HelpDeskEddy чат виджет в WebView

iOS

ViewController.swift

```
import UIKit
import WebKit

class ViewController: UIViewController, WKNavigationDelegate, WKUIDelegate {
    var webView: WKWebView!
    var documentInteractionController: UIDocumentInteractionController!

    let systemHost = "domain.com"

    override func viewDidLoad() {
        super.viewDidLoad()
        self.webView.uiDelegate = self


        let url = URL(string: "https://" + systemHost + "/ru/omnichannel/chat" )!
        webView.load(URLRequest(url: url))

        let refresh = UIBarButtonItem(barButtonSystemItem: .refresh, target: webView, action:
#selector(webView.reload))
        toolbarItems = [refresh]
        navigationController?.isToolbarHidden = false
    }

    override func loadView() {
        webView = WKWebView()
        webView.navigationDelegate = self
        view = webView
    }

    func webView(_ webView: WKWebView, decidePolicyFor navigationAction: WKNavigationAction, decisionHandler:
@escaping (WKNavigationActionPolicy) -> Void) {
        if let url = navigationAction.request.url {
            if (url.absoluteString.range(of: "/omnichannel/download") != nil ) {
                let downloadTask = URLSession.shared.downloadTask(with: url) {
                    urlOrNil, responseOrNil, errorOrNil in
                    guard let fileURL = urlOrNil else { return }
                    do {
                        let fileName = responseOrNil?.suggestedFilename ?? fileURL.lastPathComponent
                        var saveTo = FileManager.default.temporaryDirectory.appendingPathComponent("hde" +.
fileName)
                        var counter = 0

                        while (FileManager.default.fileExists(atPath: saveTo.path)) {
                            counter += 1
                            saveTo = FileManager.default.temporaryDirectory.appendingPathComponent(String(counter)
+ "-hde-" + fileName)
                        }

                        try FileManager.default.moveItem(at: fileURL, to: saveTo)
                        DispatchQueue.main.async() {
                            self.documentInteractionController = UIDocumentInteractionController.init(url: saveTo)
                            self.documentInteractionController?.delegate = self
                            self.documentInteractionController?.presentPreview(animated: true)
                        }
                    } catch {
                        print ("file error: \(error)")
                    }
                }
                downloadTask.resume()
                decisionHandler(.cancel)
                return
            } else if (url.host?.range(of: systemHost) != nil) {
                decisionHandler(.allow)
                return
            } else {
                UIApplication.shared.open(url)
```

```
                decisionHandler(.cancel)
                return
            }
        }
        decisionHandler(.cancel)
    }
}
extension ViewController: UIDocumentInteractionControllerDelegate {
    func documentInteractionControllerViewControllerForPreview(_ controller: UIDocumentInteractionController) ->
UIViewController {
        return self
    }
    func documentInteractionControllerDidEndPreview(_ controller: UIDocumentInteractionController) {
        do {
            if let url = controller.url {
                try FileManager.default.removeItem(atPath: url.path)
            }
        } catch {
            print ("file error: \(error)")
        }
    }
}
```

Android

MainActivity.java

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.ActivityCompat;
import android.Manifest;
import android.content.DialogInterface;
import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Bundle;
import android.view.Window;
import android.webkit.ValueCallback;
import android.webkit.WebChromeClient;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.annotation.*;
import android.app.*;
import android.os.*;
import android.webkit.*;

public class MainActivity extends AppCompatActivity {
    public static final int REQUEST_CODE_LOLIPOP = 1;
    private final static int RESULT_CODE_ICE_CREAM = 2;
    private WebView webView;
    private ValueCallback<Uri[]> mFilePathCallback;
    private ValueCallback<Uri> mUploadMessage;
    private String url = "https://support.helpdeskeddy.com/ru/omnichannel/chat";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_NO_TITLE);
        try {
            this.getSupportActionBar().hide();
        } catch (NullPointerException e) {
        }
        setContentView(R.layout.activity_main);
        webView = findViewById(R.id.webview);
        setUpWebViewDefaults(webView);
        webView.loadUrl(url);
        checkPermission();
        webView.setWebChromeClient(new WebChromeClient() {
            private String TAG;
            // For Android 3.0+
            public void openFileChooser(ValueCallback<Uri> uploadMsg) {
                mUploadMessage = uploadMsg;
                Intent i = new Intent(Intent.ACTION_GET_CONTENT);
                i.addCategory(Intent.CATEGORY_OPENABLE);
                i.setType("*/*");
                startActivityForResult(Intent.createChooser(i, "File Chooser"),
```

```
                    RESULT_CODE_ICE_CREAM);
        }

        // For Android 3.0+
        public void openFileChooser(ValueCallback uploadMsg, String acceptType) {
            mUploadMessage = uploadMsg;
            Intent i = new Intent(Intent.ACTION_GET_CONTENT);
            i.addCategory(Intent.CATEGORY_OPENABLE);
            i.setType("*/*");
            startActivityForResult(Intent.createChooser(i, "File Chooser"),
                    RESULT_CODE_ICE_CREAM);
        }

        //For Android 4.1
        public void openFileChooser(ValueCallback<Uri> uploadMsg, String acceptType, String capture) {
            mUploadMessage = uploadMsg;
            Intent i = new Intent(Intent.ACTION_GET_CONTENT);
            i.addCategory(Intent.CATEGORY_OPENABLE);
            i.setType("*/*");
            startActivityForResult(Intent.createChooser(i, "File Chooser"),
                    RESULT_CODE_ICE_CREAM);
        }

        //For Android5.0+
        public boolean onShowFileChooser(
                WebView webView, ValueCallback<Uri[]> filePathCallback,
                FileChooserParams fileChooserParams) {
            if (mFilePathCallback != null) {
                mFilePathCallback.onReceiveValue(null);
            }
            mFilePathCallback = filePathCallback;
            Intent contentSelectionIntent = new Intent(Intent.ACTION_GET_CONTENT);
            contentSelectionIntent.addCategory(Intent.CATEGORY_OPENABLE);
            contentSelectionIntent.setType("*/*");
            Intent[] intentArray;
            intentArray = new Intent[0];
            Intent chooserIntent = new Intent(Intent.ACTION_CHOOSER);
            chooserIntent.putExtra(Intent.EXTRA_INTENT, contentSelectionIntent);
            chooserIntent.putExtra(Intent.EXTRA_TITLE, "File chooser");
            chooserIntent.putExtra(Intent.EXTRA_INITIAL_INTENTS, intentArray);
            startActivityForResult(chooserIntent, REQUEST_CODE_LOLIPOP);
            return true;
        }
    });
    webView.setDownloadListener(new DownloadListener() {
        @Override
        public void onDownloadStart(String url, String userAgent, String contentDescription,
                                    String mimetype, long contentLength) {
            DownloadManager.Request request = new DownloadManager.Request(Uri.parse(url));
            request.allowScanningByMediaScanner();
            request.setNotificationVisibility(
                    DownloadManager.Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED);
            String fileName = URLUtil.guessFileName(url, contentDescription, mimetype);
            request.setDestinationInExternalPublicDir(Environment.DIRECTORY_DOWNLOADS, fileName);
            DownloadManager dManager = (DownloadManager) getSystemService(DOWNLOAD_SERVICE);
            dManager.enqueue(request);
        }
    });
}

@TargetApi(Build.VERSION_CODES.HONEYCOMB)
private void setUpWebViewDefaults(WebView webView) {
    WebSettings settings = webView.getSettings();
    settings.setJavaScriptEnabled(true);
    settings.setDomStorageEnabled(true);
    settings.setUseWideViewPort(true);
    settings.setLoadWithOverviewMode(true);
    webView.setWebViewClient(new WebViewClient());
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    switch (requestCode) {
        case RESULT_CODE_ICE_CREAM:
            Uri uri = null;
```

```
                if (data != null) {
                    uri = data.getData();
                }
                mUploadMessage.onReceiveValue(uri);
                mUploadMessage = null;
                break;
            case REQUEST_CODE_LOLIPOP:
                Uri[] results = null;
                if (resultCode == Activity.RESULT_OK) {
                    if (data != null) {
                        String dataString = data.getDataString();
                        if (dataString != null) {
                            results = new Uri[]{Uri.parse(dataString)};
                        }
                    }
                }
                mFilePathCallback.onReceiveValue(results);
                mFilePathCallback = null;
                break;
        }
    }

    protected void checkPermission() {
        if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
            if (checkSelfPermission(Manifest.permission.WRITE_EXTERNAL_STORAGE) !=
PackageManager.PERMISSION_GRANTED) {
                if (shouldShowRequestPermissionRationale(Manifest.permission.WRITE_EXTERNAL_STORAGE)) {
                    AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);
                    builder.setMessage("Write external storage permission is required.");
                    builder.setTitle("Please grant permission");
                    builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialogInterface, int i) {
                            ActivityCompat.requestPermissions(
                                    MainActivity.this,
                                    new String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE},
                                    200
                            );
                        }
                    });
                    builder.setNeutralButton("Cancel", null);
                    AlertDialog dialog = builder.create();
                    dialog.show();
                } else {
                    ActivityCompat.requestPermissions(
                            MainActivity.this,
                            new String[]{Manifest.permission.WRITE_EXTERNAL_STORAGE},
                            200
                    );
                }
            } else {
            }
        }
    }
}
```