Inbox

The inbox feature was made to display some delayed messages to your users. These messages can be a text, a webpage, a richpush, ... with or without buttons.

These messages and buttons interact directly with the Accengage SDK.
You can for instance send a message with an event button, trigger a URL scheme, or display a banner/interstitial when the user opens a message.
This inbox can be used to store push messages received but not opened by the user.
Indeed, with this feature, the user can browse all of the messages you sent to him before.
The inbox can be used as a reminder or as a lite mail client in order to communicate with your clients.

If you want to use this feature, Accengage servers will send all the messages information to your app, but you will need to handle yourself the display part.

With Accengage Inbox you can :

- Obtain messages and display them in your own template
- Update messages status on Accengage Servers (Read/Unread/Archived)

You can find a complete example in our sample project.

After being sure you greatly integrate our sdk with the previous chapter, you can now access to our Inbox Implementation.

Obtain Messages

You will need an async callback to be notified when an Inbox is received. Here is an example of how to define it:

```
Accengage.GetInbox(LoadInbox);

private void LoadInbox(AccengageInbox inbox)
{
    // Messages were downloaded, are ready and contained in a AccengageInbox Object.
}
```

List Messages

Once you retrieved your AccengageInbox object you can access to AccengageInboxMessage. In this example, we will just display a log with the title of each message:

> The process is asynchronous and the callback order is NOT garanteed.
> Double checking the index provided when a message is received is mandatory to avoid errors in your code.

```
private void LoadInbox(AccengageInbox inbox)
{
    // First, we will get each message, so let's do a loop
    for( int i =0; i < inbox.Size; i++){
        inbox.ObtainMessageAtIndex(i, LoadMessage);
    }
}

private void LoadMessage(AccenageInboxMessage message, int position)
{
    // do your treatment
}
```

Message interactions

Now that we have downloaded and displayed our messages, we will allow our user to interact with them.

For example, if you display a message in a list, you may want to provide the user with some interaction if he touches a row.

You will need to give the message the user interacted with to the SDK.

According to the message format, the SDK can either start a predefined action or give it back to you to be displayed.

```
// Call message 'interactWithDisplayHandler' method to hand the message to the SDK
message.InteractWithDisplayHandler(DisplayDetailInbox);

// If this method is called back, I will need to display message content to the user
myself.
private void DisplayDetailInbox(AccengageInboxMessageContent content)
{
    // do your treatment
}
```

Once it is loaded you have fully access to all the content of your message.

Work with Message Buttons

A message content can have some buttons. Here is an example of how to display and interact with these them.

```
private void DisplayDetailInbox(AccengageInboxMessageContent content)
{
    var buttons = messageContent.Buttons;
    if (buttons.Count > 0)
    {
        foreach (var button in buttons)
        {
            // Create ui button
            ...
            uiButton.Click += delegate {
                button.Interact();
            };
        }
    }
}
```

Update message status

You can change the status of a message, here is how to do it:

```
// Marks a message as read or unread
message.Read = true;
message.Read = false;

// Archive or Unarchive a message
message.Archived = true;
message.Archived = false;
```