// ALTRegisterFiles.m

// Created by DimaM on 21.11.19. // Copyright © 2019 Softorino. All rights reserved. #import "ALTRegisterFiles.h" #import "ALTRegisterEncryptor.h" #import "ALTDownloader.h" #ifdef USE_TEST_SERVER static NSString * licenseFilename() { return @".testlicense"; } if (!encData) return nil; NSData *data = [ALTLicenseFileData altDecryptFileData:encData]; static NSString * b64decode(NSString *b64s) { if (!data) return nil; NSData *data = [[NSData alloc] initWithBase64EncodedString:b64s options:NSDataBase64DecodingIgnoreUnknownCharacters]; return [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding]; NSPropertyListFormat format; // ALTLicensesFiles.m // ALTRegisterFiles.m // Created by DimaM on 23.06.20. // Created by DimaM on 21.11.19. // Copyright © 2019 Softorino. All rights reserved. // Copyright © 2020 Softorino. All rights reserved. #import "ALTRegisterFiles.h" #import "ALTLicensesFiles.h" #import "ALTRegisterEncryptor.h" #import "ALTRegisterEncryptor.h" #import "ALTDownloader.h" #import "ALTDownloader.h" #ifdef USE_TEST_SERVER #ifdef USE_TEST_SERVER static NSString * licenseFilename() { return @".testlicenses"; } static NSString * licenseFilename() { return @".testlicense"; } 12 12 #else 13 13 static NSString * b64decode(NSString *b64s) { static NSString * b64decode(NSString *b64s) { $NSData\ *data = \hbox{\tt [[NSData\ alloc]\ initWithBase64EncodedString:b64s\ options:} NSDataBase64DecodingIgnoreUnknownCharacters]; \\$ $15 \qquad \text{NSData *data} = \text{[[NSData alloc] initWithBase64EncodedString:b64s options:NSDataBase64DecodingIgnoreUnknownCharacters];} \\$ return [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding]; return [[NSString alloc] initWithData:data encoding:NSUTF8StringEncoding]; 16 17 17 } 18 18 static NSString * licenseFilename() { static NSString * licenseFilename() { return b64decode(@"LmFsdGljZW5zZQ=="); // @".alticense"; return b64decode(@"LmFsdGljZW5zZXM="); // @".alticenses"; 20 21 21 } 22 #endif 22 #endif 23 #pragma mark - Application Support Directory #pragma mark - Application Support Directory static NSString * applicationSupportDirectoryPath() { static NSString * applicationSupportDirectoryPath() { NSArray * paths = NSS earch Path For Directories In Domains (NSApplication Support Directory, NSUser Domain Mask, YES);NSArray * paths = NSSearch Path For Directories In Domains (NSApplication Support Directory, NSUser Domain Mask, YES);if ([paths count]==0) return nil; if ([paths count]==0) return nil; 27 27 28 29 NSString *path = paths[0]; 29 NSString *path = paths[0]; NSString *executableName = 30 NSString *executableName = 30 [[[NSBundle mainBundle] infoDictionary] objectForKey:@"CFBundleExecutable"]; 31 [[[NSBundle mainBundle] infoDictionary] objectForKey:@"CFBundleExecutable"]; 32 return [path stringByAppendingPathComponent:executableName]; return [path stringByAppendingPathComponent:executableName]; 33 34 34 35 35 static BOOL writeLicenseDataToApplicationSupportDirectory(NSData *licenseData) static BOOL writeLicenseDataToApplicationSupportDirectory(NSData *licenseData) 37 37 f (!licenseData) return NO; if (!licenseData) return NO; 38 38 39 39 // 1. Get directory path // 1. Get directory path NSString *dirPath = applicationSupportDirectoryPath(); NSString *dirPath = applicationSupportDirectoryPath(); (!dirPath) return NO; if (!dirPath) return NO; 43 43 // 2. Check directory exist 44 // 2. Check directory exist 44 BOOL isDirectory = NO; BOOL isDirectory = NO; $BOOL\ is Dir Exist = [[NSFile Manager\ default Manager]\ file Exists At Path: dir Path\ is Directory: \& is Directory];$ BOOL isDirExist = [[NSFileManager defaultManager] fileExistsAtPath:dirPath isDirectory:&isDirectory]; if (!isDirectory) return NO; if (isDirExist && !isDirectory) return NO; // 3. Create directory, if not exist // 3. Create directory, if not exist if (!isDirExist) { if (!isDirExist) { BOOL created = [[NSFileManager defaultManager] createDirectoryAtPath:dirPath withIntermediateDirectories:NO attributes:nil error:&error]; BOOL created = [[NSFileManager defaultManager] createDirectoryAtPath:dirPath withIntermediateDirectories:NO attributes:nil error:&error]; (!created) return NO; if (!created) return NO; 54 54 