199170013162622395476963047FDE");

if (response.Status.Code == 0)

{

    Console.WriteLine("valid: " + response.Data.Valid);

}
```

Result:

```
valid: True
```

- **Decode KHQR**

```
var response = BakongKHQR.Decode("00020101021230420016devbkhppxxx@devb01061234560208Dev Bank52045999530384054031005802KH5910John Smith6010PHNOM PENH62530106#123450211855122334550311Coffee Shop0709Cashier_19917001316310870199636304A508");

var options = new JsonSerializerOptions()

{

    WriteIndented = true

};

Console.WriteLine(JsonSerializer.Serialize(response, options));
```

Result

```
{
 "Status": {
   "Code": 0,
   "ErrorCode": null,
   "Message": null  },
  "Data": {
   "PayloadFormatIndicator": "01",
   "PointOfInitiationMethod": "12",
   "MerchantType": "30",
   "BakongAccountID": "devbkhppxxx@devb",
   "MerchantID": "123456",
   "AccountInformation": null,
   "AcquiringBank": "Dev Bank",
   "MerchantCategoryCode": "5999",
   "CountryCode": "KH",
   "MerchantName": "John Smith",
   "MerchantCity": "PHNOM PENH",
   "TransactionCurrency": "840",
   "TransactionAmount": "100",
   "BillNumber": "#12345",
   "MobileNumber": "85512233455",
   "StoreLabel": "Coffee Shop",
   "TerminalLabel": "Cashier_1",
   "Timestamp": "1631087019963",
   "CRC": "A508" }
}
```

- **Generate KHQR deeplink**

```
var url = "http://api.example.com/v1/generate_deeplink_by_qr";
var qr = "00020101021229180014jonhsmith@nbcq52045999530384054031.05802KH5910Jonh Smith6010Phnom Penh6304C297";
var sourceInfo = new SourceInfo {
    AppName = "Example App",
    AppIconUrl = "http://cdn.example.com/icons.logo.png",
    AppDeepLinkCallback = "http://app.example.com"
};
var response = BakongKHQR.GenerateDeepLink(url, qr, sourceInfo);
if (response.Status.Code == 0) {
    Console.WriteLine(response.Data.ShortLink);
} else {
    Console.WriteLine(response.Status.Message);
}
```

Result

```
https://bakongsit.page.link/DqKPTPQ521gXd4g8A
```

***** Note:** Generated short links will be different every time method is called and also depend on which api url being used.

Javascript

- **Generate Individual KHQR**

```
const {
 BakongKHQR,
   khqrData,
   IndividualInfo,
   MerchantInfo,
} = require("bakong-khqr");

const optionalData = {
   currency: khqrData.currency.usd,
   amount: 100.5,
   accountInformation: "85512233455",
   acquiringBank: "Dev Bank",
   mobileNumber: "85512233455",
   storeLabel: "Coffee Shop",
   terminalLabel: "Cashier_1",
};
const individualInfo = new IndividualInfo(
   "jonh_smith@devb",
   "Jonh Smith",
   "PHNOM PENH",
   optionalData);
const KHQR = new BakongKHQR();
const individual = KHQR.generateIndividual(individualInfo);
console.log("qr: " + individual.data.qr);
console.log("md5: " + individual.data.md5);
```

Result

```
qr: 00020101021229460015jonh_smith@devb0111855122334550208Dev
Bank5204599953038405406100.505802KH5910Jonh Smith6010PHNOM
PENH62430211855122334550311Coffee
Shop0709Cashier_199170013163115460356263048065

md5: f5a7dca43bcda57276d4fb175c17fd51
```

- **Generate Merchant KHQR**

```
const {
    BakongKHQR,
    khqrData,
    IndividualInfo,
    MerchantInfo,
} = require("bakong-khqr");
const optionalData = {
    currency: khqrData.currency.usd,
    amount: 100.5,
    mobileNumber: "85512233455",
    storeLabel: "Coffee Shop",
    terminalLabel: "Cashier_1",
};
const merchantInfo = new MerchantInfo(
    "devbkhppxxx@devb",
    "Jonh Smith",
    "PHNOM PENH",
    "123456",
    "Dev Bank",
    optionalData
);

const KHQR = new BakongKHQR();

const merchant = KHQR.generateMerchant(merchantInfo);

console.log("qr: " + merchant.data.qr);

