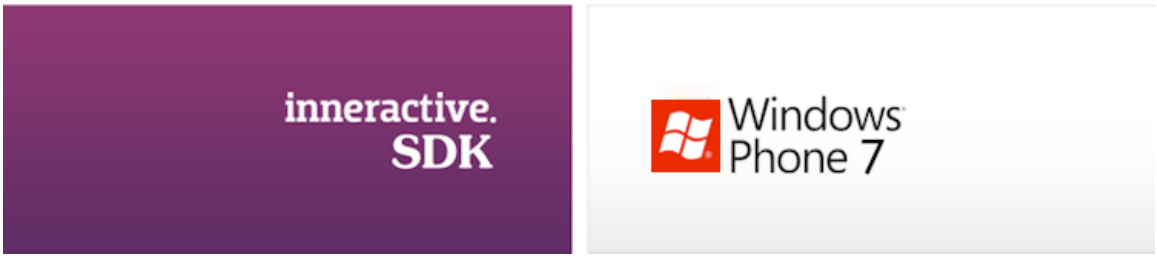
Windows Phone 7 v1.1.2 guidelines



Introduction

The Inneractive Ad SDK provides a simple way to integrate inneractive ads into mobile applications based on the Windows Phone platform. The SDK enables partners to retrieve ads – including rich media ads and location-based ads – and embed them into applications designed for Windows Phone 7.1 (Mango) and newer. For Windows Phone 8 check out our dedicated version here.

This document begins by first explaining how to integrate ads into your application using either XML or C# code. It then goes on to show you how to harness inneractive's highly targeted ad system by providing additional information with your requests. Next, you'll see how to receive and act upon events such as the user clicking an ad. Finally, you'll learn how to harness inneractive's location-based ads for maximizing your revenue.

But first, let's look at requesting an ad for display.

Integrating Ads

Integrating ads into your Windows Phone application requires just three steps:

1. Add the SDK to your project.
2. Add required references and capabilities.
3. Display ads within the application.

1. Add the SDK

To add the SDK to your project, you must add the **inneractiveAdSDK DLL** file to the project's references. To do this within Visual Studio:

1. Locate the **InneractiveAdSDK** folder in Windows Explorer and open it. Right-click the **Inneractive.Ad.dll** file and choose **Properties**. At the bottom of the window, under **Security**, click the **Unblock** button and click **Apply**.
2. Copy the **Inneractive.Ad.dll** from the **InneractiveAdSDK** folder to your project.
3. If you plan on using location-based ads, copy the **InneractiveAdLocation.cs** file from the **InneractiveAdSDK** folder to your project.

2. Add required references and capabilities

In order to use the SDK, you must add a reference to the **Inneractive.Ad.dll** file to the project. You can do that by following these steps:

1. Choose **Project | Add Reference** in Visual Studio.
2. Click the **Browse** tab.
3. Navigate to the **Inneractive.Ad.dll** file within the project and select it.
4. Click **OK**.

If you are using location-based ads in your app, you will also need to add an additional reference and capability to your project. To do that, follow these steps:

1. Choose **Project | Add Reference** in Visual Studio.
2. Click the **.NET** tab.
3. Select **System.Device** and click **OK.**

In order to use location-based ads, you must also do the following:

Open the **Properties | WMAppManifest.xml** file and add the **ID_CAP_LOCATION** capability to the **Capabilities** element. For example:

```
<App xmlns="" ProductID="{e0242d33-2eb6-42bf-a330-bdc8adc9bb20}" Title="MyApplication"

      RuntimeType="Silverlight" Version="1.1.2" Genre="apps.normal"  Author="Joe
Developer"
      Description="Sample description" Publisher="MyCompany">
    <IconPath IsRelative="true" IsResource="false">ApplicationIcon.png</IconPath>
    <Capabilities>
      <Capability Name="ID_CAP_LOCATION"/>
      <Capability Name="ID_CAP_GAMERSERVICES"/>
      <Capability Name="ID_CAP_IDENTITY_DEVICE"/>
      ...
```

If you add the **InneractiveAdLocation.cs** file to the project, you must add a reference to **System.Device** in order to run your application.

