PayU S.A.
182, Grunwaldzka Str.
60-166 Poznań
Poland

Tel. +48 61 630 60 05
Email: pomoc@payu.pl
www.payu.pl

PayTouch Windows Phone v1.3.2

Mobile Payments for PayU platform

PayTouch Windows Phone Integration

The PayTouch library makes it easy to integrate mobile payments in the Windows Phone application. The library is available in form of a single DLL dependency. It provides both network communication, security and user interface for the payment process. It takes care of the payment method management, processing payment and payment authorization (CVV, 3D Secure, bank transfer in webview, etc.)

The main use case is to accept single payment with following payment methods:

- Credit/Debit cards (Visa, Mastercard, Maestro)
- PayU Express - http://express.payu.pl/
- Pay-By-Link – http://payu.pl/blog/pay-by-link-plac-bezposrednio-z-konta/

Requirements

- Windows Phone 8.0 Silverlight or later
- Server-side OAuth2 token retrieval – described in a separate document

Importing the PayTouch

Reference NuGet package named PayTouch either by using Manage NuGet Packages… option in Visual Studio 2013 or typing in Package Manager Console `Install-Package PayTouch`.

Credentials

For production environment you will have to retrieve OAuth2 access token from the PayU backend service via your backend application. This process must be conducted by your backend since direct storage of your `client_id` and `client_secret` in the application is **prohibited**. The only request the application should do is "retrieve access token for currently logged user".

Localization

The PayTouch has support for Polish and English language. Language is selected based on current value of `Thread.CurrentThread.CurrentUICulture`.

Documentation

- This document, showing basic introduction & usage
- The sample application provided within PayTouch bundle
- API reference – refer to `PayUService` class for in-depth explanation
- Server-side integration documentation

Integration tutorial

Complete example of usage is provided as an example application. Here we present minimum set of configuration required.

PayTouch initialization

Your application should always call `PayUService.Instance.Initialize()` in App.xaml.cs constructor. This allows PayU PayTouch to initialize properly.

```
public partial class App : Application
{
    public App()
    {
        InitializeComponent();
        PayUService.Instance.Initialize();
    }
}
```

Paying using external app from PayTouch

In order to enable paying using external app (PayU app, bank app, etc) you need to do following things:

- Your application need to register URI association in `WMAppManifest.xml` file. For more information please refer to: http://msdn.microsoft.com/en-us/library/windows/apps/jj206987(v=vs.105).aspx#BKMK_URIassociations
- Pass name of your URI association to `PayUService.Instance.MerchantAppProtocol` – it's best to do this in App.xaml.cs constructor
- Create URI mapper and call `PayUService.Instance.HandleAssociationUri` passing Uri you received. Uri may contain information about payment from external app
- Pass your URI mapper to RootFrame in App.xaml.cs.

Below is example of URI mapper:

```
private class AssociationUriMapper : UriMapperBase
{
    public override Uri MapUri(Uri uri)
    {
        if (PayUService.Instance.HandleAssociationUri(uri))
        {
            // Uri was properly handled by PayTouch.

            return new Uri("/MainPage.xaml", UriKind.Relative);
        }
        else
        {
            // PayTouch can't handle provided Uri.
        }

        return uri;
    }
}
```

```
public partial class App : Application
{
        private void InitializePhoneApplication()
        {
            RootFrame.UriMapper = new AssociationUriMapper();
        }
}
```

There is also one case you need to take into account. By default applications on Windows Phone platform are launching as completely new instance. Let's consider following example. User is paying using your application. However during payment process, user have to use bank application to confirm payment. He is leaving your application and bank application is launching. While in bank application user changes his mind, minimalize bank application (using Windows key) and launch your application from tile or app list. Your application will be started as completely new instance instead of returning to cart/payment page. That might be little confusing for user.

We recommend to implement in your application feature called "fast app resume" (https://msdn.microsoft.com/en-us/library/windows/apps/jj735579%28v=vs.105%29.aspx). It allows to resume application state from point where user left it. Implementing this feature will improve user experience of your application.

Token Provider Service

In order to use PayTouch in a production environment you have to create a class that implements `IMerchantAccessTokenProvider`. This class will be used by PayTouch, in order to obtain valid access token. Pass instance of your class to `PayUService.Instance.MerchantAccessTokenProvider` - it's best to do this in App.xaml.cs constructor.

In the example application you can find the implementation that may be used as a framework. Your implementation will be called by the PayTouch in a background thread, contact your server to obtain PayU access token for currently logged user. Example:

```
public class SampleMerchantAccessTokenProvider : IMerchantAccessTokenProvider
{
    public async Task<ServiceResponse<AccessToken>> GetAccessTokenAsync()
    {
        return await Task.Factory.StartNew(() =>
        {
            // Fetch access token from your server and return result.

            var serviceResponse = new ServiceResponse<AccessToken>();
            serviceResponse.IsOk = true; // Determine if call to your server was successful.
            //serviceResponse.Exception = // Pass exception optionally.
            serviceResponse.Result = new AccessToken
            {
                TokenString = new SecureString
                {
                    Value = "access token value"
                }
            };

            return serviceResponse;
        });
    }
}
```

User logout notification

When user decides to logout from your application, PayTouch **<u>must be notified</u>** about it, in order to clear sensitive user data . It is **<u>critical to call</u>** `PayUService.Instance.LogOutUser()` after successful user logout to avoid unexpected PayTouch behaviour.

