http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:hyb3.5:guide:ios:npki_%EC%BC%80%EC%9D%B4%EC%82%AC%EC%9D%B8

NPKI(WizSign) Device API Guide Program

Outline

NPKI(Wizsign) is a guide application for eGov Device API, using the mobile device API framework to be used as a tool and a reference when developing hybrid applications. It supports the NPKI related functions of mobile smart devices through JavaScript-based NPKI DeviceAPI.

Also, it connects with web server applications based on eGov standard framework in order to authenticate certificates, save the result to server, and reference authentication results log.

Feature

This Guide Program provides **Select/authenticate Mobile Device Certification** and **View authentication log information** features. These features are realized in a way that applies **Standard Security API** to web server applications that allows for checking certificate information.

Assumptions

Category	Description
Local Device Environments	Xcode 6.3.2, PhoneGap 4.3.0
Server-side Developmental Environment	eGov Standard Framework Develeopment Environment 3.5
Works in sync with Mash up Open API	N/A
Test Device	iPhone4, iPhone6
Test Platform	iOS 7.1.2, iOS 8.3
Libraries Added	WizSign library applied wizsignpg.js, cert.db, WizSignPG.h, WizSignPGlib.a

Restriction

NPKI library

- Since NPKI Device API Guide Program does not include security module, one must request license contract and support to security module firm below.

Name	Point of Contact	Contact Phone	Homepage
KSign Inc.	Shin, Dong-Soo	02-564-0182	http://www.ksign.com

Applying eGov security standard API

A separate request for security standard API must be made in order to use eGov security standard API, which can be made at Administrative Electronic Signature Management Center (http://www.gpki.go.kr).

Follow the instructions below.

▶ When Standard API management system can be accessed
⬭ Request the API via web at [Standard API management system] (attach memorandum and diagram)
⬭ Service URL : http://api.gpki.go.kr
Send memo to Korea Local Information Research & Development Institute - Local Information Center - Information Infrastructure Branch.
The content of the memorandum should include the name of the system, Point of Contact, and the request for standard API.
- The following service can only be accessed in the government network -

▶ When commercial internet (http://api.gpki.go.kr Connection Unavailable) cannot be used
⬭ At the Government Electronic Signature Certification Management Center (http://www.gpki.go.kr) website, fill in the request form (“Downloads-Certification Request Forms-7.Standard API request instructions and Standard API request form”) along with the memorandum.
Memorandum To : Korea Local Information Research & Development Institute - Local Information Center - Information Infrastructure Branch
The content of the memorandum should include the name of the system, Point of Contact, and the request for standard API.

Refer to Government Electronic Signature Certification Management Center(http://www.gpki.go.kr) for additional information and inquiries.

Supported devices and platforms

For iPhone devices, there may be issues due to device's processing power.

- Problem: PhoneGap error.
- Solution: delay PhoneGap loading sequence with setTimeout() function.

```
    document.addEventListener('DOMContentLoaded', function () { setTimeout(loaded, 200); }, false);
```

- Problem: iScroll5 content height calculation error.
- Solution: use setTimeout() to ensure iscroll is generated after css application to contents is complete.

```
setTimeout(function()
        {
            myScroll = new iScroll(thisPage,
                                    {
                                        checkDOMChanges: true,
                                        onBeforeScrollStart:function(e)
                                        {
                                        }
                                    });
        },
```

500);
Problems may occur if alert() is included in CallBack function. (phoneGap)

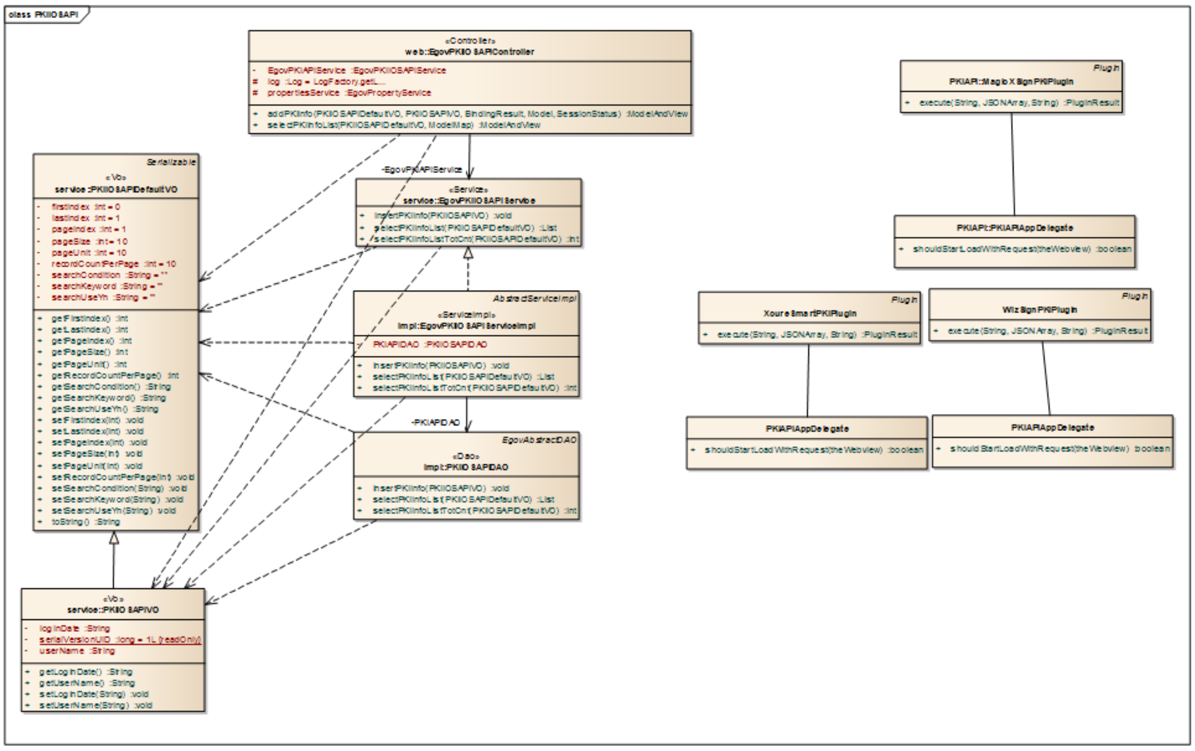
- Problem: error calling alert() message from CallBack function saved in PhoneGap.
- Solution: Use asynchronous function or avoid using functions that use thread like alert().

Cross domain usage ===When using certain outside domains or its subdomains on PhoneGap, add such domains on <key>ExternalHosts</key> at Resource/Cordova.plist.

Description

NPKI Device API Guide Program is comprised of: a) a function that selects the certificate on the mobile device and then creates the signature data, sends it, and authenticates the certificate and b) inquires the authentication log data. (refer to the Related Features section)

Class Diagram


class PKIIOSAPI
«Controller»
web::EgovPKIIOSAPIController
- EgovPKIAPIService :EgovPKIIOSAPIService
# log :Log = LogFactory.getL...
# propertiesService :EgovPropertyService
+ addPKIInfo(PKIIOSAPIDefaultVO, PKIIOSAPIVO, BindingResult, Model, SessionStatus) :ModelAndView
+ selectPKIInfoList(PKIIOSAPIDefaultVO, ModelMap) :ModelAndView
-EgovPKIAPIService
«Service»
service::EgovPKIIOSAPIService
+ insertPKIInfo(PKIIOSAPIVO) :void
+ selectPKIInfoList(PKIIOSAPIDefaultVO) :List
+ selectPKIInfoListTotCnt(PKIIOSAPIDefaultVO) :int
Serializable
«Vo»
service::PKIIOSAPIDefaultVO
- firstIndex :int = 0
- lastIndex :int = 1
- pageIndex :int = 1
- pageSize :int = 10
- pageUnit :int = 10
- recordCountPerPage :int = 10
- searchCondition :String = ""
- searchKeyword :String = ""
- searchUseYn :String = ""
+ getFirstIndex() :int
+ getLastIndex() :int
+ getPageIndex() :int
+ getPageSize() :int
+ getPageUnit() :int
+ getRecordCountPerPage() :int
+ getSearchCondition() :String
+ getSearchKeyword() :String
+ getSearchUseYn() :String
+ setFirstIndex(int) :void
+ setLastIndex(int) :void
+ setPageIndex(int) :void
+ setPageSize(int) :void
+ setPageUnit(int) :void
+ setRecordCountPerPage(int) :void
+ setSearchCondition(String) :void
+ setSearchKeyword(String) :void
+ setSearchUseYn(String) :void
+ toString() :String
AbstractServiceImpl
«ServiceImpl»
impl::EgovPKIIOSAPIServiceImpl
- PKIAPIDAO :PKIIOSAPIDAO
+ insertPKIInfo(PKIIOSAPIVO) :void
+ selectPKIInfoList(PKIIOSAPIDefaultVO) :List
+ selectPKIInfoListTotCnt(PKIIOSAPIDefaultVO) :int
-PKIAPIDAO
EgovAbstractDAO
«Dao»
impl::PKIIOSAPIDAO
+ insertPKIInfo(PKIIOSAPIVO) :void
+ selectPKIInfoList(PKIIOSAPIDefaultVO) :List
+ selectPKIInfoListTotCnt(PKIIOSAPIDefaultVO) :int
«Vo»
service::PKIIOSAPIVO
- loginDate :String
- serialVersionUID :long = 1L {readOnly}
- userName :String
+ getLoginDate() :String
+ getUserName() :String
+ setLoginDate(String) :void
+ setUserName(String) :void
Plugin
PKIAPI::MagicXSignPKIPlugin
+ execute(String, JSONArray, String) :PluginResult
PKIAPI::PKIAPIAppDelegate
+ shouldStartLoadWithRequest(theWebview) :boolean
Plugin
XoureSmartPKIPlugin
+ execute(String, JSONArray, String) :PluginResult
Plugin
WizSignPKIPlugin
+ execute(String, JSONArray, String) :PluginResult
PKIAPIAppDelegate
+ shouldStartLoadWithRequest(theWebview) :boolean
PKIAPIAppDelegate
+ shouldStartLoadWithRequest(theWebview) :boolean

