http://doi.org/10.35784/iapgos.3734

received: 12.06.2023 | revised: 19.09.2023 | accepted: 14.11.2023 | available online: 20.12.2023

INTERACTION METHOD BETWEEN WEBVIEW OBJECTS IN HYBRID JAVA APPLICATIONS

Denys Ratov, Oleh Zakhozhai

Volodymyr Dahl East Ukrainian University, Faculty of Information Technology and Electronics, Department of Information Technology and Programming, Kyiv, Ukraine

Abstract. *The article deals with method of interaction between JavaScript objects of different web pages in hybrid java applications. To solve this problem, the WebView component is used, its functionality to provide an interface for JavaScript objects, as well as the SharedPreferences global settings class, with its ability to store data in local storage. The software implementation is considered and the results of the practical use of the developed method of interaction between JavaScript WebView objects in the form of a hybrid Java application embedded in the electronic university system – the Timetable SNU electronic timetable module are presented.*

Keywords: Java, hybrid application, JavaScript, interaction, WebView

METODA INTERAKCJI POMIĘDZY OBIEKTAMI WEBVIEW W HYBRYDOWYCH APLIKACJACH JAVA

Streszczenie. *Artykuł dotyczy sposobu interakcji między obiektami JavaScript różnych stron internetowych w hybrydowych aplikacjach Java. Do rozwiązania tego problemu wykorzystywany jest komponent WebView, którego funkcjonalność zapewnia interfejs dla obiektów JavaScript, a także klasa ustawień globalnych SharedPreferences z możliwością przechowywania danych w lokalnym magazynie. Rozważono implementację oprogramowania oraz przedstawiono wyniki praktycznego wykorzystania opracowanej metody interakcji między obiektami JavaScript WebView w postaci hybrydowej aplikacji Java osadzonej w elektronicznym systemie uczelni – module elektronicznego planu lekcji Timetable SNU.*

Słowa kluczowe: Java, aplikacja hybrydowa, JavaScript, interakcja, WebView

Introduction

In today's digital world, where smartphones have become an integral part of our daily lives, mobile applications are becoming increasingly popular. This is especially true in the university environment, where effective scheduling is critical for students and faculty. Mobile applications allow students and teachers to quickly and conveniently view the class schedule without the need to use a computer or view a paper schedule. This brings convenience and flexibility to users, allowing for more efficient time and resource management. Using the mobile application, the class schedule can be updated in real time. This is especially important in case of changes in schedule or audience information. Students and teachers can quickly get updated information without having to check email. In this regard, writing a mobile application for the schedule of classes at the university becomes an urgent task [23].

Writing cross-platform applications requires a serious approach to planning future functions and capabilities of the program, since each of the selected target platforms [20], both theoretically and in practice, can impose its own limitations due to both the features of the target platform architecture and additional restrictions put forward by the developer company or the company that owns this platform.

Creating apps for Android is a complex process that requires detailed planning and has its own characteristics [18, 19]. One of them is the variety of mobile devices on which the application will run. When developing, it is necessary to take into account that users have different technical characteristics of their devices, and also use different versions of the Android operating system.

Today, if not all, then most Android devices have access to the Internet. Therefore, a large number of mobile applications, one way or another, interact with the Internet environment: they download files, log in and receive information from external web services, etc.

Hybrid applications have gained great popularity in the development of software for devices with the Android operating system. They combine the properties of both native and web applications [15, 16]. As a native app, it can be distributed to users through the app store, and it can also take advantage of numerous mobile device features. As a web application, it consists of HTML, CSS and Javascript files [17].

The benefits of this type of application include:

- many applications and interactive components can be written in JavaScript [21] for all mobile platforms;
- applications can use mobile device features such as camera, accelerometer, and others;
- all HTML, CSS and JavaScript files can be updated without waiting for a new version of the application to be approved.

To work with the network in Android, it is possible to use several methods. To get data from a specific Internet resource, you can use classes HttpUrlConnection (for HTTP protocol) and HttpsUrlConnection (for HTTPS protocol) from the Java Standard Library.