console.log("md5: " + merchant.data.md5);
```

Result

```
qr: 00020101021230420016devbkhppxxx@devb01061234560208Dev
Bank5204599953038405406100.505802KH5910Jonh Smith6010PHNOM
PENH62430211855122334550311Coffee
Shop0709Cashier_19917001316311538559786304B37C

md5: e9c634e37a8453fc790a91e55960880a
```

- **Verification KHQR**

```
let khqrString = "00020101021230420016devbkhppxxx@devb01061234560208Dev
Bank5204599953038405406100.505802KH5910Jonh Smith6010PHNOM
PENH62430211855122334550311Coffee
Shop0709Cashier_19917001316311538559786304B37C";

const isKHQR = BakongKHQR.verify(khqrString);

console.log("valid:" + isKHQR.isValid)
```

Result

```
valid:true
```

- **Decode KHQR**

```
let khqrString =  "00020101021230420016devbkhppxxx@devb01061234560208Dev Bank5204599953038405406100.505802KH5910Jonh Smith6010PHNOM PENH62430211855122334550311Coffee Shop0709Cashier_19917001316311538559786304B37C";

const decodeValue = BakongKHQR.decode(khqrString)

console.log(decodeValue)
```

Result

```
{
  status: { code: 0, errorCode: null, message: null },
  data: {
    merchantType: '30',
    bakongAccountID: 'devbkhppxxx@devb',
    accountInformation: null,
    merchantID: '123456',
    acquiringBank: 'Dev Bank',
    billNumber: null,
    mobileNumber: '85512233455',
    storeLabel: 'Coffee Shop',
    terminalLabel: 'Cashier_1',
    payloadFormatIndicator: '01',
    pointofInitiationMethod: '12',
    merchantCategoryCode: '5999',
    transactionCurrency: '840',
    transactionAmount: '100.50',
    countryCode: 'KH',
    merchantName: 'Jonh Smith',
    merchantCity: 'PHNOM PENH',
    timestamp: '00131631153855978',
    crc: 'B37C'
  }
}
```

- **Generate KHQR deeplink**

```
const khqrString = "00020101021229190015john_smith@devb52045999530384054051 00.05802KH5910John Smith6010Phnom Penh6304BF30";
const sourceInfo = new SourceInfo(
    "http://cdn.example.com/icons.logo.png",
    "Example App",
    "http://app.example.com");
const deeplink = khqr.generateDeepLink(
    "http://api.example.com/v1/generate_deeplink_by_qr",
    khqrString,
    sourceInfo
);
deeplink.then((data) => {
    if (data.status.code == 0) {
        console.log(data.data.shortLink);
    } else {
        console.log(data.status.message);
    }
});
```

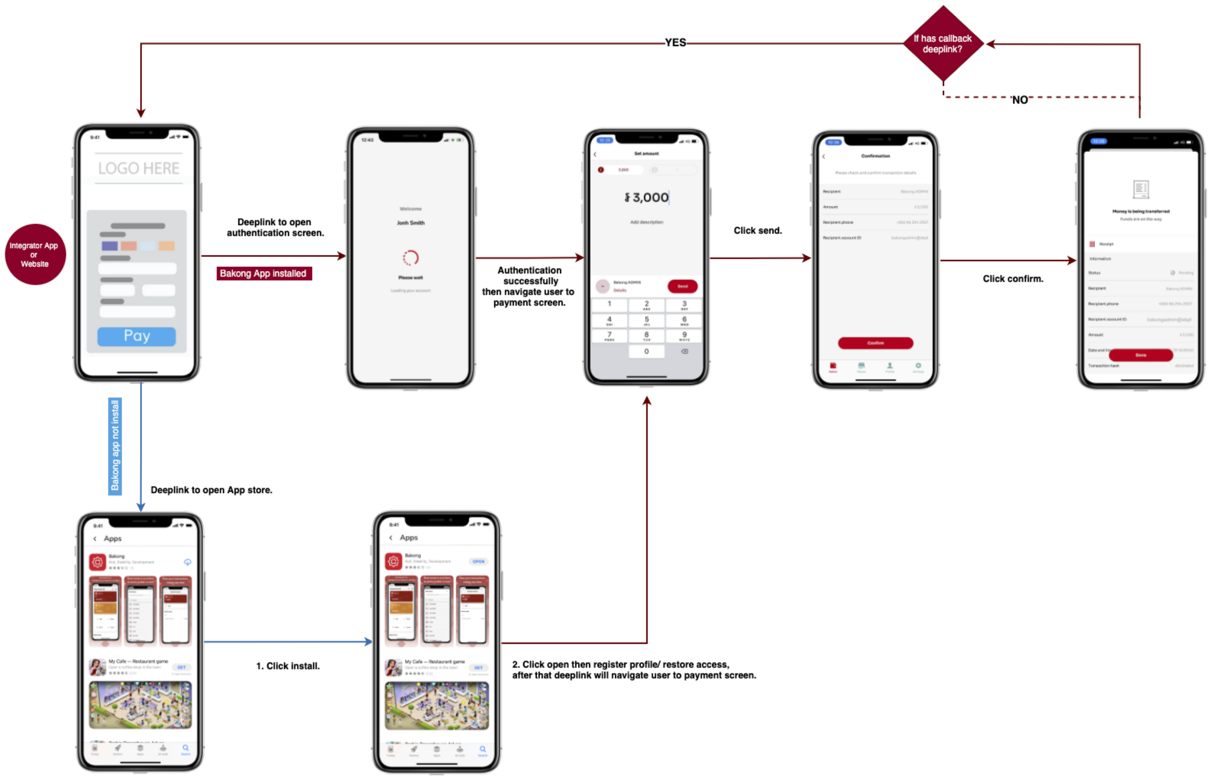
Result