3. Display ads

Depending on how you've structured your application, you can choose to add inneractive ads by either adding them to the layout XML file, or directly through your C# code.

Option 1 - XAML

If you are building your application through the use of XAML layout files, you can easily add the **InneractiveAd** directly to your layout. For example, you can add the following code to your **MainPage.xaml** file to display a banner ad at the bottom of the fifth row of a grid:

```
<ad:InneractiveAd
 xmlns:ad="clr-namespace:Inneractive.Nokia.Ad;assembly=Inneractive.Ad"
 AppID="MyCompany_MyApp"
 AdType="IaAdType_Banner"
 ReloadTime="60"
 Grid.Row="5"
 Name="InneractiveXamlAd"
/>
```

The inneractive SDK does not support live layout in the Visual Studio designer, but the application will run property despite any exceptions related to the designer.

The inneractive SDK does not currently support Interstitial ad display via XAML code. We recommend using the DisplayAd function in order to display an interstitial ad.

The following parameters are **required**:

AppID	An App ID is provided to you upon the creation of an app on your inneractive console .
AdType	The type of the ad you want to display. You have two options: **Banner** — `IaAdType_Banner` **Text** — `IaAdType_Text`

ReloadTime	The number of seconds between ad refreshes. (This value is not relevant for interstitial ads.) The minimum interval is 30 seconds. The default interval is 60 seconds.

You also have the option of adding additional optional parameters to improve user experience and targeting.

Option 2 - C#

If you're taking a more programmatic approach, you may wish to add your ads directly from your C# code. The inneractive SDK provides two options for doing that: calling a method, or using a control.

The first step is to reference the SDK:

```
...
using Microsoft.Phone.Controls;

using Inneractive.Nokia.Ad;

namespace MyApp {
...
```

To actually add the ad to the display, you can either use the **DisplayAd()** method or the **InneractiveAd** control.

The static **DisplayAd()** method takes all of the required information in a single step:

```
InneractiveAd.DisplayAd(<AppId>, <AdType>, <ContainerName>, <ReloadTime> ,
<optionalParams>);
```

For example:

```
InneractiveAd.DisplayAd("MyCompany_MyApp", InneractiveAd.IaAdType.IaAdType_Banner,
ContentPanel, 60, optionalParams);
```

The alternative is to create the **InneractiveAd** control, then add it to the container:

```
InneractiveAd iaBanner = new InneractiveAd(<AppId>, <AdType>, <ReloadTime>,
<optionalParams>);
ContentPanel.Children.Add(iaBanner);
```

For example:

```
InneractiveAd iaBanner = new InneractiveAd("MyCompany_MyApp",
InneractiveAd.IaAdType.IaAdType_Banner, 60, optionalParams);
ContentPanel.Children.Add(iaBanner);
```

The required parameters are the same as they were for the XAML version:

AppID	An App ID is provided to you upon the creation of an app on your inneractive console .

AdType	The type of the ad you want to display. You have the following options: **Banner** — `IaAdType_Banner` **Text** — `IaAdType_Text` **Interstitial** — `IaAdType_Interstitial`
ReloadTime	The number of seconds between ad refreshes. The minimum interval is 30 seconds. The default interval is 60 seconds. (This value is not relevant for interstitial ads.)

You also have the option of adding additional optional parameters to improve user experience and targeting.

Improve Performance

The inneractive SDK gives you several opportunities to improve the way your users experience ads within your application, and thus your revenue. Some of these options include:

- Supplying additional parameters to improve ad targeting.
- Providing a more seamless integration between ads and application content.
- Restricting application use if ads are not available.
- Using location-based ads.

