PayWithEaseBuzz Payment kit Integration (Android)

1. Copy **peb-lib.aar** or **peb-lib-android-x.aar(For Android-x Project)** file into app/libs/ folder of the merchant application.
2. Proguard Rules configurations:
   Add below line to your proguard rules.

```
-keepclassmembers class com.easebuzz.payment.kit.**{
        *;
      }
```

3. build.gradle(module) modifications.
   Add this line to build.gradle (module)

```
defaultConfig {
            multiDexEnabled true
        }
```

   Add the following lines to packagingOptions,

```
    exclude 'META-INF/DEPENDENCIES'
    exclude 'META-INF/NOTICE'
    exclude 'META-INF/LICENSE'
    exclude 'META-INF/LICENSE.txt'
    exclude 'META-INF/NOTICE.txt'
```

   Add the following line to dexOptions.

```
    javaMaxHeapSize "4g"
```

   Add repositories section as follows.

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

   Add the following dependencies
   All the dependencies mentioned below are compulsory. If your app is already using any of the following dependencies then you do not require to add these particular dependencies.

   Note : Add the dependencies based on Android and Android-X Project.

   Dependencies For Android :

   1. implementation(name: 'peb-lib', ext: 'aar')
   2. implementation 'com.squareup.okhttp:okhttp:2.4.0'
   3. implementation 'com.android.support:multidex:1.0.1'
   4. implementation 'com.squareup.okhttp:okhttp-urlconnection:2.2.0'
   5. implementation 'com.squareup.retrofit2:retrofit:2.3.0'
   6. implementation 'com.squareup.retrofit2:converter-gson:2.3.0'

Dependencies For Android-X : (For Android-X Projects)

1. implementation (name: 'peb-lib-android-x', ext: 'aar')
2. implementation 'com.google.android.material:material:1.0.0'
3. implementation 'com.squareup.okhttp:okhttp:2.4.0'
4. implementation 'androidx.multidex:multidex:2.0.0'
5. implementation 'com.squareup.okhttp:okhttp-urlconnection:2.2.0'
6. implementation 'com.squareup.retrofit2:retrofit:2.5.0'
7. implementation 'com.squareup.retrofit2:converter-gson:2.5.0'
8. implementation 'com.google.android.gms:play-services-auth:17.0.0'
9. implementation 'com.google.android.gms:play-services-auth-api-phone:17.1.0'

Your Easebuzz Kit configuration is done. Follow on for steps to integrate.

4. **Initiate Payment :** Generate access key using the Initiate Payment API at your backend.
(It is mandatory to integrate the Initiate Payment API at Backend only)
Initiate Payment API Doc : https://docs.easebuzz.in/api/initiate-payment

5. Initiating Payment Transaction Request from Merchant App.
After having successfully set up the build.gradle configuration files and successfully added the **peb-lib.aar** to your app/libs, your app is required to start the PWECouponsActivity of PayWithEaseBuzz module. The detailed process is described below.

5.1. Import these packages

```
import com.easebuzz.payment.kit.PWECouponsActivity;
import datamodels.PWEStaticDataModel;
```

5.2. Fetch that access key into your app and pass that access key to payment intent.

5.3. On click of the pay button from the app you need to start PWECouponsActivity and pass the necessary parameters to PWECouponsActivity using Intent. (Use startActivityForResult() method to start the Activity).

5.4 startActivityForResult() method syntax:

```
startActivityForResult(intent, PWEStaticDataModel.PWE_REQUEST_CODE);
```

PWEStaticDataModel.PWE_REQUEST_CODE is integer constant, having value 100.
Here the **intent** is used to start the PWECouponsActivity. Also you need to pass some crucial parameters using putExtra() method described below in detail,

```
Intent intentProceed = new Intent(MerchantAppActivity.this, PWECouponsActivity.class);
intentProceed.setFlags(Intent.FLAG_ACTIVITY_REORDER_TO_FRONT);  // This is mandatory flag
intentProceed.putExtra("access_key",access_key);
intentProceed.putExtra("pay_mode",payment_mode);
startActivityForResult(intentProceed, PWEStaticDataModel.PWE_REQUEST_CODE);
```

Note : Please do not change the name parameter of the putExtra() method mentioned in the above code. The datatype of the value can be changed according to the value to be sent.

6. Handling the response of the payment process.

The transaction is initiated using startActivityForResult() method, your app must override the OnActivityResult() method of Activity class. This method will receive the status and response of the transaction.

Syntax of the onActivityResult() as follows :

protected void *onActivityResult(**int** requestCode, **int** resultCode, Intent data) { }.*

Here the Parameter data of type Intent receives the result and response of the payment process.

Merchant App can get the result as follows :

*String result = data.getStringExtra(**"result"**);*

This result indicates the status of the transaction as success or failed the detailed code is described below.

The **result** can be ,

1. "payment_successfull"
2. "payment_failed"
3. “txn_session_timeout”
4. "back_pressed"
5. "user_cancelled"
6. "error_server_error"
7. "error_noretry"
8. "invalid_input_data"
9. "Retry_fail_error"
10. “trxn_not_allowed”

Your app can get the detailed response as follows :

*String response = data.getStringExtra(**"payment_response"**);*

This response is a json string and can be parsed to get the details about the ongoing transaction.