Device Application

Source

Type	Title	Remark
CSS	www/css/egovframwork/mbl/hyb/PKIWizSignAPI.css	NPKIAPI Guide Program Core Cascading Style Sheets
IMAGE	www/images/egovframwork/mbl/hyb/	NPKIAPI Guide Program main Image Folder

JS	www/js/egovframwork/mbl/hyb/PKIWizSignAPI.js	NPKIAPI Guide Program main JavaScript
JS	www/js/egovframwork/mbl/hyb/wizsignpg.js	NPKIAPI Guide Program main JavaScript
JS	www/js/egovframwork/mbl/hyb/messages_ko.js	JavaScript for Validate Message Processing
HTML	www/NPKIWizSignAPI.html	NPKIAPI main page
HTML	www/license.html	NPKIAPI license page
HTML	www/overview.html	NPKIAPI feature description page

Function API

[WizSign API DOC]

APIs Used

doSignature

- Conducts electronic signature using selected Certificate and returns signature value

- Parameters: Certificate #, Certificate password, subject original
- Return value (hash table)

```
'signedData' : signed data
'errMsg' : error message
var args = new Array();
args[0] = selectCertNum.toString() ;
args[1] = '1';
args[2] = stringToSign;

WizSignPG.doSignature(args, function(result) {
            var signedData = result['signedData'];      // signed data
     }, function(error) {
            alert(error['errMsg']);              // error message
});
```

getCertificates

- Calls and returns the saved Certificate list

- Parameters : N/A
- Return value (hash table)

```
'Certificates' : Certificate list
'errMsg' : error message
WizSignPG.getCertificates("", function(result) {
     var certList = result['Certificates'];

     for(var i=0 ; i<certList.length ; i++) {
          certList[i]['NUM'];
          certList[i]['HOST'];
          certList[i]['ISSUED BY'];
```

```
            certList[i]['EXPIRATION DATE'];
        }

        }, function(error) {
            alert("error['errMsg']);
});
```

Certificate information hash table

Hash table	Description
NUM	Certificate No.
Version	Certificate version
Serial No.	Certificate Serial No.
Signature algorithm	Certificate Signature algorithm
Issuer	Certificate issuer information
Date of effect	Certificate's date of effect
Expiration Date	Certificate's Expiration Date
Subject	Certificate subject data
Public key algorithm	Certificate Public key algorithm
Issuer Serial No.	Issuer Serial No.
Public Key	Public Key value
Institution key identifier	Institution key identifier
Subject identifier	Subject identifier
Policy	Policy
Subject alternative name	Subject alternative name
CRL division point	CRL division point
Institution information access	Institution information access (OCSP)
Key use	Purpose of key use
Signature	Certificate signed value

verifyCertPassword

- Verifies selected Certificate's password.

- Parameters: Certificate No., Certificate password
- Return value (hash table)

'result' : Certificate password verification result('OK' when successful)
'errMsg' : error message

```
var args = new Array();
args[0] = certNum.toString();
args[1] = certPass;

WizSignPG.verifyCertPassword(args, function(result) {
        var runResult = result['result'];

        if(runResult == 'OK') {
                alert('Correct Password.');
        }

        }, function(error) {
        alert(error['errMsg']);
});
```

changeCertPassword

- Changes selected Certificate's password.

- Parameters: Certificate #, Certificate password, new Certificate password
- Return value (hash table)

'result' : Certificate password change result('OK' when successful)
'errMsg' : error message

```
var args = new Array();
args[0] = certNum.toString();
args[1] = beforePass;
args[2] = afterPass;

WizSignPG.changeCertPassword(args, function(result) {
        var runResult = result['result'];

        if(runResult == 'OK') {
                alert('Certification Password Changed.');
        }

        }, function(error) {
        alert(error['errMsg']);
});
```

removeCert

- Delete selected Certificate.

- Parameters: Certificate No.
- Return value (hash table)

'result': Certificate delete result ('OK' when successful)
'Certificate' : information of deleted Certificate
'errMsg' : error message

```
var args = new Array();
args[0] = certNum.toString();

WizSignPG.removeCert(args, function(result) {
    var runResult = result['result'];

    if(runResult == 'OK') {
        alert('Certification Deleted.')
    }

    }, function(error) {
        alert(error['errMsg']);
});
```

doValidateCert

- Conducts validation of selected Certificate. (CRL verification)

- Parameters: Certificate No.
- Return value (hash table)

```
'result': Certificate CRL verification result('OK' when successful)
'status': Certificate status
'errMsg' : error message
var args = new Array();
args[0] = certNum.toString();

WizSignPG.doValidateCert(args, function(result) {
    var runResult = result['result'];
    var certStatus = result['status'];

    if(runResult == 'OK') {
        alert(certStatus);
    }

    }, function(error) {
        alert(error['errMsg']);
});
```

Server Application

Source

Type	Title	Remark
Controller	egovframework.hyb.ios.pki.web.EgovPKIiOSAPIController.java	NPKIAPI Guide Program Controller Class
Service	egovframework.hyb.ios.pki.service.EgovPKIiOSAPIService.java	NPKIAPI Guide Program Service Class
ServiceImpl	egovframework.hyb.ios.pki.service.impl.EgovPKIiOSAPIServiceImpl.java	NPKIAPI Guide Program ServiceImpl Class
VO	egovframework.hyb.ios.pki.service.PKIiOSAPIDefaultVO.java	NPKIAPI Guide Program VO Class

VO	egovframework.hyb.ios.pki.service.PKIiOSAPIVO.java	NPKIAPI Guide Program VO Class
VO	egovframework.hyb.ios.pki.service.PKIiOSAPIXmlVO.java	NPKIAPI Guide Program XML related VO Class
DAO	egovframework.hyb.ios.pki.service.impl.PKIiOSAPIDAO.java	NPKIAPI Guide Program Dao Class
QUERY XML	resources/egovframework/sqlmap/hyb/ios/pki/EgovPKIiOSAPIGuide_SQL_XXX.xml	NPKIAPI Guide Program QUERY XML
Idgen XML	resources/egovframework/spring/context-idgen.xml	NPKIAPI Guide Program ID generation Idgen XML