Suppy additional parameters to improve targeting

Monetization of your app works best when the ads presented to your users are relevant to them, and that means you'll want to provide the best targeted ads possible. To do that, you can pass a number of parameters with your ad request, including:

Parameter name (C#)	Parameter name (XML)	Description
Key_Age	**Age**	The user's age
Key_Gender	**Gender**	The user's gender (allowed values are **M**, **m**, **F**, **f**, **Male**, and **Female**)
Key_Gps_Coordinates	**GPS**	The GPS ISO code location data in latitude, longitude format, with no spaces. For example: **53.542132,-2.239856** **Note:** in case the location services is already turned on (see here), there is no need to add this parameter, the SDK itself will send the right coordinates
Key_Keywords	**Keywords**	Keywords relevant to this user's specific session, separated by commas, with no spaces. For example: **cars,music,sports**
Key_Location	**Location**	Comma separated list of country,state/province,city, with no spaces. For example: **US,NY,NY**

Key_Ad_Alignment	**AdAlignment**	Alignment of the banner within the container (not relevant for interstitial ads). Possible values: • **InneractiveAd.IaAdAlignment.TOP_LEFT** • **InneractiveAd.IaAdAlignment.TOP_CENTER** • **InneractiveAd.IaAdAlignment.TOP_RIGHT** • **InneractiveAd.IaAdAlignment.BOTTOM_LEFT** • **InneractiveAd.IaAdAlignment.BOTTOM_CENTER** • **InneractiveAd.IaAdAlignment.BOTTOM_RIGHT** • **InneractiveAd.IaAdAlignment.CENTER_LEFT** • **InneractiveAd.IaAdAlignment.CENTER** • **InneractiveAd.IaAdAlignment.CENTER_RIGHT**

Retrieve specific ad sizes

You can request for a specific ad size using the optional parameters. The inneractive SDK provides support for multiple screen sizes and densities. When the SDK renders the ad, it takes into account screen density. Pixel density may be scaled up/down for the desired ad size.The ad sizes parameters relevant for banner ads only, and include:

Banner Required Ad's Width	Enforcing a specific Ad width for the ad. In case that there was no ad with the required width size found, a house Ad will be displayed. Using required parameter may affect the fill rate, so please use this parameter only if you have a limited banner's placeholder.
Banner Required Ad's Height	Enforcing a specific Ad height for the ad. In case that there was no ad with the required height size found, a house Ad will be displayed. Using required parameter may affect the fill rate, so please use this parameter only if you have a limited banner's placeholder.
Banner Optional Ad's Width	This parameter is relevant only for Banner Ads. Using optional parameter does not guarantee you'll receive an ad of this width unless the server accommodates it. In order to guarantee the ad's width please use required parameter.
Banner Optional Ad's Height	This parameter is relevant only for Banner Ads. Using optional parameter does not guarantee you'll receive an ad of this height unless the server accommodates it. In order to guarantee the ad's height please use required parameter.

List of Supported ad sizes

WindowsPhone
300 x 50
320 x 50
480 x 80
300x250 (rectangle)
320 x 480 (interstitial)

To use these parameters, you set them using a Dictionary, then provide the Dictionary when requesting the ad. For example:

Adding optional parameters

```
using System.Collections.Generic;
...
            Dictionary<InneractiveAd.IaOptionalParams, string> optionalParams = new
Dictionary<InneractiveAd.IaOptionalParams, string>();
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Age, "25");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Gender, "m");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Keywords,
"test,inneractive");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Gps_Coordinates,
"53.5422,-2.2396");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Location,
"US,NY,NY");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Ad_Alignment,
InneractiveAd.IaAdAlignment.CENTER.ToString());
   optionalParams.Add(InneractiveAd.IaOptionalParams.Key_OptionalAdWidth, "480");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_OptionalAdHeight,
"80");


            InneractiveAd iaBanner = new InneractiveAd("MyCompany_MyApp",
InneractiveAd.IaAdType.IaAdType_Banner, 60, optionalParams);
            ContentPanel.Children.Add(iaBanner);
...
```

You can also set parameters in your XAML. For example:

Adding optional parameters

```
...
<ad:InneractiveAd
    xmlns:ad="clr-namespace:Inneractive.Nokia.Ad;assembly=Inneractive.Ad"
 AppID="MyCompany_MyApp"
 AdType="IaAdType_Banner"
 ReloadTime="60"
 Grid.Row="5"
 Name="InneractiveXamlAd"

    Age="25"
    Gender="m"
    GPS="53.5422,-2.2396"
    Keywords="test,inneractive"
    Location="US,NY,NY"
 AdAlignment="CENTER"
 OptionalAdWidth="480"
    OptionalAdHeight="80"
/>
...
```

The more and better information you pass to inneractive, the more relevant ads your users will see, and the more likely they will be to click them.