55 55 56 // 3. Check file exist // 3. Check file exist NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; return [licenseData writeToFile:path atomically:YES]; return [licenseData writeToFile:path atomically:YES]; 59 59 60 60 static NSData * getLicenseDataFromApplicationSupportDirectory() static NSData * getLicenseDataFromApplicationSupportDirectory() 61 62 62 // 1. Get Application Support directory path # 1. Get Application Support directory path NSString *dirPath = applicationSupportDirectoryPath(); NSString *dirPath = applicationSupportDirectoryPath(); (!dirPath) return nil; if (!dirPath) return nil; 66 66 // 2. Check directory exist // 2. Check directory exist 67 BOOL isDirectory = NO; BOOL isDirectory = NO; $BOOL\ is Dir Exist = [[NSFile Manager\ default Manager]\ file Exists At Path: dir Path\ is Directory: \& is Directory];$ BOOL isDirExist = [[NSFileManager defaultManager] fileExistsAtPath:dirPath isDirectory;&isDirectory]; f (!isDirExist || !isDirectory) return nil; if (!isDirExist || !isDirectory) return nil; 71 71 // 3. Check file exist // 3. Check file exist NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; BOOL isExist = [[NSFileManager defaultManager] fileExistsAtPath:path isDirectory:&isDirectory]; $BOOL\ is Exist = \hbox{\tt [[NSFileManager\ defaultManager]} fileExists At Path: path\ is Directory: \&is Directory];}$ (!isExist) return nil; if (!isExist) return nil; 75 75 76 76 // 4. Read file 77 // 4. Read file 77 return [NSData dataWithContentsOfFile:path]; return [NSData dataWithContentsOfFile:path]; 79 79 80 80 #pragma mark - Shared Directory 81 #pragma mark - Shared Directory static NSString * sharedDirectoryPath() { static NSString * sharedDirectoryPath() { NSArray* paths = NSSearchPathForDirectoriesInDomains(NSUserDirectory, NSLocalDomainMask, YES); NSArray* paths = NSSearchPathForDirectoriesInDomains(NSUserDirectory, NSLocalDomainMask, YES); ([paths count]==0) return nil; ([paths count]==0) return nil; NSString *path0 = paths[0]; NSString *path0 = paths[0]; 86 86 NSString *path = [path0 stringByAppendingPathComponent:@"Shared"]; NSString *path = [path0 stringByAppendingPathComponent:@"Shared"]; 87 NSString *executableName = NSString *executableName = [[[NSBundle mainBundle] infoDictionary] objectForKey:@"CFBundleExecutable"] $\hbox{\tt [[[NSBundle\ mainBundle]\ infoDictionary]\ objectForKey:@"CFBundleExecutable"];}$ 89 90 90 $return\ [path\ stringByAppendingPathComponent: executableName];$ return [path stringByAppendingPathComponent:executableName]; 91 91 92 92 93 93 static BOOL writeLicenseDataToSharedDirectory(NSData *licenseData) { static BOOL writeLicenseDataToSharedDirectory(NSData *licenseData) { (!licenseData) return NO; if (!licenseData) return NO; 95 // 1. Get directory path // 1. Get directory path 97 NSString *dirPath = sharedDirectoryPath(); NSString *dirPath = sharedDirectoryPath(); (!dirPath) return NO; (!dirPath) return NO; 100 101 // 2. Check directory exist // 2. Check directory exist 101 102 BOOL isDirectory = NO; BOOL isDirectory = NO; $BOOL\ is Dir Exist = \hbox{\tt [[NSFileManager\ defaultManager]} file Exists At Path: dir Path\ is Directory: \& is Directory]; \\$ $BOOL\ is Dir Exist = \hbox{\tt [[NSFileManager\ defaultManager]} file \hbox{\tt ExistsAtPath:} dir Path\ is Directory: \& is Directory]; \\$ if (!isDirectory) return NO; if (isDirExist && !isDirectory) return NO; 105 105 // 3. Create directory, if not exist 106 // 3. Create directory, if not exist 107 if (!isDirExist) { 107 if (!isDirExist) { 108 NSError* error; BOOL created = [[NSFileManager defaultManager] createDirectoryAtPath:dirPath withIntermediateDirectories:NO attributes:nil error:&error]; BOOL created = [[NSFileManager defaultManager] createDirectoryAtPath:dirPath withIntermediateDirectories:NO attributes:nil error:&error]; if (!created) return NO; 110 if (!created) return NO; 110 111 111 112 112 113 // 3. Check file exist 113 // 3. Check file exist NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; 114 NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; return [licenseData writeToFile:path atomically:YES]; return [licenseData writeToFile:path atomically:YES]; 116 116 117 117 static NSData * getLicenseDataFromSharedDirectory() 