Related Tables

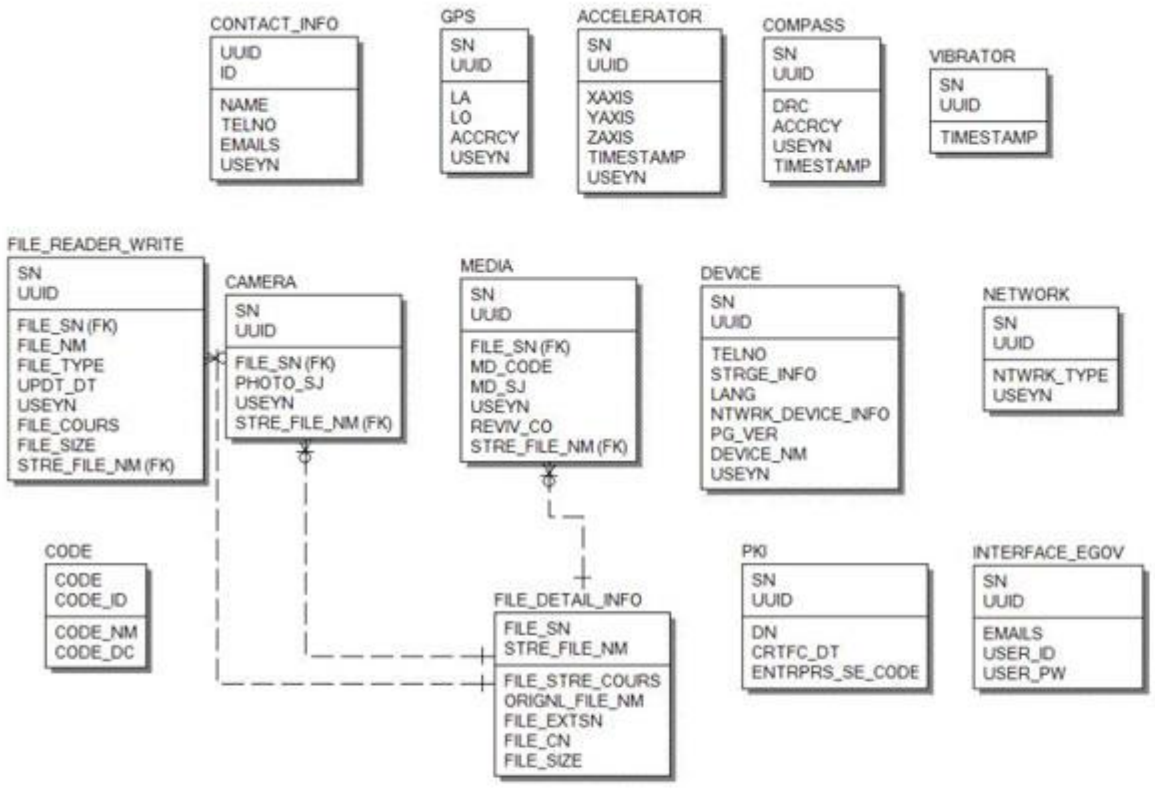
Title	Table	Remark
PKI	PKI	Certification Recognition Log Management

Table Definition

- PKI

No	Column ID	Title of Column	Type	Length	Null
1	SN	Serial No.	NUMERIC	6	NotNull
2	UUID	UUID	VARCHAR	50	NotNull
3	DN	Authentication data	VARCHAR	255	Null
4	CFTFC_DT	Authentication date and time	DATETIME		Null
5	ENTRPRS_SE_CODE	Enterprise code	CHAR	15	Null

ERD



Standard API for Security

```
public String verifyCert(PKIAndroidAPIVO pkiVo) throws Exception {
        // API initialization
        GpkiApi.init("C:/libgpkiapi_jni/conf");
        String sign;
        sign = pkiVo.getSign();
        return verify(Base64.decode(sign));
}

private String verify(final byte[] bSignedData)    {
        String sClientName = "";
        try {
                // authenticates signature
                SignedData signedData = null;
                signedData = new SignedData();
                signedData.verify(bSignedData);

                // acquires server's signing Certificate in order to authenticate subject's Certificate
                X509Certificate clientCert = null;
                clientCert = signedData.getSignerCert(0);

                // Certificate authentication
                CertPathValidator certPathValiditor = null;
```

```
            certPathValiditor = new CertPathValidator("C:/libgpkiapi_jni/conf/gpkiapi.conf");

            // adds top truted Certificate
            X509Certificate rootCertRsa = null;
            rootCertRsa = Disk.readCert("C:/libgpkiapi_jni/conf/root-rsa2.der");
            X509Certificate rootCertRsaSha = null;
            rootCertRsaSha = Disk.readCert("C:/libgpkiapi_jni/conf/root-rsa-sha2.der");
            certPathValiditor.addTrustedRootCert(rootCertRsa);
            certPathValiditor.addTrustedRootCert(rootCertRsaSha);

            // sets client's Certificate authentication level
            certPathValiditor.setVerifyRange(CertPathValidator.CERT_VERIFY_FULL_PATH);

            // sets verification on whether or not the client's Certificate will be purged (sets CRL/ARL
verification)
            certPathValiditor.setRevokationCheck(CertPathValidator.REVOKE_CHECK_ARL |
CertPathValidator.REVOKE_CHECK_CRL);

            // requests Certificate authentication
            certPathValiditor.validate(CertPathValidator.CERT_SIGN, clientCert);

            sClientName = clientCert.getSubjectDN();


        } catch (Exception e) {
            sClientName = "";
        }
        return sClientName;
}
```

Configuration Settings

Necessary sections and settings for using NPKI related features of mobile device, provided by NPKI Device API Guide Program, are as follows.

Device Application

config.xml

Plugin

```
<featurename="InterfaceAPI">
<paramname="ios-package"value="EgovInterface"/>
</feature>
<featurename="WizSignPG">
<paramname="ios-package"value="WizSignPG"/>
</feature>
```

[Project_Name]/eGovModule/EGovComModule.h

```
<!-- Server Directory for eGov Interface Device API Class -->
#define kSERVER_URL         @"Server_URL"
```

Server Application

resource/egovframework/sqlmap/sql-map-config_[DB NAME].xml

```
<sqlMapresource="egovframework/sqlmap/hyb/ios/pki/EgovPKIiOSAPIGuide_SQL_[DB NAME].xml"/>
```

Standard API for Security

Setting reference

Related features

NPKI Device API guide is comprised of **Select/authenticate mobile device Certificate** , **View authentication log** functions.

Select/authenticate mobile device Certificate

Business Logic

Inquires the list of certificates saved on the mobile device through Device API. Authenticates selected Certificate from the list.

Related Code

Inquires the list of Certificates through JavaScript code that uses the inquiry function within the Device API. Signs using the JavaScript that creates signature data.

```
// inquire the list of Certificates
function fn_egov_go_certlist()
{
	console.log("PKIWizSignAPIGuide fn_egov_go_certlist");
	$.mobile.showPageLoadingMsg('a');
	WizSignPG.getCertificates(fn_egov_getcertlistSuccess, fn_egov_getcertlistFail);
}

// verifies Certificate password
function fn_egov_confirm_password() {
	console.log('PKIWizSignAPIGuide fn_egov_confirm_password()');

	var args = new Array();
	var tmpIndex = document.getElementById("xsigncertindex").value;
	args[0] = '1';//tmpIndex.toString();
	args[1] = $("#loginPasswd").val();
	console.log(args);

	WizSignPG.verifyCertPassword(args, function(result) {
							//Certificates
							var runResult = result['result'];
							var error = result['errMsg'];

							if(error!=null){
								alert(error);
							}
```

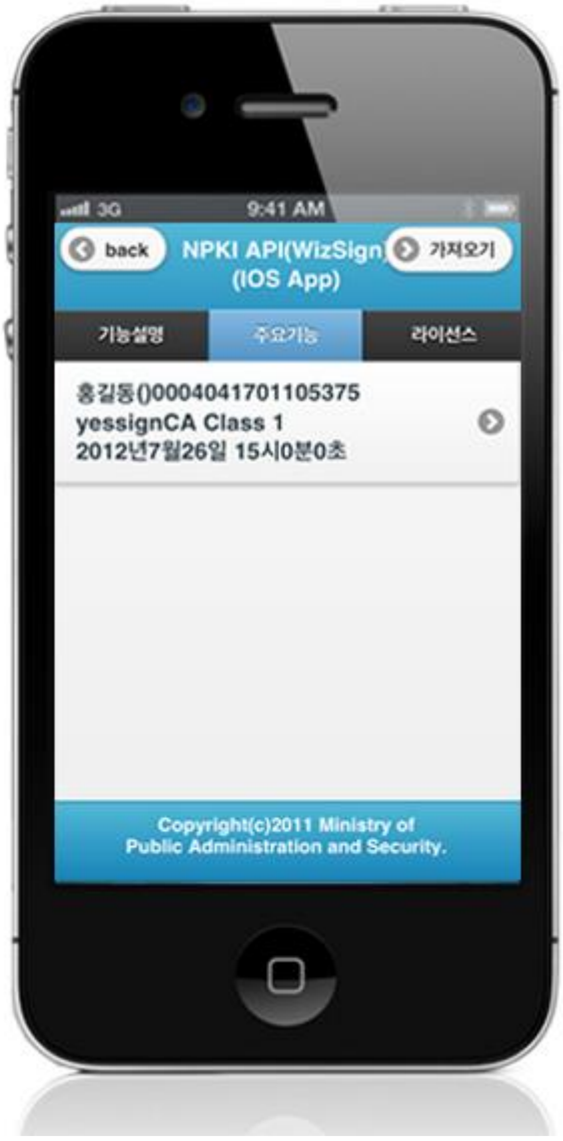
```
                                    if(runResult == 'OK') {
                                        console.log("PKIWizSignAPIGuide
fn_egov_confirm_password Completed");
//                                        alert('Correct Password.');
                                        fn_egov_make_sign();
                                    } else {
                                        console.log("PKIWizSignAPIGuide
fn_egov_confirm_password Failed");
                                    }
                                }, function(error) {
                                    alert("Error: \r\n" + error['errMsg']);
                                });
}

// signs the Certificate
function fn_egov_make_sign()
{
    console.log('PKIWizSignAPIGuide fn_egov_make_sign()');
    var args = new Array();
    args[0] = '1';//document.getElementById("xsigncertindex").value;
    args[1] = $("#loginPasswd").val();
    args[2] = "usrId=&password=&name=";

    WizSignPG.doSignature(args, fn_egov_makesign_ok, fn_egov_makesign_fail);
}

// requests authentication to Certificate signature data server
function fn_egov_makesign_ok(arg)
{
    var signedData = arg['signedData'];
    var params = {uuid :    device.uuid,
            sign: signedData,
            entrprsSeCode: 'PKI02'};

    alert('Http Method:POST\nAcceptType:JSON\nSendData:' + JSON.stringify(params));
    $.mobile.showPageLoadingMsg('a');
    EgovInterface.submitAsynchronous(
                                    [params, "/pki/addPKIiOSInfo.do"],
                                    function(result) {
                                        console.log("PKIWizSignAPIGuide
fn_egov_makesign_ok request Completed");
                                        var str = '{';
                                        for (myKey in result){
                                            str += myKey + ':' + result[myKey] + '\n';
                                        }
                                        str += '}';
                                        alert('Response
Method:RESTful\nResponseType:json, post\nParam:\n' + str);
                                        //window.history.back();
                                        $.mobile.hidePageLoadingMsg('a');
                                        location.href = "index.html";
                                    },
                                    function(error) {
```

```
                                                                console.log("PKIWizSignAPIGuide
fn_egov_makesign_ok request Failed");
                                                                var str = '{';
                                                                for (myKey in error){
                                                                        str += myKey + ' : ' + error[myKey] + '\n';
                                                                }
                                                                str += '}';
                                                                alert('Response Method:RESTful\nSendType:json,
post\nParam:\n' + str);
                                                                $.mobile.hidePageLoadingMsg('a');
                                                        }
                                                        );
}

// calls Olleh Certificate
function doKISAShowApp() {
        var args = new Array();
        args[0] = 'PhoneGapTest';
        args[1] = '01';

        WizSignPG.runShowApp(args, function(result) {
                                                // result
                                                var runResult = result['result'];
                                                // runResult == 'OK' -> successful
                                                console.log("PKIWizSignAPIGuide doKISAShowApp
Completed");
                                        }, function(error) {
                                                console.log("PKIWizSignAPIGuide doKISAShowApp Failed");
                                                navigator.notification.alert("Error: \r\n" + error['errMsg']);
                                        });
}

// saves PKCS#12 data, converting it to Certificate
function makeCert(strP12, certPass, newPass) {

        var args = new Array();
        args[0] = strP12;
        args[1] = certPass;
        args[2] = newPass;

        WizSignPG.importPKCS12(args, function(result) {

                                        var runResult = result['result'];
                                        var certInfo = result['Certificate'];

                                        if(runResult == 'OK') {
                                                console.log("PKIWizSignAPIGuide makeCert Completed");
                                                navigator.notification.alert('[' + certInfo['HOST'] + ']
Certification Created.1')
                                                fn_egov_go_certlist();
                                        }
                                        }, function(error) {
                                                console.log("PKIWizSignAPIGuide makeCert Failed");
                                                navigator.notification.alert("Error: \r\n" + error['errMsg']);
                                        });
}

// processes URL data from KISA ollehApp and returns PKCS#12 data.
function handleOpenURL(url)
{
      console.log('handleOpenURL');
      var g_p12cert = callback_kisaShowApp(url);
      makeCert(g_p12cert, 'han9476046946', 'han9476046946');
}
```

Related Screen and Implementation Manual

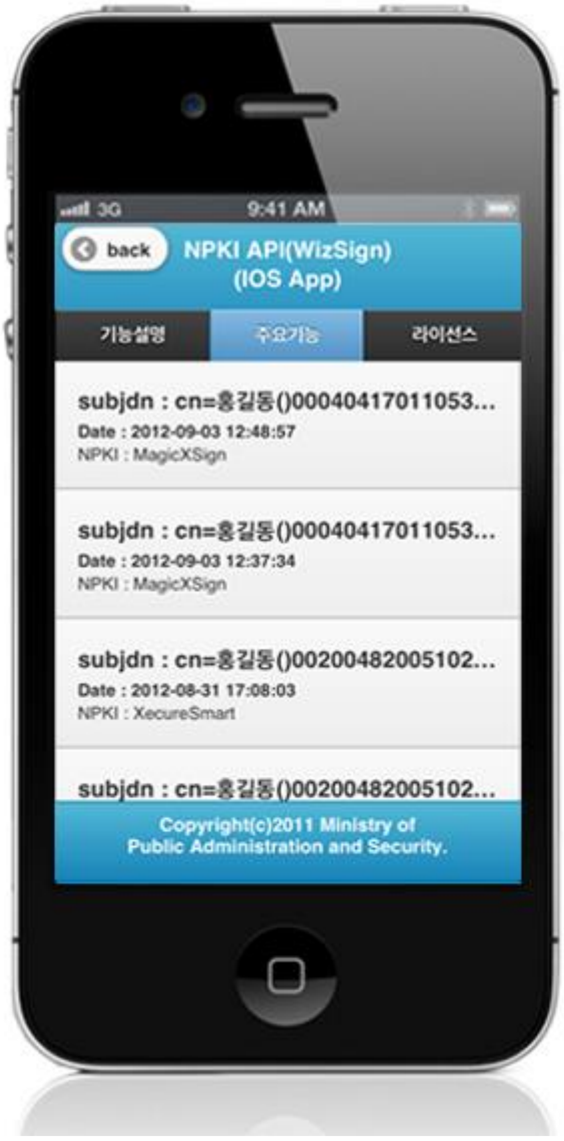
Action	URL	Controller method	QueryID
Certificate authentication	/pki/xml/addPKIiOSInfo.do	addPKIInfoXml	“PKIiOSAPIDAO.insertPKIInfo”

Certificate list	Certificate authentication



홍길동()0004041701105375
yessignCA Class 1
2012년7월26일 15시0분0초

Copyright(c)2011 Ministry of
Public Administration and Security.



subjdn : cn=홍길동()00040417011053...
Date : 2012-09-03 12:48:57
NPKI : MagicXSign

subjdn : cn=홍길동()00040417011053...
Date : 2012-09-03 12:37:34
NPKI : MagicXSign

subjdn : cn=홍길동()00200482005102...
Date : 2012-08-31 17:08:03
NPKI : XecureSmart

subjdn : cn=홍길동()00200482005102...

Copyright(c)2011 Ministry of
Public Administration and Security.

Select the Certificate to be authenticated from the Certificate list window. Enter the password on the password section of the authentication window, and click the "confirm" button.
An error message will be displayed if conditions are insufficient upon checking validation on the password section.

Confirm authentication: enter the Certificate password on the password section adn click "confirm" button.
Back button : moves to **NPKI Device API Guide Program menu** window or**Certificate list** window.

View authentication log

Business Logic

Updates the Certificate Authorization Log out of the web server application.

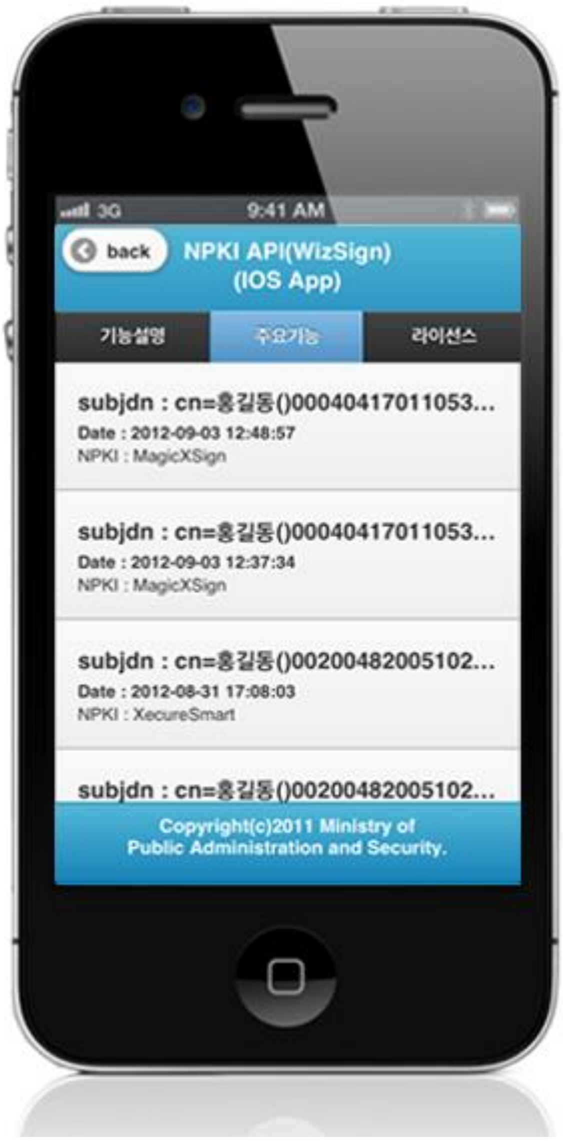
Related Code

```
function fn_egov_go_loginInfoList()
{
        $.mobile.changePage("#loginInfoList", "slide", false, false);

        // get the data from server
        console.log('fn_egov_go_loginInfoList()');
        var accept_type = "json";
        $.mobile.showPageLoadingMsg('a');
        // get the data from server
        EgovInterface.submitAsynchronous(
                                                                        ["/pki/pkiInfoList.do"],
                                                                        function(result) {
                                                                                console.log("PKIWizSignAPIGuide
fn_egov_go_loginInfoList Completed");
                                                                                var list_html = "";
                                                                                var totcnt = result.pkiInfoList.length;

                                                                                for (var i = 0; i < totcnt; i++) {
                                                                                        var data = result.pkiInfoList[i];
                                                                                        var entrprsSe = "NONE";
                                                                                        var entrprsSeCode = data.entrprsSeCode;
                                                                                        if(entrprsSeCode == 'PKI01')
                                                                                                entrprsSe = "MagicXSign";
                                                                                        else if(entrprsSeCode == 'PKI02')
                                                                                                entrprsSe = "WizSign";
                                                                                        else if(entrprsSeCode == 'PKI03')
                                                                                                entrprsSe = "XecureSmart";

                                                                                        list_html += "<li><h3>subjdn : " + data.dn +
"</h3>";

                                                                                        list_html += "<p><strong>Date : " +
data.crtfcDt + "</strong></p>";

                                                                                        list_html += "<p>NPKI : " + entrprsSe +
"</p></li>";

                                                                                }
                                                                                var theList = $('#theLogList');
                                                                                theList.html(list_html);
                                                                                $.mobile.changePage("#loginInfoList", "slide",
false, false);

                                                                                theList.listview("refresh");
                                                                                $.mobile.hidePageLoadingMsg('a');
                                                                                setTimeout(loadiScrollList, 1000);
                                                                        },
                                                                        function(error) {
                                                                                console.log("PKIWizSignAPIGuide
fn_egov_go_loginInfoList Failed");
                                                                                var str = '{';
                                                                                for (var myKey in error){
                                                                                        str += myKey + ' : ' + error[myKey] + '\n';
                                                                                }
                                                                                str += '}';
```

```
                                    alert('Response Method:RESTful\nSendType:json,
post\nParam:\n' + str);
                                    $.mobile.hidePageLoadingMsg('a');
                              }
                              );
}
```

Related Screen and Implementation Manual

Function	URL	Controller	method	QueryID
Inquire Certificate authentication results log	/pki/xml/pkiInfoList.do	EgovPKIiOSAPIController	selectPKIInfoListXml	PKIiOSAPIDAO.selectPKIInfoList



subjdn : cn=홍길동()00040417011053...
Date : 2012-09-03 12:48:57
NPKI : MagicXSign

subjdn : cn=홍길동()00040417011053...
Date : 2012-09-03 12:37:34
NPKI : MagicXSign

subjdn : cn=홍길동()00200482005102...
Date : 2012-08-31 17:08:03
NPKI : XecureSmart

subjdn : cn=홍길동()00200482005102...

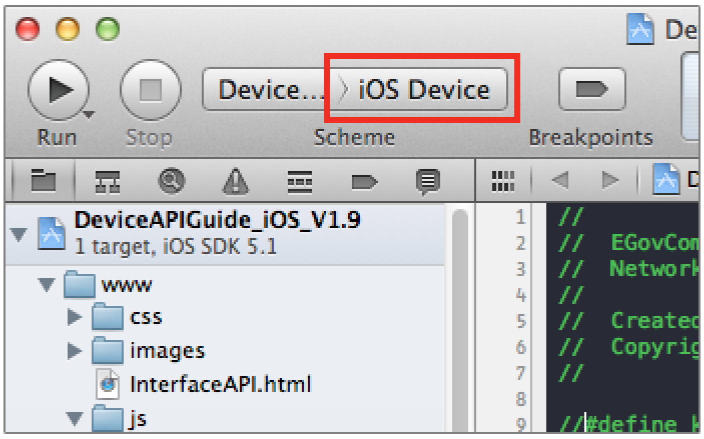
Copyright(c)2011 Ministry of Public Administration and Security.

Compiling, debugging, distributing

Compiling

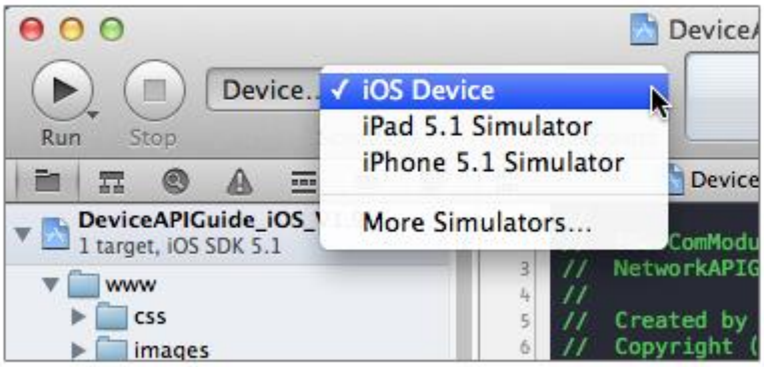
How to compile NPKI Device Applicaton

1. To execute on the device or simulator, click on red border area.



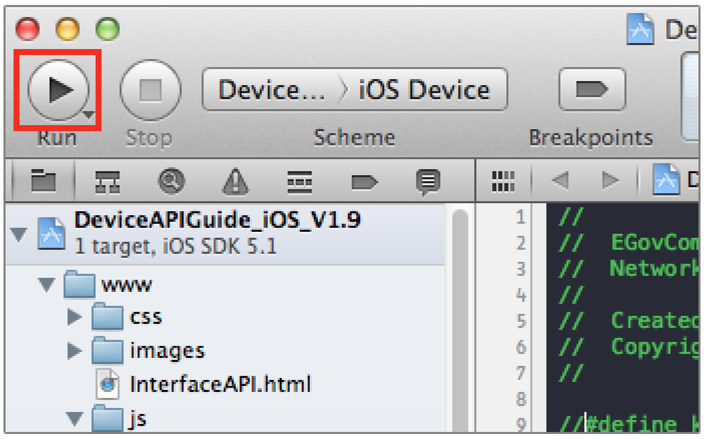
```
//
//   EGovCom
//   Network
//
//   Created
//   Copyrig
//

//#define k
```

2. Select device or simulator.



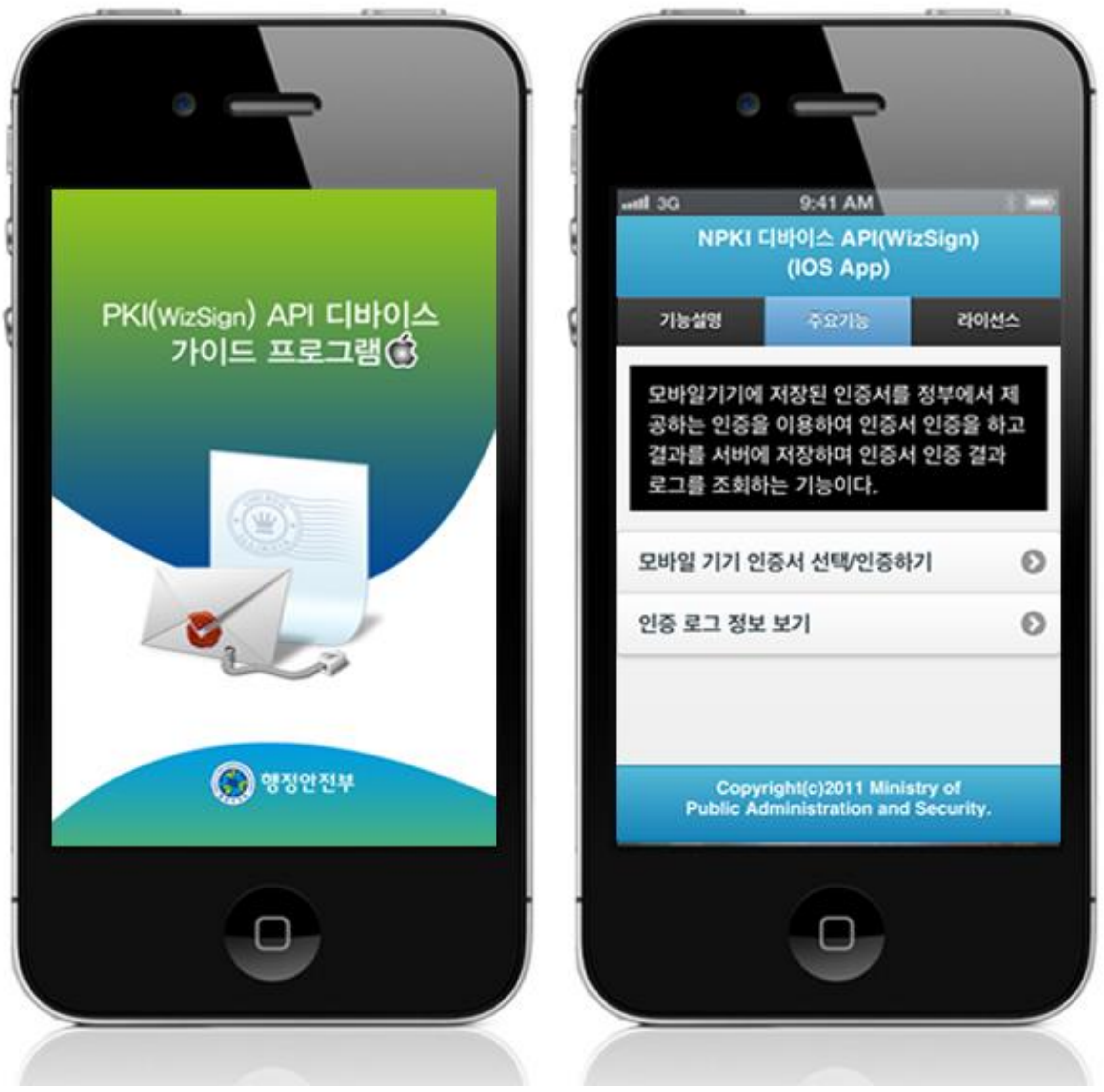
```
//  ComModu
//  NetworkAPIG
//
//  Created by
//  Copyright (
```

3. Click on "Execute."



```
//
//  EGovCom
//  Network
//
//  Createc
//  Copyrig
//

//#define k
```

4. Check intro and main screen.

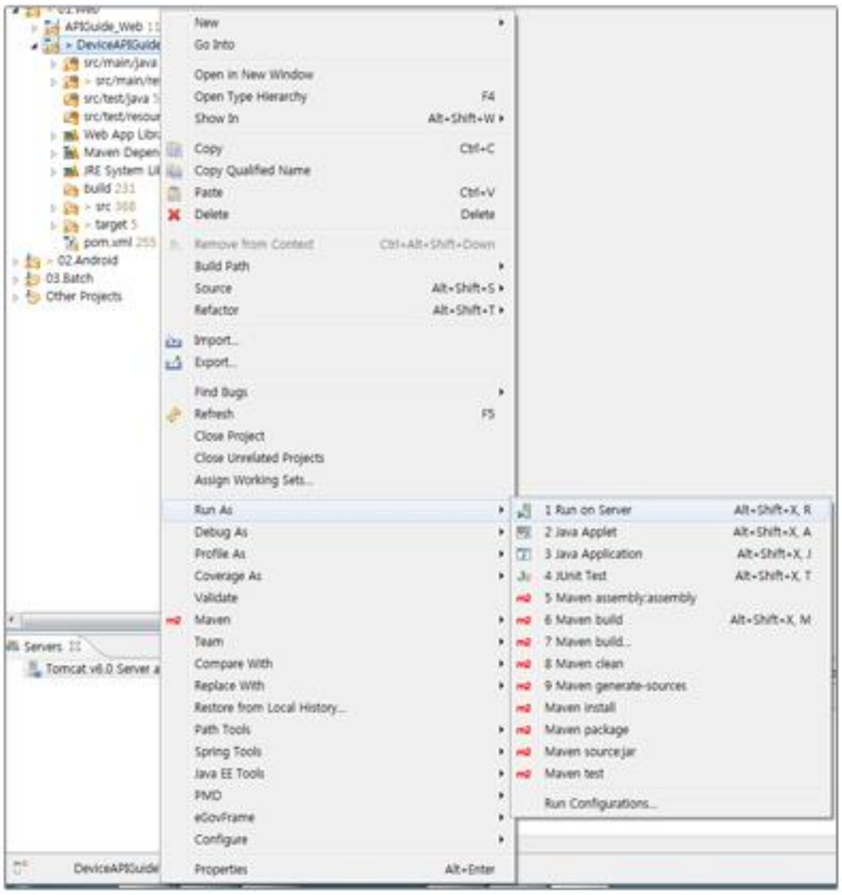


모바일기기에 저장된 인증서를 정부에서 제공하는 인증을 이용하여 인증서 인증을 하고 결과를 서버에 저장하며 인증서 인증 결과 로그를 조회하는 기능이다.

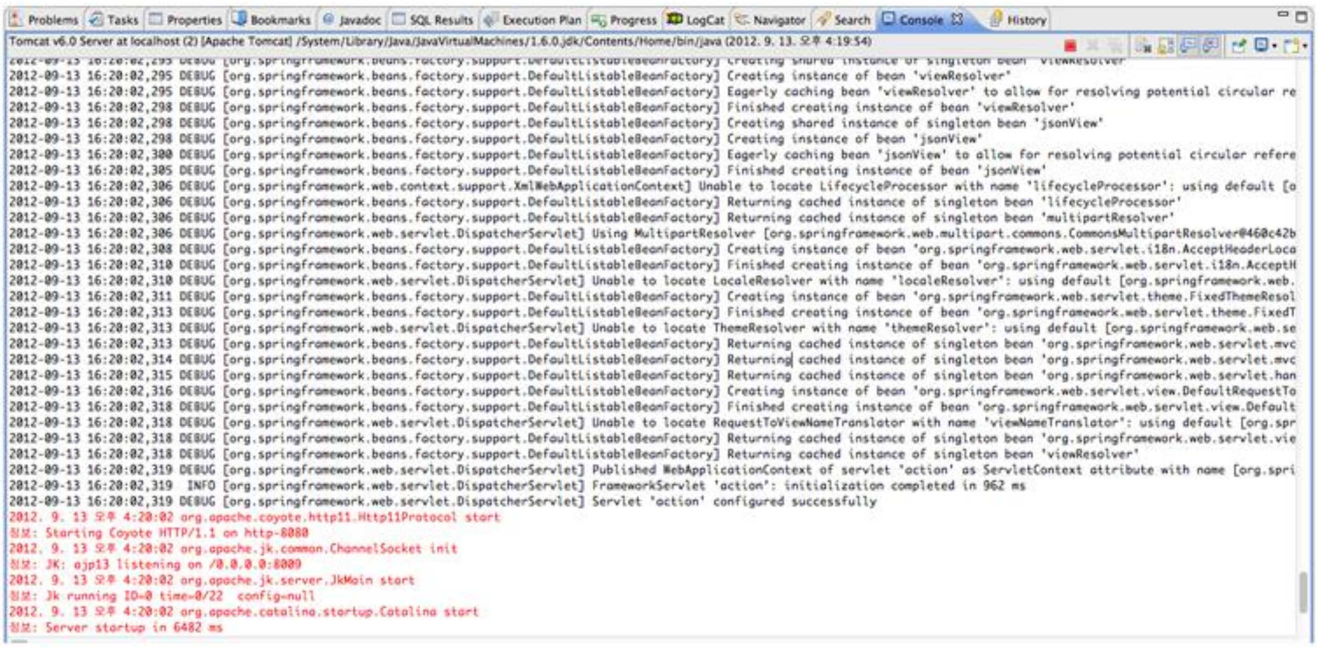
Copyright(c)2011 Ministry of Public Administration and Security.

How to compile NPKI Server Applicaton

- Right-click on the project and click on Run As>Run On Server in order to run the NPKIAPI server-side Guide Program.



- When the build is successfully completed, a message reading 'Server Startup in xxx ms' will display on the console window on the Eclipse.



```
Tomcat v6.0 Server at localhost (2) [Apache Tomcat] /System/Library/Java/JavaVirtualMachines/1.6.0.jdk/Contents/Home/bin/java (2012. 9. 13. 오후 4:19:54)
2012-09-13 16:20:02,295 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Creating shared instance of singleton bean 'viewResolver'
2012-09-13 16:20:02,295 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Creating instance of bean 'viewResolver'
2012-09-13 16:20:02,295 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Eagerly caching bean 'viewResolver' to allow for resolving potential circular re
2012-09-13 16:20:02,298 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Finished creating instance of bean 'viewResolver'
2012-09-13 16:20:02,298 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Creating shared instance of singleton bean 'jsonView'
2012-09-13 16:20:02,298 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Creating instance of bean 'jsonView'
2012-09-13 16:20:02,300 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Eagerly caching bean 'jsonView' to allow for resolving potential circular refere
2012-09-13 16:20:02,305 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Finished creating instance of bean 'jsonView'
2012-09-13 16:20:02,306 DEBUG [org.springframework.web.context.support.XmlWebApplicationContext] Unable to locate LifecycleProcessor with name 'lifecycleProcessor': using default [o
2012-09-13 16:20:02,306 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Returning cached instance of singleton bean 'lifecycleProcessor'
2012-09-13 16:20:02,306 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Returning cached instance of singleton bean 'multipartResolver'
2012-09-13 16:20:02,306 DEBUG [org.springframework.web.servlet.DispatcherServlet] Using MultipartResolver [org.springframework.web.multipart.commons.CommonsMultipartResolver@460c42b
2012-09-13 16:20:02,308 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Creating instance of bean 'org.springframework.web.servlet.i18n.AcceptHeaderLoca
2012-09-13 16:20:02,310 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Finished creating instance of bean 'org.springframework.web.servlet.i18n.AcceptH
2012-09-13 16:20:02,310 DEBUG [org.springframework.web.servlet.DispatcherServlet] Unable to locate LocaleResolver with name 'localeResolver': using default [org.springframework.web.
2012-09-13 16:20:02,311 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Creating instance of bean 'org.springframework.web.servlet.theme.FixedThemeResol
2012-09-13 16:20:02,313 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Finished creating instance of bean 'org.springframework.web.servlet.theme.FixedT
2012-09-13 16:20:02,313 DEBUG [org.springframework.web.servlet.DispatcherServlet] Unable to locate ThemeResolver with name 'themeResolver': using default [org.springframework.web.se
2012-09-13 16:20:02,313 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Returning cached instance of singleton bean 'org.springframework.web.servlet.mvc
2012-09-13 16:20:02,314 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Returning cached instance of singleton bean 'org.springframework.web.servlet.mvc
2012-09-13 16:20:02,315 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Returning cached instance of singleton bean 'org.springframework.web.servlet.han
2012-09-13 16:20:02,316 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Creating instance of bean 'org.springframework.web.servlet.view.DefaultRequestTo
2012-09-13 16:20:02,318 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Finished creating instance of bean 'org.springframework.web.servlet.view.Default
2012-09-13 16:20:02,318 DEBUG [org.springframework.web.servlet.DispatcherServlet] Unable to locate RequestToViewNameTranslator with name 'viewNameTranslator': using default [org.spr
2012-09-13 16:20:02,318 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Returning cached instance of singleton bean 'org.springframework.web.servlet.vie
2012-09-13 16:20:02,318 DEBUG [org.springframework.beans.factory.support.DefaultListableBeanFactory] Returning cached instance of singleton bean 'viewResolver'
2012-09-13 16:20:02,319 DEBUG [org.springframework.web.servlet.DispatcherServlet] Published WebApplicationContext of servlet 'action' as ServletContext attribute with name [org.spri
2012-09-13 16:20:02,319  INFO [org.springframework.web.servlet.DispatcherServlet] FrameworkServlet 'action': initialization completed in 962 ms
2012-09-13 16:20:02,319 DEBUG [org.springframework.web.servlet.DispatcherServlet] Servlet 'action' configured successfully
2012. 9. 13 오후 4:20:02 org.apache.coyote.http11.Http11Protocol start
정보: Starting Coyote HTTP/1.1 on http-8080
2012. 9. 13 오후 4:20:02 org.apache.jk.common.ChannelSocket init
정보: JK: ajp13 listening on /0.0.0.0:8009
2012. 9. 13 오후 4:20:02 org.apache.jk.server.JkMain start
정보: Jk running ID=0 time=0/22  config=null
2012. 9. 13 오후 4:20:02 org.apache.catalina.startup.Catalina start
정보: Server startup in 6482 ms
```

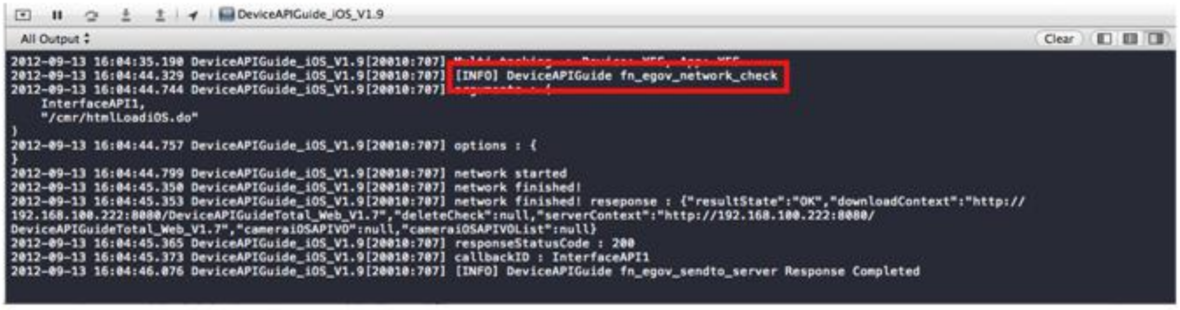
Debugging

Use console.log in order to check the details on any errors on the device application, and to conduct debugging. Debug codes in console.log are available in JavaScript syntaxes that you can use in both Eclipse and Xcode.

- Example of actual console log

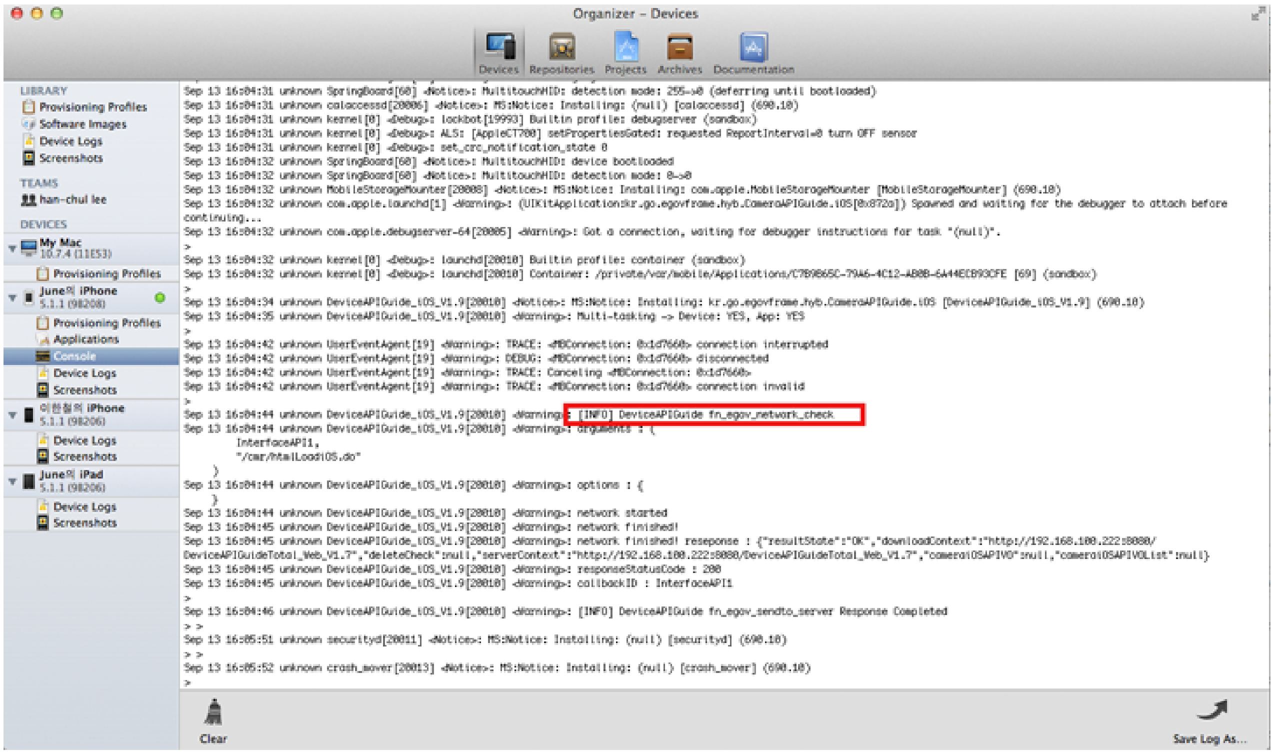
```
function fn_egov_network_check(doCheck)
{
        console.log('DeviceAPIGuide fn_egov_network_check');
        var networkState = navigator.network.connection.type;
        ...
}
```

- xCode console window



```
2012-09-13 16:04:35.190 DeviceAPIGuide_iOS_V1.9[20010:707] Multi-tasking -> Device: YES, App: YES
2012-09-13 16:04:44.329 DeviceAPIGuide_iOS_V1.9[20010:707] [INFO] DeviceAPIGuide fn_egov_network_check
2012-09-13 16:04:44.744 DeviceAPIGuide_iOS_V1.9[20010:707] arguments : (
    InterfaceAPI1,
    "/cmr/htmlLoadiOS.do"
)
2012-09-13 16:04:44.757 DeviceAPIGuide_iOS_V1.9[20010:707] options : {
}
2012-09-13 16:04:44.799 DeviceAPIGuide_iOS_V1.9[20010:707] network started
2012-09-13 16:04:45.350 DeviceAPIGuide_iOS_V1.9[20010:707] network finished!
2012-09-13 16:04:45.353 DeviceAPIGuide_iOS_V1.9[20010:707] network finished! reseponse : {"resultState":"OK","downloadContext":"http://
192.168.100.222:8080/DeviceAPIGuideTotal_Web_V1.7","deleteCheck":null,"serverContext":"http://192.168.100.222:8080/
DeviceAPIGuideTotal_Web_V1.7","cameraiOSAPIVO":null,"cameraiOSAPIVOList":null}
2012-09-13 16:04:45.365 DeviceAPIGuide_iOS_V1.9[20010:707] responseStatusCode : 200
2012-09-13 16:04:45.373 DeviceAPIGuide_iOS_V1.9[20010:707] callbackID : InterfaceAPI1
2012-09-13 16:04:46.076 DeviceAPIGuide_iOS_V1.9[20010:707] [INFO] DeviceAPIGuide fn_egov_sendto_server Response Completed
```

- Organizer log window



```
Sep 13 16:04:31 unknown SpringBoard[60] <Notice>: MultitouchHID: detection mode: 255->0 (deferring until bootloaded)
Sep 13 16:04:31 unknown calaccessd[20006] <Notice>: MS:Notice: Installing: (null) [calaccessd] (690.10)
Sep 13 16:04:31 unknown kernel[0] <Debug>: lockbot[19993] Builtin profile: debugserver (sandbox)
Sep 13 16:04:31 unknown kernel[0] <Debug>: ALS: [AppleCT700] setPropertiesGated: requested ReportInterval=0 turn OFF sensor
Sep 13 16:04:31 unknown kernel[0] <Debug>: set_crc_notification_state 0
Sep 13 16:04:32 unknown SpringBoard[60] <Notice>: MultitouchHID: device bootloaded
Sep 13 16:04:32 unknown SpringBoard[60] <Notice>: MultitouchHID: detection mode: 0->0
Sep 13 16:04:32 unknown MobileStorageMounter[20008] <Notice>: MS:Notice: Installing: com.apple.MobileStorageMounter [MobileStorageMounter] (690.10)
Sep 13 16:04:32 unknown com.apple.launchd[1] <Warning>: (UIKitApplication:kr.go.egovframe.hyb.CameraAPIGuide.iOS[0x872a]) Spawned and waiting for the debugger to attach before
continuing...
Sep 13 16:04:32 unknown com.apple.debugserver-64[20005] <Warning>: Got a connection, waiting for debugger instructions for task "(null)".
>
Sep 13 16:04:32 unknown kernel[0] <Debug>: launchd[20010] Builtin profile: container (sandbox)
Sep 13 16:04:32 unknown kernel[0] <Debug>: launchd[20010] Container: /private/var/mobile/Applications/C7B9B65C-79A6-4C12-AB0B-6A44ECB93CFE [69] (sandbox)
>
Sep 13 16:04:34 unknown DeviceAPIGuide_iOS_V1.9[20010] <Notice>: MS:Notice: Installing: kr.go.egovframe.hyb.CameraAPIGuide.iOS [DeviceAPIGuide_iOS_V1.9] (690.10)
Sep 13 16:04:35 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: Multi-tasking -> Device: YES, App: YES
>
Sep 13 16:04:42 unknown UserEventAgent[19] <Warning>: TRACE: <MBConnection: 0x1d7660> connection interrupted
Sep 13 16:04:42 unknown UserEventAgent[19] <Warning>: DEBUG: <MBConnection: 0x1d7660> disconnected
Sep 13 16:04:42 unknown UserEventAgent[19] <Warning>: TRACE: Canceling <MBConnection: 0x1d7660>
Sep 13 16:04:42 unknown UserEventAgent[19] <Warning>: TRACE: <MBConnection: 0x1d7660> connection invalid
>
Sep 13 16:04:44 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: [INFO] DeviceAPIGuide fn_egov_network_check
Sep 13 16:04:44 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: arguments : (
        InterfaceAPI1,
        "/cmr/htmlLoadiOS.do"
    )
Sep 13 16:04:44 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: options : {
    }
Sep 13 16:04:44 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: network started
Sep 13 16:04:45 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: network finished!
Sep 13 16:04:45 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: network finished! reseponse : {"resultState":"OK","downloadContext":"http://192.168.100.222:8080/
DeviceAPIGuideTotal_Web_V1.7","deleteCheck":null,"serverContext":"http://192.168.100.222:8080/DeviceAPIGuideTotal_Web_V1.7","cameraiOSAPIVO":null,"cameraiOSAPIVOList":null}
Sep 13 16:04:45 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: responseStatusCode : 200
Sep 13 16:04:45 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: callbackID : InterfaceAPI1
>
Sep 13 16:04:46 unknown DeviceAPIGuide_iOS_V1.9[20010] <Warning>: [INFO] DeviceAPIGuide fn_egov_sendto_server Response Completed
> >
Sep 13 16:05:51 unknown securityd[20011] <Notice>: MS:Notice: Installing: (null) [securityd] (690.10)
> >
Sep 13 16:05:52 unknown crash_mover[20013] <Notice>: MS:Notice: Installing: (null) [crash_mover] (690.10)
>
```

NPKI API Guide Program will output the following console information for debugging.

Debug code	**Debug information**
PKIWizSignAPIGuide deviceready Success	Device ready successful
PKIWizSignAPIGuide fn_egov_makesign_ok request Completed	Certificate authentication from web server application successful
PKIWizSignAPIGuide fn_egov_makesign_ok request Failed	Certificate authentication from web server application failed

PKIWizSignAPIGuide	fn_egov_makesign_ok	Success	Certificate signing successful
PKIWizSignAPIGuide	fn_egov_makesign_fail	Failed	Certificate signing failed
PKIWizSignAPIGuide	fn_egov_getcertlistSuccess	Success	Certificate list inquiry successful
PKIWizSignAPIGuide	fn_egov_getcertlistFail	Failed	Certificate list inquiry failed
PKIWizSignAPIGuide	doKISAShowApp	Failed	Olleh Certificate call failed
PKIWizSignAPIGuide	doKISAShowApp	Completed	Olleh Certificate call succssful
PKIWizSignAPIGuide	makeCert	Completed	Olleh Certificate save succssful
PKIWizSignAPIGuide	makeCert	Failed	Olleh Certificate save failed

Distribution

Download NPKI Device API Guide: Click

References

- UX/UI library : jQuery MobileClick
- Phonegap 4.3.0 : Click
- NPKIAPI : KSign Inc. http://www.ksign.com
- Standard security API : http://www.gpki.go.kr/