Improving the user experience: Receive InneractiveAd events

One way to avoid letting ads get in the way of your application is to work with the user's interactions, and not against them. For example, you may wish to pause your app when the user clicks an ad. Fortunately, you can receive **InneractiveAd** events using event handlers and delegates.

The available events are:

Event name	Description
AdReceived	Called when a new paid ad has been received and is about to be displayed.
DefaultAdReceived	Called when a new house ad has been received and is about to be displayed.
AdFailed	Called when an ad request fails, and does not retrieve an ad.
AdClicked	Called when the user clicks an ad.

To react to these events, you need to create functions for them and set up event handlers to call them.

Examples:

Example using an instance of the InneractiveAd object:

```
...
   InneractiveAd iaBanner = new InneractiveAd("MyCompany_MyApp",
InneractiveAd.IaAdType.IaAdType_Banner, 60);
            // Add event listeners to the InneractiveAd
            iaBanner.AdReceived += new
InneractiveAd.IaAdReceived(InneractiveAd_AdReceived);
   iaBanner.AdFailed += new InneractiveAd.IaAdFailed(InneractiveAd_AdFailed);
    iaBanner.DefaultAdReceived += new
InneractiveAd.IaDefaultAdReceived(InneractiveAd_DefaultAdReceived);
   iaBanner.AdClicked += new InneractiveAd.IaAdClicked (InneractiveAd_AdClicked);
    ContentPanel.Children.Add(iaBanner);
        }

        void InneractiveAd_AdReceived(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd: AdReceived");
        }


  void InneractiveAd_AdFailed(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd: AdFailed");
        }

  void InneractiveAd_DefaultAdReceived (object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd: DefaultAdReceived");
        }

  void InneractiveAd_AdClicked (object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd: AdClicked");
        }

 }
...
```

Example using the static DisplayAd method with :

```
...
     IaAdEventHandlers eventHandlers = new MyIaAdEventHandlers();
        InneractiveAd.DisplayAd("MyCompany_MyApp",
InneractiveAd.IaAdType.IaAdType_Text, GetGrid(), 60, eventHandlers);
     }

 }

    class MyIaAdEventHandlers : IaAdEventHandlers
    {
        public override void AdFailedEventHandler(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd.DisplayAd: AdFailed");
        }
        public override void AdReceivedEventHandler(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd.DisplayAd: AdReceived");
        }
        public override void DefaultAdReceivedEventHandler(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd.DisplayAd:
DefaultAdReceived");
        }
        public override void AdClickedEventHandler(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd.DisplayAd: AdClicked");
        }
        public MyIaAdEventHandlers()
        {
        }
    }
...
```

Example using an XAML based ad, with "InneractiveXamlAd" as its name parameter, add the following to your page's .cs file:

```
...
   InneractiveAd xamlAd = (InneractiveAd)this.FindName("InneractiveXamlAd");
            xamlAd.AdReceived += new InneractiveAd.IaAdReceived
(InneractiveAd_AdReceived);
   xamlAd.AdFailed += new InneractiveAd.IaAdFailed (InneractiveAd_AdFailed);
   xamlAd.DefaultAdReceived += new InneractiveAd.IaDefaultAdReceived
(InneractiveAd_DefaultAdReceived);
   xamlAd.AdClicked += new InneractiveAd.IaAdClicked(InneractiveAd_AdClicked);


   void InneractiveAd_AdReceived(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd: AdReceived");
        }
  void InneractiveAd_AdFailed(object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd: AdFailed");
        }
  void InneractiveAd_DefaultAdReceived (object sender)
        {
            System.Diagnostics.Debug.WriteLine("InneractiveAd: DefaultAdReceived");
        }

  void InneractiveAd_AdClicked(object sender)
     {
         System.Diagnostics.Debug.WriteLine("InneractiveAd: AdClicked");
     }
}
...
```

Requiring connectivity: knowing when ads are available

In order to maximize revenue, we recommend allowing your application to run only when ads are available. For example, you might require the user to have internet access available, or terminate the application. You might also provide a limited experience unless ads are enabled.

In order to provide this capability, you can check the return value of the **DisplayAd()** method; it will return `true` if the ad is ready to load, and false if not. For example:

```
...
if (!InneractiveAd.DisplayAd("MyCompany_MyApp",
InneractiveAd.IaAdType.IaAdType_Banner, ContentPanel, 60, optionalParams)){
    MessageBox.Show("This application is free but requires an internet connection.
Please configure your connectivity settings and retry.");
    NavigationService.GoBack();
}
...
```

Note that the Windows Phone Emulator will ALWAYS return true, even if an internet connection is not available.

Using location-based ads

Another way to increase your revenue is to use the **IaLocationClass** to enjoy high paying location-based ads. To do that, you will need to enable your application to make use of location information, then create an event handler.

If you didn't already prepare your application for location access, go ahead and do that now. From there, you can use the **IaLocationClass** that

comes with the inneractive SDK to detect changes in location. For example:

```
...
IaLocationClass iaLocation = new IaLocationClass();
iaLocation.Done += new System.EventHandler<IaLocationEventArgs>(iaLocation_Done);
iaLocation.StartWatchLocation();
...

void iaLocation_Done(object sender, IaLocationEventArgs e)
{
   if (e != null && e.location != null) {
      try
      {
         optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Gps_Coordinates,
e.location);
      }
      catch (ArgumentException ex)
      {
         System.Diagnostics.Debug.WriteLine("Location problem: "+e.location);
      }
   }
   InneractiveAd.DisplayAd("MyCompany_MyApp", InneractiveAd.IaAdType.IaAdType_Banner,
LayoutRoot, 60, optionalParams);
}
...
```

More advice on optimal ad placement

inneractive's **Ad Placement Strategy** (APS) provides a wealth of information on the different types of ads available to you, and some of the best ways to maximize user response while providing a positive user experience.

Putting it together: A complete example

If you've followed all the steps on this page, your **MainPage.xml.cs** page will look something like this:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Animation;
using System.Windows.Shapes;
using Microsoft.Phone.Controls;
using System.Collections.Generic;
using Inneractive.Nokia.Ad;
using InneractiveAdLocation;

namespace MyTestApp
{
    public partial class MainPage : PhoneApplicationPage
    {
        Dictionary<InneractiveAd.IaOptionalParams, string> optionalParams;
```

```
        public MainPage()
        {
            InitializeComponent();
            optionalParams = new Dictionary<InneractiveAd.IaOptionalParams, string>();
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Age, "25");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Gender, "m");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Keywords,
"test,inneractive");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Ad_Alignment,
InneractiveAd.IaAdAlignment.CENTER.ToString());
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Gps_Coordinates,
"53.5422,-2.2396");
            optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Location,
"US,NY,NY");

            IaLocationClass iaLocation = new IaLocationClass();
            iaLocation.Done += new
System.EventHandler<IaLocationEventArgs>(iaLocation_Done);
            iaLocation.StartWatchLocation();

            if (!InneractiveAd.DisplayAd("MyCompany_MyApp",
InneractiveAd.IaAdType.IaAdType_Banner, ContentPanel, 60, optionalParams)){
                MessageBox.Show("This application is free but requires an internet
connection. Please configure your connectivity settings and retry.");
                NavigationService.GoBack();
            }
        }
        void iaLocation_Done(object sender, IaLocationEventArgs e)
        {
            if (e != null && e.location != null)
            {
                try
                {

optionalParams.Add(InneractiveAd.IaOptionalParams.Key_Gps_Coordinates, e.location);
                }
                catch (ArgumentException ex)
                {
                    System.Diagnostics.Debug.WriteLine("Location problem:
"+e.location);
                }
            }
```

```
        }
    }
}
```

The Windows Phone emulator returns an invalid location, but the proper location should be detected on an actual phone.

Additional information

Here are some additional tips and tricks for adding inneractive ads to your Windows Phone applications.

Banner size

The inneractive SDK returns banner ads of up to 480 x 80 pixels.

Enforcing a specific Ad width for the ad