Success Response :

```
{
txnid : '1001',
firstname : 'John Doe',
email : 'johndoe@gmail.com',
phone : '7767819428',
key : 'DF3252FDSF',
mode : 'DC',
status : 'success',
unmappedstatus : 'failed',
cardCategory : 'domestic',
addedon : '2016-07-22 17:17:08',
payment_source : 'Easebuzz',
PG_TYPE : 'SBIPG',
bank_ref_num : '',
bankcode : 'MAST',
error : 'E600',
error_msg : 'Bank denied transaction on card.',
name_on_card : 'John',
cardnum : '519620XXXXXX7840',
issuing_bank : '',
card_type : '',
easepayid : 'H5T2RYZKW',
amount : '100.00',
net_amount_debit : '100.00',
cash_back_percentage : '50',
deduction_percentage : '2.50',
productinfo : 'Tshirt',
udf10 : '',
udf9 : '',
udf8 : '',
udf7 : '',
udf6 : '',
udf5 : '',
udf4 : '',
udf3 : '',
udf2 : '',
udf1 : '',
hash :
'ce2d0588f8648c62db86475d343d3433d00b87827502c676a093730f04cec5fea2eb0e8bb4f47fcdea955f61b674171f1
93c883686d2da42300d00e921a217c3'
}
```

Failed Response :


```
{
   txnid : '1001',
   firstname : 'John Doe',
   email : 'johndoe@gmail.com',
   phone : '7767819428',
   key : 'DF3252FDSF',
   mode : 'DC',
   status : 'failure',
   unmappedstatus : 'failed',
   cardCategory : 'domestic',
   addedon : '2016-07-22 17:17:08',
   payment_source : 'Easebuzz',
   PG_TYPE : 'SBIPG',
   bank_ref_num : '',
   bankcode : 'MAST',
   error : 'E600',
   error_msg : 'Bank denied transaction on card.',
   name_on_card : 'John',
   cardnum : '519620XXXXXX7840',
   issuing_bank : '',
   card_type : '',
   easepayid : 'H5T2RYZKW',
   amount : '100.00',
   net_amount_debit : '100.00',
   cash_back_percentage : '50',
   deduction_percentage : '2.50',
   productinfo : 'Tshirt',
   udf10 : '',
   udf9 : '',
   udf8 : '',
   udf7 : '',
   udf6 : '',
   udf5 : '',
   udf4 : '',
   udf3 : '',
   udf2 : '',
   udf1 : '',
   hash :
'ce2d0588f8648c62db86475d343d3433d00b87827502c676a093730f04cec5fea2eb0e8bb4f47fcdea955f61b674171f1
93c883686d2da42300d00e921a217c3'
}
```

OnActivityResult() Detailed code and description.

Note : Handle the response only when the parameter **'Intent data'** is not null in onActivityResult() method.

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
  if(data != null ) {

        if (requestCode == PWEStaticDataModel.PWE_REQUEST_CODE)
        {

         try {
                String result = data.getStringExtra("result");
                String payment_response = data.getStringExtra("payment_response");
               if (result.contains(PWEStaticDataModel.TXN_SUCCESS_CODE)) {
                    //PWEStaticDataModel.TXN_SUCCESS_CODE is a string constant and its value is "payment_successfull"
                    //Code here will execute if the payment transaction completed successfully.
                   // here merchant can show the payment success message.

               } else if(result.contains(PWEStaticDataModel.TXN_TIMEOUT_CODE))  {
               //PWEStaticDataModel.TXN_TIMEOUT_CODE is a string constant and its value is "txn_session_timeout"
                     //Code here will execute if the payment transaction failed because of the transaction time out.
                      // here merchant can show the payment failed message.
               }
               else if(result.contains(PWEStaticDataModel.TXN_BACKPRESSED_CODE)) {
                 //PWEStaticDataModel.TXN_BACKPRESSED_CODE is a string constant and its value is "back_pressed"
                 //Code here will execute if the user pressed the back button on coupons Activity.
                 // here merchant can show the payment failed message.
               }
               else if(result.contains(PWEStaticDataModel.TXN_USERCANCELLED_CODE))   {
                  //PWEStaticDataModel.TXN_USERCANCELLED_CODE is a string constant and its value is "user_cancelled"
                   //Code here will execute if the the user pressed the cancel button during the  payment process.
                   // here merchant can show the payment failed message.
               } else if(result.contains(PWEStaticDataModel.TXN_ERROR_SERVER_ERROR_CODE))   {
                      //PWEStaticDataModel.TXN_ERROR_SERVER_ERROR_CODE is a string constant and its value
                        is  "error_server_error"
                     //Code here will execute if the server side error occured during payment process.
                     // here merchant can show the payment failed message.
               } else if(result.contains(PWEStaticDataModel.TXN_ERROR_TXN_NOT_ALLOWED_CODE))
                      {
                         //PWEStaticDataModel.TXN_ERROR_TXN_NOT_ALLOWED_CODE is a string constant and its value is
                          "trxn_not_allowed"
                         //Code here will execute if the the transaction is not allowed.
                        // here merchant can show the payment failed message.
                      } else if(result.contains(PWEStaticDataModel.TXN_BANK_BACK_PRESSED_CODE))
                      {
                         //PWEStaticDataModel.TXN_BANK_BACK_PRESSED_CODE is a string constant and its value is
                          "bank_back_pressed"
```

```
                    //Code here will execute if the the customer press the back button on bank screen.
                    // here merchant can show the payment failed message.
                }

            else{
                    // Here the value of result is "payment_failed" or "error_noretry" or "retry_fail_error"
                    //Code here will execute if payment is failed some other reasons.
                    // here merchant can show the payment failed message.
                }

        }catch (Exception e){
            //Handle exceptions here
        }

    }
  }
}
```