```
https://bakongsit.page.link/16ahB249Xu9TQc676
```

***** Note:** Generated short links will be different every time method is called and also depend on which api url being used.

Deeplink Diagram

Deeplink Rounting


YES
If has callback deeplink?
NO
Integrator App or Website
LOGO HERE
Pay
Deeplink to open authentication screen.
Bakong App installed
Welcome
Jonh Smith
Please wait
Authentication successfully then navigate user to payment screen.
Set amount
3,000
Send
Click send.
Confirmation
Confirm
Click confirm.
Money is being transferred
Done
Bakong app not install
Deeplink to open App store.
Apps
Bakong
My Cafe — Restaurant game
GET
OPEN
1. Click install.
2. Click open then register profile/ restore access, after that deeplink will navigate user to payment screen.

Standard Response

Response Format

```
{
 "status": {
     "code": 0,
     "errorCode": null,
     "message": ‘This is message’
   },
   "data": {
     //response based on each function
         qr: “00020101021229460015john_smith@devb0111855122334550208Dev
Bank52045999530384054031005802KH5910John Smith6010Phnom
Penh62500106#123450211855122334550311Coffee
Shop07060123459917001316311703855496304980C”
   }
```

Response Key and Code with definition

Key	Type	Value	Message
code	int		
		0	Success
		1	Failed
errorCode	int		
		1	Bakong Account ID cannot be null or empty
		2	Merchant name cannot be null or empty
		3	Bakong Account ID is invalid
		4	Amount is invalid

		5	Merchant type cannot be null or empty
		6	Bakong Account ID Length is invalid
		7	Merchant Name Length is invalid
		8	KHQR provided is invalid
		9	Currency type cannot be null or empty
		10	Bill Number Length is invalid
		11	Store Label Length is invalid
		12	Terminal Label Length is invalid
		13	Cannot reach Bakong Open API service. Please check internet connection
		14	Source Info for Deep Link is invalid
		15	Internal Server Error
		16	Payload Format Indicator Length is invalid
		17	Point of Initiation Length is invalid
		18	Merchant Category Length is invalid
		19	Transaction Currency Length is invalid
		20	Country Code Length is invalid
		21	Merchant City Length is invalid
		22	CRC Length is invalid
		23	Payload Format Indicator cannot be null or empty
		24	CRC cannot be null or empty
		25	Merchant Category cannot be null or empty
		26	Country Code cannot be null or empty
		27	Merchant City cannot be null or empty

		28	Unsupported currency
		29	Deep Link URL is not valid
		30	Merchant ID cannot be null or empty
		31	Acquiring Bank cannot be null or empty
		32	Merchant ID Length is invalid
		33	Acquiring Bank Length is invalid
		34	Mobile Number Length is invalid
		35	Tag not in order
		36	Account information length is invalid
message	String		The message describes the result.
data	Any Object		The response will be different based on function
qr	String		emv qr code provided by KHQR library.
md5	String		hash value of bakong KHQR to verify transaction status.

FAQ

1. **If my bank wants to know more or inquire about KHQR, who should I talk to?**

   KHQR team of Association of Bank in Cambodia

2. **If my bank wants to know more or inquire about How To Implement KHQR By Using Bakong As a Payment Switch, who should I talk to?**

   Bakong team of National Bank of Cambodia