When developing a hybrid application to support the concept of Model-View-Controller [11], it is necessary to respect the separation of application data and control logic into three separate components: model, view and controller. The data received during the application from the activity web page is stored in the content model (JavaScript objects) of this web page [4]. Therefore, when loading other web content into the same activity, data is lost from the previous content model. So, for example, having loaded the content of a web page with authorization into an activity, it becomes necessary to transfer the result of the obtained authorization parameters to an activity with new web content. Therefore, there is a need to create a method of interaction between JavaScript objects of different web pages on activity in hybrid Java applications.

1. Application architecture

In the process of creating a hybrid application, among the standard elements, an important role is played by the WebView component [3], which is a full-fledged browser implemented as a subclass of View and capable of loading content from a specific URL, and therefore is designed to render html code. The operation of the component is based on the free engine for displaying WebKit web pages, which was developed by Apple. Using the WebKit engine ensures that the content will be displayed in the same way as in other browsers built on this engine – Google Chrome and Safari. Thanks to this, WebView can be used as a custom web browser, viewing content from the Internet through it [10].

The WebView component has many properties and methods that allow you to create the full functionality of a regular browser, and makes it possible to bind JavaScript code to Java code using an interface class [20]. To do this, the WebView component program uses the addJavaScriptInterface method, which is passed a class that provides an interface for JavaScript, and a name that will be used to display the instance in JavaScript (for example, "iAndroid"). In the MVC concept, this ensures the creation of a controller component.

At the beginning of the application, the WebView component is initialized in the MainActivity class, using the identifier previously defined in activity_main.xml (Fig. 1). Then, to enable the existing WebView element to be associated with JavaScript objects, we set the interface class to the WebView [14].

```
browser  = (WebView) findViewById(R.id.webBrowser);
browser.addJavascriptInterface(new WebAppInterface(this),
                                    "iAndroid");
```

Fig. 1. Setting the interface class in WebView

This will create an interface named iAndroid that will be available to JavaScript objects running in the content of the page loaded in the WebView component. With the help of the added interface class, the following possibilities appear:

1) call from the JavaScript module to the method described in the Java code;
2) execute a method from Java code, which is described in the JavaScript module.

Let's describe the WebAppInterface interface class with methods, the functionality of which is necessary in the JavaScript module code (Fig. 2).

```
public class WebAppInterface {
    private Context mContext;
    WebAppInterface(Context context) {   mContext = context; }
    @JavascriptInterface
    public void setParam(String key, String val) {
        String lastVal = "";
        prefEditor = settings.edit();
        if ( val.length() > 0 && ( key.equals("TM_idGroup") ||
            key.equals("TM_group") ) )
        {
            lastVal = settings.getString(key, "");
            prefEditor.putString(key,
                (lastVal.length() > 0
                    ? lastVal + ", "
                    : "") + val);
        }
        else {
            prefEditor.putString(key, val);
        }
        prefEditor.apply();

        if (key.equals("TM_FioUser")){
            Toast.makeText(mContext, (val.length() > 0
                                        ? " збережено "
                                        : " видалено ") + val,
                                    Toast.LENGTH_SHORT).show();
        }
    }
    @JavascriptInterface
    public String getParam(String key){
        return settings.getString(key, "");
    }
}
```

Fig. 2. Interface class WebAppInterface

On the content side of the page loaded in the WebView, to interact with JavaScript objects with an instance of the Java class, we use the name of the created interface "iAndroid".

```
System.saveUser = function(){
let iStd    = $('#iStd'),   iTch    = $('#iTch'),
    inpFio  = $('#inpFio'), selFio  = $('#selFio');

    iAndroid.setParam('TM_sigUser', iStd.checked ? 's' : 't');
    if (iStd.checked){
        iAndroid.setParam('TM_FioUser', inpFio.value);
        iAndroid.setParam('TM_idGroup',
            selFio.options[selFio.selectedIndex].dataset.id_group);
        iAndroid.setParam('TM_group',
            selFio.options[selFio.selectedIndex].dataset.group);
    }
    else {
        iAndroid.setParam('TM_FioUser',
            selFio.options[selFio.selectedIndex].dataset.fio);
        iAndroid.setParam('TM_dolTeacher',
            selFio.options[selFio.selectedIndex].dataset.dol);
        iAndroid.setParam('TM_idUser',
            selFio.options[selFio.selectedIndex].dataset.id_fio);
    }
}
```

Fig. 3. An example of using the setParam method in the iAndroid interface

To pass parameters from web content objects to the iAndroid interface, we use its setParam method (Fig. 3). The name of the key and its value are passed as parameters. For reading, we use the getParam method (Fig. 4) with the key name parameter.

To maintain the separation of application data and control logic in the iAndroid interface, it becomes necessary to store data obtained from web content objects for further use. For example, user data, configuration settings, etc.

```
window.addEventListener( 'load', ()=>{
    System.status       = iAndroid.getParam('TM_sigUser');
    System.idfio        = iAndroid.getParam('TM_idUser');
    System.fio          = iAndroid.getParam('TM_FioUser');
    System.dol          = iAndroid.getParam('TM_dolTeacher');
    System.isOpenPanel  = iAndroid.getParam('TM_isOpenPanel');
})
```

Fig. 4. An example of using the getParam method in the iAndroid interface

To do this, Android has the concept of Preferences or settings. Settings are a group of key-value pairs that are used by an application. Settings are stored in xml files in unencrypted form in local storage. They are invisible, so they are not available to the average user.

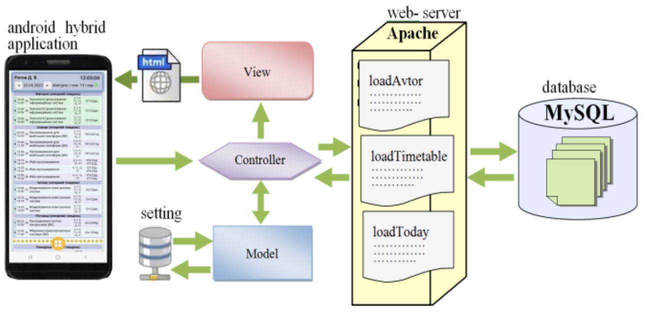


Fig. 5. Scheme of organizing support for the MVC concept in the context of a hybrid application

To store data from the content model, we use the android.content.SharedPreferences class. To manage the settings, an object of the SharedPreferences.Editor class is used. Asynchronous data storage allows key-value data to be stored on the mobile device, ensuring that the data is persistent even after the application is closed. The mechanism works asynchronously, which avoids blocking the user interface when saving or retrieving data. The use of asynchronous storage ensures the creation of model components in the MVC concept (Fig. 5).

A diagram of the process of internal interaction between JavaScript objects, an instance of the iAndroid interface class, and an instance of the SharedPreferences class is shown in Fig. 6. To make application-level requests to the server from JavaScript objects, the XMLHttpRequest API is used, which makes it possible to create asynchronous AJAX requests [2]. At the same time, the WebView component can reload its content using the loadUrl method.

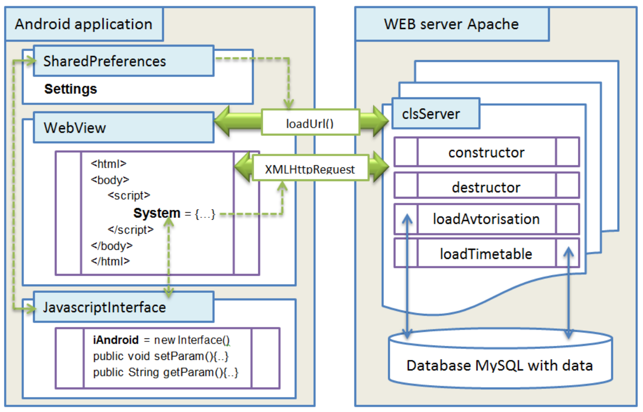


Fig. 6. Diagram of the interaction between JavaScript objects, an instance of the iAndroid interface class, and an instance of the SharedPreferences class

When forming a request to the server from the WebView component or JavaScript objects of any loaded content, the local storage (an instance of the SharedPreferences class) is first accessed. This makes it possible to get or save parameter values not only for cross-platform interaction between JavaScript objects and Java classes, but also between JavaScript objects of different content that is loaded into the same WebView component.

2. Results of the implementation of the method

The considered method of data exchange between WebView objects in hybrid Java applications has been tested in practice. It is implemented in the system of the electronic university, namely the electronic timetable module Timetable SNU – Volodymyr Dahl East Ukrainian University (Kyiv).

The application is complex, that is, it follows the client-server architecture pattern [1], which is one of the software architectural patterns and has become the dominant concept in the creation of distributed network applications and involves interaction and data exchange between them.

The server part of the schedule application of the busy university is responsible for the processes:

- Data management: saving and managing the schedule, information about classes, groups of students, teachers and other data.
- Data exchange via API: providing an API (application programming interface) that allows the client part of the application to receive and send data. This may include requests to receive a class schedule, update class data, or interact with other features of the application.
- Security and data protection: a centralized server part implies better data security control than a distributed system, where each element can have its own security flaws.
- Scalability and performance: the server simultaneously processes many requests and quickly responds to them, and also has mechanisms for caching and optimizing requests to the database. Scalability also includes the ability to easily expand the server infrastructure, for example, adding additional servers or using cloud services to ensure the reliability and availability of the application. This makes it possible to ensure the operation of the application when the load and the number of users increase.

The client part of the application implements software functionality:

- User interface: the application has a convenient interface that allows students and teachers to easily view the busy schedule by selecting a group or teacher through search, view busy details and changes. It is important that the interface is intuitive and provides convenient navigation through the application.
- Local storage of settings: the application can save the user's settings and personal parameters locally on the device. This allows you to store a user object, preferred settings and other personalized data without the need to constantly request them from the server.
- Interaction with the server: the application interacts with the server through the API, receiving data about the schedule, lists of teachers and groups, and other additional information.
- Mobile capabilities: the application can use the device's mobile capabilities, such as geolocation, camera, notifications, and others. For example, students can be given the opportunity to view the location of classrooms on a map, and teachers – the opportunity to download materials for each lesson.

  Local data processing: the application can perform some data processing locally on the device without the need for constant access to the server. For example, it can filter the schedule according to certain criteria.

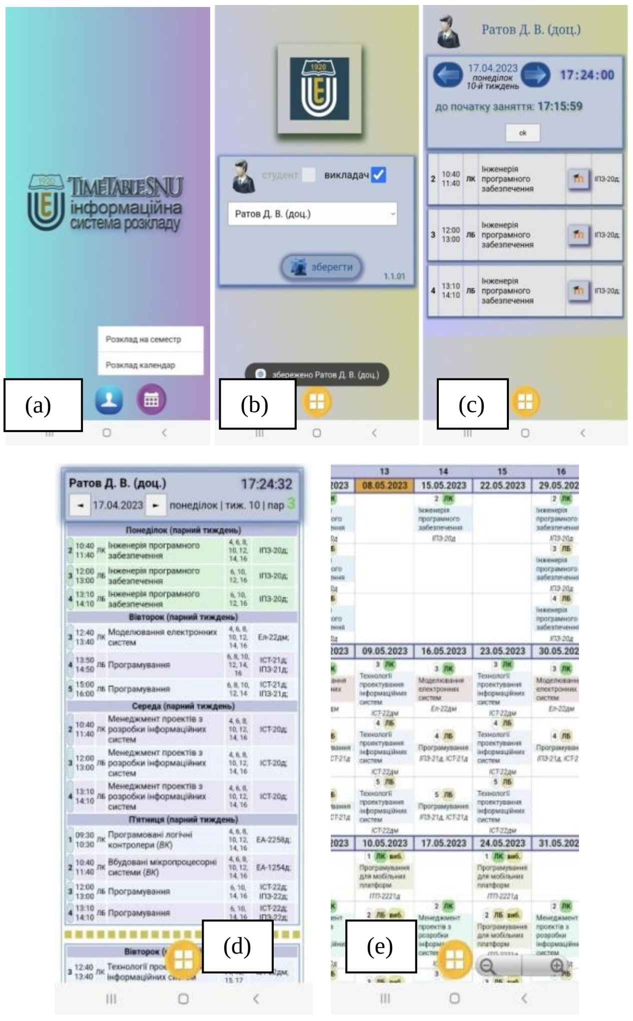

(a)
TimeTableSNU інформаційна система розкладу
Розклад на семестр
Розклад календар
(b)
студент
викладач
Ратов Д. В. (доц.)
зберегти
1.1.01
збережено Ратов Д. В. (доц.)
(c)
Ратов Д. В. (доц.)
17.04.2023 понеділок 10-й тиждень
17:24:00
до початку заняття: 17:15:59
ok
Інженерія програмного забезпечення
(d)
Ратов Д. В. (доц.)
17:24:32
17.04.2023
понеділок | тиж. 10 | пар 3
Понеділок (парний тиждень)
Вівторок (парний тиждень)
Середа (парний тиждень)
П'ятниця (парний тиждень)
Програмування
(e)
08.05.2023
15.05.2023
22.05.2023
29.05.2023

Fig. 7. Timetable SNU electronic timetable user interface

In terms of user interface (UI), the university class schedule application is divided into 2 Activities with data display that simplifies user interaction with the application (Fig. 7).

The main Activity contains application control components (Fig. 7(a)). The second Activity contains the WebView component, in which web pages of all functionality with different JavaScript objects are loaded from the server. These are the pages:

1) user authorization (Fig. 7(b));
2) a form with a schedule for the current or selected day (Fig. 7(c));
3) a form with a schedule for the entire semester (Fig. 7(d));
4) a form with a schedule in the form of a calendar (Fig. 7(e)).

The interaction between JavaScript objects of different pages loaded into the same WebView component is carried out through a chain of links: "iAndroid" – an instance of the interface class → "Settings" – an instance of the SharedPreferences class → "System" JavaScript object.

The program has created an access control system, which allows you to control access to resources and services, set access rights for different users or user groups. To do this, an object of the authorization and authentication system was created on the web page, which provides access only to authorized users. Having loaded the content of the web page with authorization into the activity (Fig. 7(b)), the received authorization parameters become available through the "Settings" JavaScript object to objects of other web content – the schedule page of the current day (Fig. 7(c)) and the schedule page for the semester (Fig. 7(d)). On the web page with the schedule form for the current or selected day (Fig. 7(c)), JavaScript objects are created that execute asynchronous Ajax requests of the application level to the server. At the same time, the "Settings" object, an instance of the SharedPreferences class, participates in the requests.

Asynchronous Ajax requests of the application layer to the server allow students and teachers to connect to video conferencing platforms for remote classes. Also, the teacher has the opportunity to monitor their classes: the web page contains JavaScript objects that allow you to get a list of students who have attended the selected classes.

3. Conclusions

The method of data exchange between JavaScript objects in the WebView of hybrid Java applications described in the article makes it possible to get or save parameter values not only for cross-platform interaction between JavaScript objects and Java classes, but also between JavaScript objects of different content that is loaded into the same WebView component . This approach uses a modern and proven technology stack, which allows you to quickly develop and easily maintain the interaction of hybrid applications.

The method considered in the work made it possible to organize the interaction of JavaScript and Java objects in the developed Timetable SNU electronic timetable application. The application contributes to more efficient management of the class schedule at the university, which is critical for students and teachers. With the app, users can quickly and conveniently view class schedules on their mobile devices, eliminating the need to use a computer or view a paper timetable. This gives flexibility and efficient time and resource management. In addition, the chosen architecture allows you to always get up-to-date information about the schedule, which is especially important when the schedule changes. The mobile application allows you to get updated information quickly and conveniently without checking email or following ads. As a result of the introduction of a mobile timetable application at the university, students and teachers will be able to plan their time more efficiently, and the university will receive a convenient and modern tool for information support of the educational process.

The method of interaction between JavaScript objects and Java classes proposed in the work had a positive impact on the performance of the application's user interface and the possibility of scaling the functionality of the information system itself when implementing hybrid technologies in Java applications.

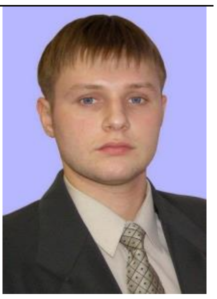
References

[1] Architecture Client-Server [https://ru.frwiki.wiki/wiki/Client-serveur] (available: 28.05.2020).
[2] Crane D., Pascarello E.: Ajax in action. Williams, Moscow 2006.
[3] Deitel P., Deitel H., Deitel E.: Android for Developers. Peter, St. Petersburg 2015.
[4] ECMAScript Language Specification – ECMA-262 Edition 5.1 [https://262.ecma-international.org/5.1/] (available: 28.05.2020).
[5] Expo – Create amazing apps that run everywhere [https://docs.expo.dev/get-started/expo-go/] (available: 28.05.2020).
[6] Expo Application Services (EAS) [https://expo.dev/eas] (available: 28.05.2020).
[7] Expo Go – Expo documentation [https://docs.expo.dev/get-started/expo-go/] (available: 28.05.2020).
[8] Griffiths D., Griffiths D.: Android Programming. O'Reilly Media, 2016.
[9] Kotlin Multiplatform – Kotlin Documentation [https://kotlinlang.org/docs/multiplatform.html] (available: 28.05.2020).
[10] Mednieks Z., Dornin L., Meike G. B.: Masumi Nakamura: Programming Android. O'Reilly Media, 2013.
[11] MVC architecture [http://www.gwtproject.org/articles/mvp-architecture.html] (available: 28. 05.2020).
[12] Niemeyer P.: Java Programming. Ekmo, Moscow 2014.
[13] Out-of-Tree Platforms – React Native [https://reactnative.dev/docs/out-of-tree-platforms] (available: 28.05.2020).
[14] Phillips B., Stuart K., Marsicano K.: Android. Programming for professionals. Peter, St. Petersburg 2017.
[15] Ratov D.: Architectural paradigm of the interactive interface module in the cloud technology model. Applied Computer Science 16(4), 2020, 48–55 [http://doi.org/10.23743/acs-2020-28].
[16] Ratov D.: Integration with the software interface of the com server for authorized user. Applied Computer Science 17(2), 2021, 5–13 [http://doi.org/10.23743/acs-2021-09].
[17] Ratov D.: Model of the user interface module of the information web system. Mathematical machines and systems 4, 2021, 74–81.
[18] React Native – Learn once, write anywhere [https://reactnative.dev/docs/getting-started] (available: 28.05.2020).
[19] React Navigation – Routing and navigation for Expo and React Native apps [reactnavigation.org] (available: 28.05.2020).
[20] Schildt H.: The Complete Guide. Williams, Moscow 2015.
[21] Stefanov S.:. JavaScript. Patterns. O'Reilly Media, 2010.
[22] vscode.dev – Visual Studio Code for the Web [https://code.visualstudio.com/blogs/2021/10/20/vscode-dev] (available: 28.05.2020).
[23] Zakhozhai O., Lyfar V., Ivanov V., Baturin O.: Uniform interaction model of educational process agents in unified management system of higher education institution. Information Technologies and Learning Tools 78(4), 2020, 266–277.

Ph.D. Denys Ratov
e-mail: denis831102@gmail.com

Assistant professor in Department of Information Technologies and Programming at the Faculty of Information Technologies and Electronics, Volodymyr Dahl East Ukrainian National University, Ukraine. The field of scientific interests is mathematical and computer modeling, decision support systems, software development using information technologies in applied fields and production.

http://orcid.org/0000-0003-4326-3030



Ph.D. Oleh Zakhozhai
e-mail: zakhozhay.oleg@gmail.com

Professor in Department of Information Technologies and Programming at the Faculty of Information Technologies and Electronics, Volodymyr Dahl East Ukrainian National University, Ukraine.
The author's area of research is focused on patterns recognition and data processing in complex systems, mathematical and computer modeling and visualization.

http://orcid.org/0000-0002-9078-3242