**New Rapt-less GX Documentation**

**What is a Guided Experience?**

Guided Experience is an interactive video experience that serves to educate the user, solicit input from the user and usually make recommendations based on user input. It often uses humor and a very personable actor to make the experience engaging, and in the background it can leverage sophisticated logic to help the user make an informed choice.

Structurally it’s an interconnected network of videos, some of which may never be seen by the user – depending on the choices the user makes throughout the experience. The video player is embedded in a web page and often the two work together through the Brightcove API, with the page displaying supplemental information supporting the choices the user made in the video.

A GX doesn’t have to depend completely on user input, you could pass user specific data through an SSO connection or query string for example and tailor the experience based on this info.

Guided Experience has traditionally leveraged an interactive HTML layer generated by the Rapt Media solution. Moving forward we intend to use (wherever possible) this new approach which uses a JSON configuration file and the Brightcove API to build this interactive HTML layer.

***The Walmart GX*** *represents the latest version of the Raptless GX. It features two players on the page rather than one. The second player hosts the wait sequence of the video loaded into player 1. This provides for a seamless transition from main video to wait sequence. It also includes support for closed captioning with the cc state maintained across all videos. This frees the user from having to reselect their cc preferences with each new video. The Walmart GX is a demo. Not all menu items are clickable.*

*This project is on* [*Github*](https://github.com/aon-hewitt/walmartgx) *and in addition to exploring the file set you are encouraged to fork the code, enhance the tool and submit pull requests.*

**General Notes**

1. The traditional GX – Traditionally we have built GX’s from video content provided by Media Moiré, ported to Rapt Media which provided a GUI for overlaying an interactive HTML layer over the top and embedding this into a site via an iframe. We can lay additional HTML layers over the top of the iframe for further customization.
2. The new concept – One of the goals of the new GX is to remove Rapt-media from the mix and build an interactive video experience directly with a Brightcove player using the Brightcove API. This approach promises faster development without the burden of working with the Rapt Media tool which was primarily designed for non-developers. The new GX also incorporates the configuration file approach which defines in a single JSON structure all identifiable properties that make up a GX, its player environment and its html overlay. This file could be edited directly, much like a spreadsheet resulting in much faster edits by developers and non-developers. It would also be portable and hostable on CloudCMS meaning anyone could edit it from any device anywhere. In its simplest form the GX code base could load this config file and run with it - using default behaviors built in to the code base and specified in the config file. A more sophisticated solution might have custom event handlers added to the code base that preempt default behaviors. The config file identifies these event handlers and the code base would implement them.
3. The config file – The config file is a JSON file that will provide many values required by the GX. This GX object could grow and morph over time becoming more capable with every iteration while not breaking compatibility with previous GX’s. The file may be constructed and edited with a text editor, a JSON editor or plugged into a forms engine like [Alpaca Forms](http://www.alpacajs.org/) where validation and specialized controls could be employed, reducing the risk of bad user input. The config file defines things like
   1. Global GX properties that span the entire experience (like a menu bar)
   2. Videos, their Brightcove ID’s, their default end behavior, any text tracks and all onscreen elements which can be questions, answers, weightings, images, any clickable element.
   3. The array of plan recommendations and the weighted votes assigned with each user interaction
4. Events can now be triggered by interacting with onscreen elements with event handlers specified in the configuration file, or by defining text tracks which reside locally in the project and provide fast editing of events parameters. Previously events were defined in Rapt Media and edited though a slow and tedious process. The vtt Text Track files are also portable – like the config file and hostable on CloudCMS for universal editablility.
5. The code base – this is where all behaviors are processed, all events are handled and all custom logic performed. It is also where recommendations are tabulated and videos queued for display.
6. Description of every JSON property Reference - TBD
7. File structure of a GX project required folders:
   1. data folder – holds the local JSON config file if not hosted on CloudCMS
   2. vtt folder – holds a vtt file for every video that has vtt events, located at the root.
8. One of the goals in designing a new GX process was to make the development of a Guided Experience less dependent on custom – one-off code. Hard coding was avoided as much as possible, a default behavior was built in so that in its most basic form a developer need do little more than plug in a JSON configuration file and get a working Guided Experience capable of interpreting user choices and making a final recommendations. The questions and answers offered to the user, the videos available to be queued up, the final suite of recommendations and the weighting of each user click during the experience are all defined in the JSON file.
9. Every clickable element can have a default behavior defined in the config file, but also a custom event handler field which can be used for complex custom logic which supersedes the default behavior.
10. A optional slide-in overlay feature was added should the video require more real estate than what’s available within the borders of the video player.
11. The host page is made aware of every user interaction with the video and will manage the video experience by means of the Brightcove API.
12. Responsive display is configured through media queries. All onscreen elements can have one or more classes defined in the config file.
13. Events throughout the experience can be fired by user interactions with onscreen elements or by cue points defined within a vtt file. This makes creating and fine tuning of events much faster than working with them through Rapt Media or Brightcove.
14. The file structure is more modular with a plug and play methodology. The config file, the ttf file(s), the HTML content of the slide-in can all be edited by different people, hosted and served from CloudCMS or from local site files.

The configuration file:

