SOFTWARE SUPPORT OF SOME PROCESSES OF DEVELOPMENT OF MOBILE INTERACTIVE ADS

Ulyana Volodzina, Iosif Vojteshenko
MRAID Development LLC, Minsk, Belarus
Faculty of Applied Mathematics and Computer Science, Belarusian State University
E-mail: ulyanavalodzina@gmail.com, voit@bsu.by

Developing mobile interactive advertising in a game format involves several labor-intensive processes. One such process is uploading the necessary images from a psd file. This file is typically provided by designers or directly by the customer. The purpose of uploading these images is to prepare them for the subsequent layout of the advertising application's screens. To speed up the process of developing interactive advertisements, a psd-parser was selected, tested and integrated into the workflow; an application that uses the psd-parser to generate images from layers of psd-files and create pixi containers based on them was designed and implemented.

Introduction

The advertising services market has long since become a separate segment of the economy with significant volumes and high growth rates. According to some estimates, the global digital advertising market will reach 1.2 trillion USD by 2027. The production and display of advertising is actively supported by software and information and communication technologies, from graphic editors to artificial intelligence [1,2].

Mobile advertising has a number of features compared to advertising on other platforms: a limited screen; high density of content within a short time; the need and possibility of fine-tuning to the target audience; geolocation (user location) capabilities; a variety of formats (banners, videos, interactive ads, etc.); a variety of distribution channels (mobile applications, mobile sites, social networks and messengers).

Mobile interactive advertising in game format (playable ads) is a mini-game that demonstrates the basic functions of the game. The duration of the user's interaction with such ads is designed for 15-60 seconds on average.

Interactive advertising in game format is a productive way to attract customers.

I. MRAID standard and dashboard.mraid.io platform

The MRAID (Mobile Rich Media Ad Interface Definitions) standard provides a standardized interface between ads and mobile applications, which simplifies the process of creating, placing and measuring ad campaigns on mobile devices [3].

The interface defined by the MRAID standard includes the following main elements: an ad container; the MRAID API; a set of methods for handling user gestures; MRAID extensions; mechanisms for controlling access to data collected in an ad; and requirements for media content elements to be compatible with mobile devices and applications.

Dashboard.mraid.io is a platform for developing interactive mobile ads that allows the creation of ads that can interact with the user, such as banner ads, drop-down ads, promotional videos, etc. [4]. The platform is developed by Mraid.io and supports the MRAID standard.

II. Peculiarities and development process of interactive mobile advertising

The peculiarities of the development of such advertising lies in several factors.

First, the advertising application should be quite "lightweight", no more than 2-5 megabytes, taking into account images, audio and possibly video materials. This severely limits the developer, forcing him to increase image compression, crop audio files, reduce the frame rate in video, etc.

Second, the advertising customer usually needs multiple versions of the advertising application to upload them to different ad networks, and each may have different requirements for the files to be uploaded.

Third, advertising products must be developed in the shortest possible time. The usual limit is no more than a week (five working days).

The process of developing interactive mobile advertising may involve several steps.

1. Defining goals and objectives, identifying the target audience, and conceptualizing the advertisement. At this stage, the advertiser usually works with an agency or design team to determine the overall direction and style of the advertisement. .
2. Creating the layout of the advertisement in a graphics editor such as Adobe Photoshop or Adobe Illustrator. The layout may contain images, text, buttons, and other elements of interactivity.
3. Developing code using HTML, CSS, JavaScript or other programming languages.
4. Testing on a variety of mobile devices and browsers, including using MRAID Validator.
5. Uploading to the ad network. Advertiser can select various ad display parameters such as targeting, budget and display period.

One of the labor-intensive development processes is uploading the necessary images from

a psd-file provided by designers or directly by the customer, for the purpose of subsequent layout of the screens of the advertising application. Practice shows that the developer spends up to a third of his time on this. The sequence of his actions is as follows: uploading individual layers with the help of Adobe Photoshop in .png image format, renaming all images in the same style accepted in the company, creating pixi containers [5], which contain the following information: the name of the container (unique), the name of the image, the position on the screen relative to the center - all this information already initially contains a PSD file. These actions can be performed by a software application. By automating the development process, we can minimize the human factor in development, get rid of typos, and significantly speed up the development process.

The application was developed for integration into the dashboard.mraid.io platform.

The central stage of automated layout is parsing (analyzing the syntactic structure) of psd-images.

The most famous is the parser of psd-files by Adobe [6]. However, it does not create layout in the format used in interactive advertising. In addition, its integration into the dashboard.mraid.io platform would be difficult.

Since the Dashboard.mraid.io platform is developed using node.js and its visual part using the vue.js framework, we turned our attention to one of the parsers whose program code is posted on github.com. It is written in JavaScript and consists of several classes, the main ones being PSD and Image classes. The parsing scheme is shown in the Figure.

The functionality of this parser turned out to be easy to integrate with the capabilities of the open source library for working with 2D graphics in the browser Pixi.js [5].

The integration of the psd-parser and the pixi.js programmatic library for their joint use in ad layout within the Dashboard.mraid.io platform has a number of features.

1. When transferring layout from psd layout to Pixi.js containers, it is necessary to take into account the differences in coordinate systems. In psd format, layer coordinates are set relative to the upper left corner of the image, while in Pixi.js it is customary to use the center point of the container as the initial coordinate point.
2. The layer order in Pixi.js will be reflected in reverse order compared to the layer order in the psd layout. This is because in psd format, layers are arranged from top to bottom, while in Pixi.js, containers are created in the order listed in the code.
   To flip the layer tree from psd format to Pixi.js format, the psd layers were traversed in depth, starting from the root layer.
3. Photoshop allows you to create layers with the same name in the same document. This is possible because in a psd file, layers and layer groups are identified not only by name, but also by other properties such as layer IDs and nesting level. Thus, even if there are layers with the same names in the document, they will still be correctly identified.
   However, in Pixi.js, containers and objects are identified only by their name.Therefore, layer names had to be checked to see if they matched those already extracted and converted if necessary.

With these features in mind, an application was designed and implemented that uses psd-parser to generate images from layers and create pixi containers. This application was integrated into dashboard.mraid.io by adding new containers and handlers to the vue.js project that call the necessary methods.

III. Conclusion

To speed up the process of developing interactive advertisements, a psd-parser was selected, tested and integrated into the development workflow. An application was designed and implemented that uses psd-parser to generate images from psd-file layers and create pixi containers based on them. This application was integrated into dashboard.mraid.io by adding new containers and handlers to the vue.js project that call the necessary methods. By doing so, the time spent in the image processing and layout creation phases was significantly reduced.

1. Yogesh K. Dwivedi. Setting the future of digital and social media marketing research: Perspectives and research propositions / Yogesh K. Dwivedi, Elvira Ismagilova, D. Laurie Hughes at el. // International Journal of Information Management, Volume 59, 2021, 102168. https://doi.org/10.1016/j.ijinfomgt.2020.102168.
2. Biao Gao. Artificial Intelligence in Advertising: Advancements, Challenges, and Ethical Considerations in Targeting, Personalization, Content Creation, and Ad Optimization / Biao Gao, Yiming Wang, Huiqin Xie, Yi Hu, Yi Hu// SAGE Open. October-December 2023: 1–20. doi/10.1177/21582440231210759
3. Mobile Rich Media Ad Interface Definition (MRAID) Version 3.0 [Electronic resource]. – Mode of access: https://www.iab.com/wp-content/uploads/2017/07/MRAID_3.0_FINAL.pdf. – Date of access: 25.09.2024
4. Dashboard Mraid.io [Electronic resource]. – Mode of access: https://mraid.io/#dashboard. – Date of access: 25.09.2024.
5. PixiJS Documentation [Electronic resource]. – Mode of access: https://pixijs.download/dev/docs/index.html. – Date of access: 25.09.2024.
6. .PSDParser [Electronic resource]. – Mode of access: https://developer.adobe.com/experience-manager/reference-materials/6-5/javadoc/org/apache/tika/parser/image/PSDParser.html. – Date of access: 25.03.2024.