Payment method chooser

PayTouch provides complete payment method management. Your user is able to select or add payment method across different scenarios.

The only required call is invoking payment method selection process. In the result of this process user will select payment method, log in to PayU account or add a new card.

Once the user selects the payment method it is stored internally. There is no need for you to provide it to the payment process.

```
public partial class CartPage : PhoneApplicationPage
{
    private void ButtonChangePaymentMethod_Click(object sender, RoutedEventArgs e)
    {
        PayUService.Instance.NavigateToPaymentMethodChooser();
    }
}
```

Payment

Payment is the last point of the process. When you invoke `PayUService.Instance.SendOrderAsync` the user will be directed through the payment process. In the best case scenario the payment will be finished with no interaction. However keep in mind that the user may be asked to confirm payment using some verification method (CVV, 3DS, etc.). All the UI is provided by PayU and you are not required to do any action. Your application needs only to provide progress bar until payment is completed. Please refer to sample application.

Note: Every single payment must be confirmed server side as described here.

Sending order is asynchronous. Example payment:

```
PayUService.Instance.SendOrderAsync(
    // Order description that will be displayed to the user
    description: "example payment",
    // Notifications are sent in JSON format using POST method
    notifyUrl: "http://notify.me/notify-endpoint",
    // Order identifier assigned by the Seller
    externalOrderId: Guid.NewGuid().ToString(),
    // Payment amount defined in a fractional part of defined Currency
    amount: 500,
    // Currency instance
    currency: Currency.PLN);
```

For retrieving order status subscribe to event `PayUService.Instance.SendOrderCompleted.` We suggest subscribing to that event in App.xaml.cs constructor, so you won’t miss any order status. Sometimes order status can be determined after being tombstoned in some page in PayTouch or after calling `PayUService.Instance.HandleAssociationUri()` in your URI mapper.

Payment method widget

For your convenience we have prepared `PaymentMethodWidget` control, which can be used in your cart view. This control displays selected payment method and allows to navigate to payment method chooser (after tap). Below is sample code, which shows how to use it from XAML:

```
<phone:PhoneApplicationPage
    x:Class="MerchantSampleApp.CartPage"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:controls="clr-namespace:PayU.SDK.Controls;assembly=PayU.WP8.SDK"
    xmlns:phone="clr-namespace:Microsoft.Phone.Controls;assembly=Microsoft.Phone">
    <Grid x:Name="LayoutRoot">
        <controls:PaymentMethodWidget Theme="Theme1" />
    </Grid>
</phone:PhoneApplicationPage>
```

Please note that widget won’t be visible in designer, because resources for it are resolved during runtime.

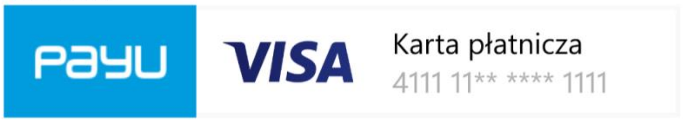
Additionally you can subscribe to event `PayUService.Instance.PaymentMethodChanged` and check if user has selected payment method (`PayUService.Instance.IsPaymentMethodSelected`). This is useful in scenario where you want to disable “pay” button until user selects payment method.

`PaymentMethodWidget` comes with six themes, which you can use. Choose theme, which best suits into your app. Below are presented available themes:

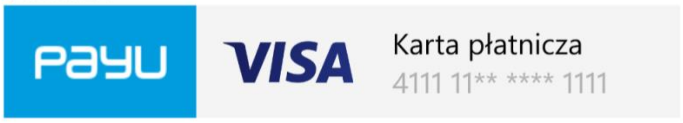
PayU S.A.
182, Grunwaldzka Str.
60-166 Poznań
Poland

Tel. +48 61 630 60 05
Email: pomoc@payu.pl
www.payu.pl

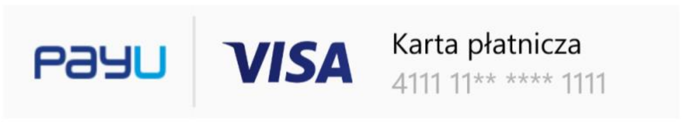
Theme1



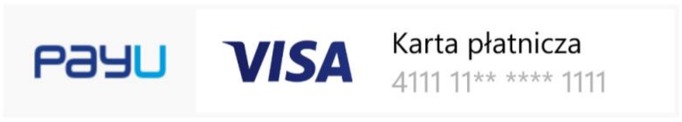
Theme2



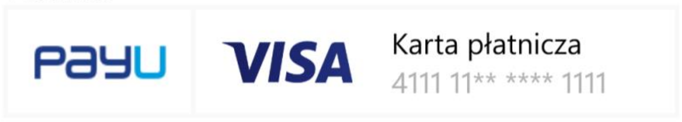
Theme3



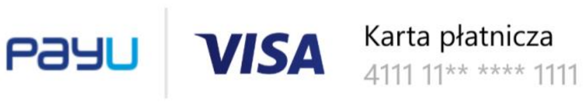
Theme4



Theme5



Theme6


Karta płatnicza
4111 11** **** 1111