HTML, CSS, JAVASCRIPT
IN-APP DEVELOPMENT WORKFLOW

Date   September 2020

Author: Calvin Parks: cparks@laughlin.com

Overview Discussion:

History: The Laughlin Email development strategy was created by Dariusz jarosz and perhaps other Dev's that I am not aware of.

I started developing In-Apps for Laughlin when Dariuz left the company in October of 2019.

Observations:

Braz's In-App development is pretty simple to understand. They are HTML content served in a Webview in the Hoosier IOS and Android native app.

Dev Team

Laughlin Email Team members

- Jenny Maglio: Marketing Automation Associate Director- jmaglio@laughlin.com
- Cathleen LaFond: Marketing Automation Specialist - clafond@laughlin.com

Documentation Links

VIDEO: Learn How to Use Appboy's In-App Messages
https://www.youtube.com/watch?v=nwp7prafy3Y

In-App Messages using HTML5

https://github.com/Appboy/appboy-custom-html5-in-app-message-templates

HTML In-App Messages

https://www.braze.com/docs/user_guide/message_building_by_channel/in-app_messages/customize/

BRAZE SDK for IOS and Android Development
?????????????

Hoosier Lottery website

https://hoosierlottery.com

The official IOS Hoosier Lottery App

https://apps.apple.com/us/app/hoosier-lottery/id894092490

The official Android Hoosier Lottery App

https://play.google.com/store/apps/details?id=com.gtechg2.mobile.indiana&hl=en_US

SETUP TOOLS

PhotoShop - Raster graphics editor

- Website https://photoshop.com

TinyPNG - Image compression site that uses "smart lossy compression techniques" to make your images smaller.

- Website https://tinypng.com

Laughlin In-App Repo - Repository for the final email that has been launched.
- https://github.com/laughlin/lc_client_emails/tree/master/clients/Hoosier%20MyLottery/In-app

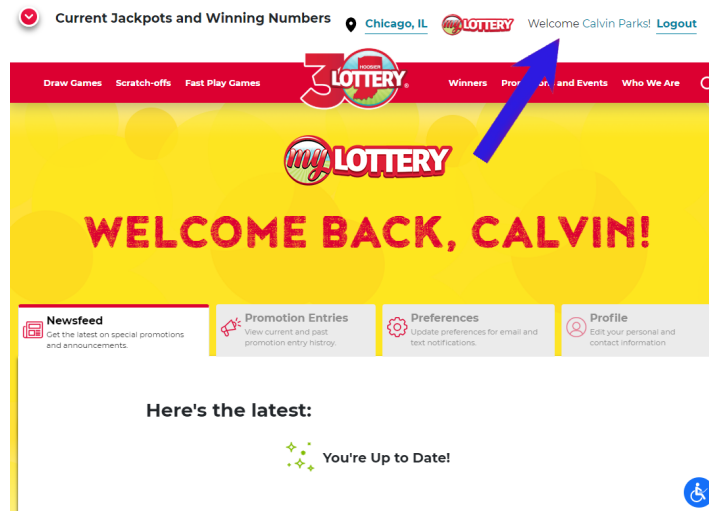**Braze -** Braze is a customer engagement platform used to create email campaigns

https://www.braze.com

- Login: hoosierbraze@laughlin.com - Griffin2018

Download apps and set up notifications

Log into https://hoosierlottery.com/ site and signup and sign-in the set up notifications

Download hoosier lottery app and sign-in to it using the username and password that created on the website
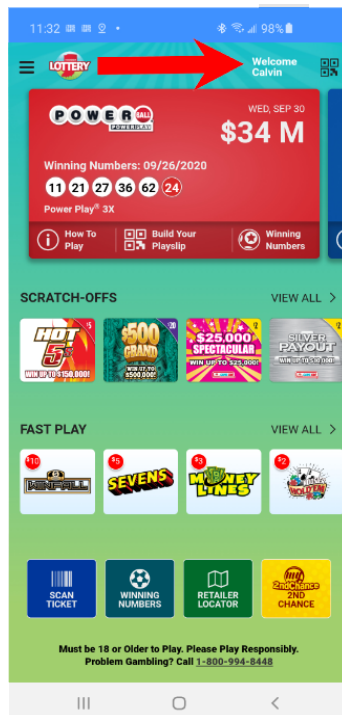
Step 1 In-App Construction

There are 3 ways you can begin your coding
1. You can start your code completely from scratch
2. You can start your code using In-App code from Laughlin Repo
3. You can use a Braze In-App template from the Braze(formally Appboy Repo)

Code can consist of the following:
- Javascript
- HTML
- CSS
- Supports local and external code libraries
  - GreenSock
  - CreateJS
  - jQuery
  - Slick
  - Pagination
  - etc

- InApp Links
  - Open the News Feed **View Feed**.
  - Close the in-app message **Close**
  - Open link using mobile browser and not webview
    - ****
  - etc

- Constraints
  - All Code and assets should be placed in the same directory
  - width: 370px;
  - height: 601px;
  - App weight is 400K

Step 2 testing

When building the bulk of your code it is convenient to:

- Test locally on Chrome browser
  - Note: Android webview are using Chrome
- Test locally on Safari browser
  - Note: IOS webviews are using Safari

Step 3 Upload to Braze

in-app Icons

Choose an in-app

Showing rows 11 to 20 of 38

Priority Sorter

© Braze 2020

Campaigns > Edit 'Hoo Knew Trivia In-App Screen 2'

Custom Code
Use HTML, CSS, Javascript, and assets

After your assets and code is in the system click here to send in-app To devices

Preview

DEVICE TYPE

Phone

We do not support previewing Custom HTML messages at this time.

We did not detect a close button in your In-App Message. Include a link with a Close Button action in your HTML to allow users to dismiss your message:

href="appboy://close">X

✏ Compose   ⚙ Settings   👁 Test

Upload only CSS, Javascript & Images

Place your Code here

⊕ LANGUAGE   Add Languages

HTML + ASSET ZIP

Upload Zip   OR

Enter Zip URL

Paste Your HTML Here

IOS DEVICE OPTIONS   Change
Sending to all devices

‹ Back   **Compose** › Delivery › Target Users › Conversions › Confirm   Forward ›   Save as Draft

Step 5 Test inside of the Hoosier App

Android and IOS

- Send Test Notifications
- Send test email and check results on a limited number of Devices
- Debug any issues and resend test until all bugs are fixed

Save as a draft when done.

Inform the Marketing Specialist that the email is ready for the internal team to review.

The team might request updates to the in-app. If so then complete the change request and **repeat step 5.**

Once the internal team has reviewed the email and is satisfied, they will it send it to the QA department for full device/client testing.

Any bug found will be for a specific device/client combinations

You should then fix any bugs reported. Then test then on the specific device/client where the bug was reported. Then you should **repeat step 2**

If no more bugs are found, then the Internal team will inform you that the email is completed and will soon be approved.

Step 7 Upload it into REPO.

After the email is officially approved add it to the repo.