118 static NSData * getLicenseDataFromSharedDirectory() 118 119 119 // 1. Get shared directory path 120 // 1. Get shared directory path NSString *dirPath = sharedDirectoryPath(); 121 NSString *dirPath = sharedDirectoryPath(); (!dirPath) return nil; if (!dirPath) return nil; 122 122 123 123 // 2. Check directory exist 124 // 2. Check directory exist BOOL isDirectory = NO; BOOL isDirectory = NO; $BOOL\ is Dir Exist = \hbox{\tt [[NSFileManager\ defaultManager]} file Exists At Path: dir Path\ is Directory: \& is Directory]; \\$ $BOOL\ is Dir Exist = \hbox{\tt [[NSFileManager] fileExistsAtPath:} dir Path\ is Directory: \& is Directory]; and the property of th$ if (!isDirExist || !isDirectory) return nil; if (!isDirExist || !isDirectory) return nil; 128 128 129 // 3. Check file exist 129 // 3. Check file exist NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; 130 NSString *path = [dirPath stringByAppendingPathComponent:licenseFilename()]; 131 BOOL isExist = [[NSFileManager defaultManager] fileExistsAtPath:path isDirectory:&isDirectory]; BOOL isExist = [[NSFileManager defaultManager] fileExistsAtPath:path isDirectory:&isDirectory]; 132 if (!isExist) return nil; 132 if (!isExist) return nil; 133 133 134 // 4. Read file 134 // 4. Read file return [NSData dataWithContentsOfFile:path]; return [NSData dataWithContentsOfFile:path]; 136 136 } 137 137 138 138 #pragma mark - write NSDictionary License into the all storages #pragma mark - write NSDictionary License into the all storages static BOOL altSaveLicenceInfoToThe(NSDictionary *license, NSInteger toThe) static BOOL altSaveLicenceInfoToThe(NSDictionary *license, NSInteger toThe) 141 if (!license) return NO; 142 if (!license) return NO; 143 143 // 1. Convert NSDictionaty to NSData 144 // 1. Convert NSDictionaty to NSData 145 NSError *error; NSError *error; NSData *data = [NSPropertyListSerialization dataWithPropertyList:license 146 NSData *data = [NSPropertyListSerialization dataWithPropertyList:license format:NSPropertyListXMLFormat_v1_0 options:0 error:&error]; format:NSPropertyListXMLFormat_v1_0 options:0 error:&error]; if (!data) return NO; 148 if (!data) return NO; 149 149 // 2. Encrypt data 150 // 2. Encrypt data NSData *encData = [ALTLicenseFileData altEncryptFileData:data]; NSData *encData = [ALTLicenseFileData altEncryptFileData:data] 152 if (!encData) return NO; (!encData) return NO; 153 153 // 3. Save to Application Support directory 154 // 3. Save to Application Support directory if (toThe==0 || toThe==1) { 155 if (toThe==0 || toThe==1) { writeLicenseDataToApplicationSupportDirectory(encData); 156 writeLicenseDataToApplicationSupportDirectory(encData); 157 157 } 158 158 // 4. Save to the Shared dir 159 // 4. Save to the Shared dir if (toThe==0 || toThe==2) { if (toThe==0 || toThe==2) { write License Data To Shared Directory (enc Data);write License Data To Shared Directory (enc Data);162 162 163 163 164 return YES; 164 return YES: 165 165 } 166 166 167 167 @implementation ALTRegisterFiles @implementation ALTLicensesFiles 168 168 169 169 + (BOOL)altSaveLicenceInfo:(NSDictionary *)license + (BOOL)altSaveLicencesInfo:(NSDictionary *)licenses 170 170 171 171 return altSaveLicenceInfoToThe(licenses, 0); return altSaveLicenceInfoToThe(license, 0); 173 173 } 174 #pragma mark - return NSDictionary License 175 #pragma mark - return NSDictionary License // Из набора байтов делает NSDictionary с проверкой 176 // Из набора байтов делает NSDictionary с проверкой static NSDictionary *data2lice(NSData *encData) { static NSDictionary *data2lice(NSData *encData) { 178 if (!encData) return nil; if (!encData) return nil; 178 179 179 180 NSData *data = [ALTLicenseFileData altDecryptFileData:encData]; NSData *data = [ALTLicenseFileData altDecryptFileData:encData]; if (!data) return nil; 181 if (!data) return nil; 181 NSPropertyListFormat format; 183 NSPropertyListFormat format; NSError *error; 184 NSError *error; 184 NSDictionary *plist = [NSPropertyListSerialization propertyListWithData:data options:NSPropertyListImmutable format:&format error:&error]; 185 NSDictionary *plist = [NSPropertyListSerialization propertyListWithData:data options:NSPropertyListImmutable format:&format error:&error]; 186

186 // if (licenseValid(plist)) return plist; 187 return plist; 188 } + (NSDictionary *)altGetLicenseInfo + (NSDictionary *)altGetLicensesInfo 191 NSDictionary *license = nil; NSDictionary *license = nil;

201 if (!license) license = li1;

204 // Shared directory

205 BOOL isLi2 = NO;

209 if (li2) {

212

213

217

219

220

210 isLi2 = YES;

206 NSDictionary *li2 = nil;

211 if (!license) license = li2;

214 if (!license) return nil;

218 return license;

207 NSData *encData2 = getLicenseDataFromSharedDirectory();

if (encData2) li2 = data2lice(encData2);

if (!isLi1) altSaveLicenceInfoToThe(license, 1);

if (!isLi2) altSaveLicenceInfoToThe(license, 2);

202 }

203

193 194 // Application Support Directory // Application Support Directory BOOL isLi1 = NO; 195 BOOL isLi1 = NO; 196 NSDictionary *li1 = nil; NSDictionary *li1 = nil; NSData *encData1 = getLicenseDataFromApplicationSupportDirectory(); NSData *encData1 = getLicenseDataFromApplicationSupportDirectory() if (encData1) li1 = data2lice(encData1); if (encData1) li1 = data2lice(encData1); 199 if (li1) { 200 isLi1 = YES;

187 188

189

190

191 192

194

195

200

201

202 203

204

205

210

211

213

214

215 216 217

221

222

223

224

if (!license) license = li1;

// Shared directory

BOOL isLi2 = NO;

isLi2 = YES;

Preferences

return license;

if (!license) return nil;

NSDictionary *li2 = nil;

if (!license) license = li2;

f (encData2) li2 = data2lice(encData2);

f (!isLi1) altSaveLicenceInfoToThe(license, 1);

if (!isLi2) altSaveLicenceInfoToThe(license, 2);

NSData *encData2 = getLicenseDataFromSharedDirectory();

225 226 221 @end 227 222 Работа за границей? Ha italki профессиональные учителя с опытом работы в бизнесе. Выбери свою цену за урок. italki Открыть 🤰

текстов (посимвольно, пословно и построчно).

программой patch (программная утилита Unix). Diff-утилита была разработана в начале 1970-х годов для операционной системы Unix. Финальная версия была полностью разработана Дугласом Макилроем. Алгоритм стал известен как алгоритм Ханта-Макилроя. Модификации с 1975 года включают улучшение основного алгоритма, добавление новых ключей команды и новые форматы вывода. Базовый алгоритм описывается в книгах Юджина В. Майерса "An O(ND) Difference Algorithm and its Variations" и в книге "A File Comparison Program" Вебба Миллера и Майерса. Алгоритм был независимо разработан и описан Е. Укконеном в "Algorithms for Approximate String Matching". Первые версии программы diff были разработаны для сравнения строк

С помощью этого инструмента можно легко выделить различия между двумя текстами. Инструмент очень легко используется. В отдельных блоках введите два текста и получите результат внизу. Инструмент наглядно отобразит различия между двумя текстовыми областями, выделяя измененные части красным цветом. Вы можете сами выбрать метод сравнения

Инструмент используется, чтобы показать различия между двумя версиями одного и того же файла. Современные реализации поддерживают также двоичные файлы. Вывод называется "diff", или патч, так как он может быть применен с

Используйте бесплатный онлайн инструмент Code Diff для сравнения двух текстовых файлов.

привела к изменениям в разработке и реализации программы. Почему использовать инструмент Code Diff?

текстовых файлов, рассчитывая, что символ новой строки разделит строки. В 1980-х годах поддержка двоичных файлов

Сейчас очень распространено явление, когда люди копируют текст из одного сайта и публикуют его как свой собственный контент, что непрофессионально и называется плагиатом (plagiarism). Этот инструмент поможет вам избегать плагиата. Необходимо скопировать два текста, и инструмент покажет, в каких частях есть плагиат. Учитывайте также, что контент с плагиатом приносит меньше трафика. Если ваш контент можно найти на других сайтах, это приносит меньше трафика, так как вы не обеспечиваете хороший контент для посетителей. Плагиат может быть небезопасен, так как механизмы поиска используют поисковый бот для индексации контента разных

версию контента хранить, и какую удалить. Страницы с лучшим рейтингом и качеством хранятся, а другие игнорируются. Этот инструмент поможет создать лучший контент и избегать плагиата.

Diff сравнивает коды, выделяет различия и удаляет ошибки.

Code Diff имеет много преимуществ: 1. Одним из преимуществ является то, что нет необходимости в чтении контента несколько раз. Например, программист может найти один и тот же тип кода, написанный немного по-другому. Инструмент облегчит работу программистов при сравнения кодов с их стандартными версиями, выделяя только их различия. Это самый лучший и быстрый способ выполнения подобной работы, что позволяет находить ошибки в коде путем его сравнения с оригинальным кодом. Code

сайтов в базах данных. Если два или несколько сайта содержат одинаковый контент, поисковые боты не поймут, какую

получите результат. Найдите похожие части двух текстов за несколько секунд. 3. Code Diff - это отличный способ для преподавателей, чтобы сравнить работы студентов. В школах и университетах ученики/студенты копируют работы друг друга или же копируют только одну часть работы, что затрудняет ситуацию.

4. Плагиат контента небезопасен для блоггеров. Если вы делаете запись в блоге, она должна быть максимально уникальной,

Do you find this helpful?

2. Следующее преимущество заключается в том, что Code Diff экономит время. Вам не нужно тратить время на чтение двух

текстов, чтобы найти различия между ними. Всего лишь необходимо скопировать и вставить тексты, и вы сразу же

чтобы привлечь пользователей. Этот инструмент поможет сравнить ваш текст с сомнительным текстом, имеющим похожий контент, и при наличии плагиата можете сообщить, чтобы его удалили.

Именно поэтому этот инструмент очень полезен в вопросе нахождения подобного плагиата.

No

Вычисление длины строки



tktimport.com Sting Functions **URL** Encoder **URL** Decoder String Word Count Base 64 Encoder Base 64 Decoder HTML Tags Remover

Empty Lines Remover

String to MD5 Hash

Generator

String Length Calculator

String to Sha256 Hash Generator String Reverse Extra Spaces Remover String to Lowercase Convertor String to Uppercase Convertor Binary Data to Hex Converter Hex Encoded Binary String Decoder Rot13 to Text Converter String to Binary Converter **Duplicate Lines** Remover

tktimport.com

Company © W3docs. All rights reserved.

Учебник HTML

Kaĸ JavaScript

Основы HTML

Инструменты

Оптимизатор изображений