http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:hyb3.5:guide:ios:npki_%EB%93%9C%EB%A6%BC%EC%8B%9C%ED%81%90%EB%A6%AC%ED%8B%B0

NPKI(MagicXsign) Device API Guide Program

Outline

NPKI(MagicXsign) 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	MagicXSign Library Application MagicXSign.js, PhoneGap.a, libMRSPC_LIB.a, libWDSTKI30_LIB.a, libKT_SCM_Client.a, MagicXSignPlugin.h

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

Dream Security Inc. Kim, Hyeong Geom 02-2233-5533 http://dreamsecurity.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().

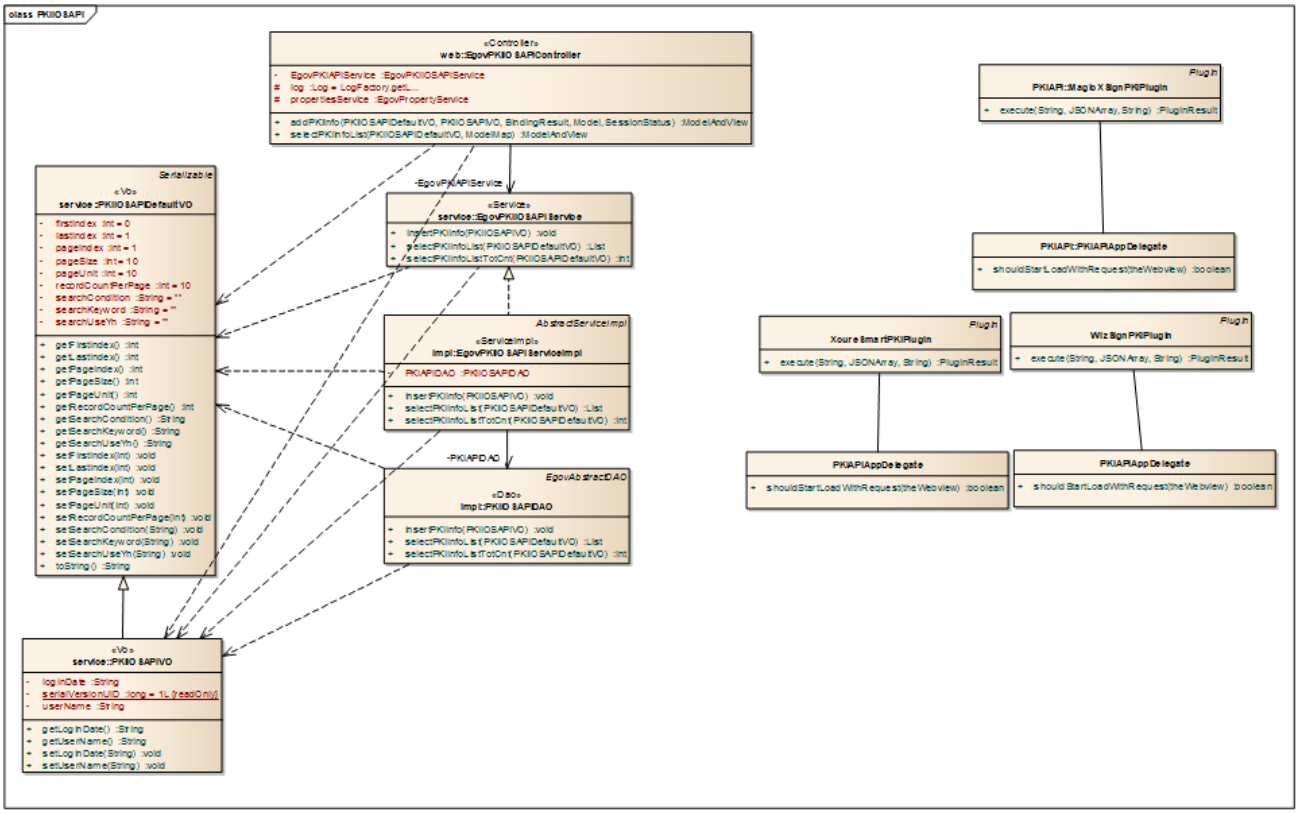
Using cross domain

When using certain outside domains or its subdomains on PhoneGap, add such domains on <access origin="ExternalHosts"/> at [Project_Name]/Supproting Files/config.xml

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



Device Application

Source

Type	Title	Remark

CSS	www/css/egovframwork/mbl/hyb/PKIMagicXSignAPI.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/PKIMagicXSignAPI.js	NPKIAPI Guide Program Core JavaScript
JS	www/js/egovframwork/mbl/hyb/MagicXSign.js	NPKIAPI Guide Program Core JavaScript
JS	www/js/egovframwork/mbl/hyb/messages_ko.js	JavaScript for Validate Message Processing
HTML	www/NPKIMagicXSignAPI.html	NPKIAPI Main Page
HTML	www/license.html	NPKIAPI License Page
HTML	www/overview.html	NPKIAPI Function Description Page

Function API

MagicXSign API DOC

APIs Used

window.plugins.magicxsign.init

- Conducts initialization in order to use MagicXSign.

window.plugins.magicxsign.init(var Flag)

Option	Description	Remark
Flag	MagicXSign Debug displayed Y/N	true: Yes, false: No

window.plugins.magicxsign.init("false");
window.plugins.magicxsign.init("true");

window.plugins.magicxsign.getcertlist

- Loads the device's certificate in the jsonString format.

window.plugins.magicxsign.getcertlist(success return , fail return, jsonString);

Option	Description	Remark
success return	Function returned upon success	
fail return	Returned upon failure	
jsonString	Certification Information Request	

jsonString

- Certification Information

Property	Description	Property	Description
issuer	Certification Issuing Organization	name	User Name
ver	Version	sn	Serial No.
issuedn	Issuer	start	Expiration Date (Start)
end	Expiration Date (Finish)	subjdn	Subject
pubkeyalgo	Public Key algorithm	pubkey	Public Key
aia	Institution information access	aki	Issuer key identifier
ski	Subject key identifier	keyuse	Key use
policy	Policy	policyid	Policy ID
subaltname	Subject alternative name	crl	CRL location

```
var setDefine = ["oidname","issuer","name","subjdn","start","end"];
window.plugins.magicxsign.getcertlist(getcertlistSuccess, getcertlistFail, JSON.stringify(setDefine));
```

window.plugins.magicxsign.certdel

- Delete Certificate.

```
window.plugins.magicxsign.certdel(success return, fail return, Cert Index);
```

Option	Description	Remark
success return	Function returned upon success	
fail return	Returned upon failure	
Cert Index	Certificate Index(from 0)	index value for window.plugins.magicxsign.getcertlist call result

```
window.plugins.magicxsign.certdel(certdelSuccess, certdelFail, 0);
```

window.plugins.magicxsign.certchagepassword

- Changes Certificate's password.

```
window.plugins.magicxsing.certchagepassword(success return, fail return, jsonString);
```

Option	Description	Remark
success return	Function returned upon success	
fail return	Returned upon failure	
jsonString	Certificate Index, sends old/new passwords via jsonString	

```
var setDefine = {};
setDefine["certindex"] = iCertIndex;        //selected Certificate index
setDefine["oldpassword"] = oldpassword;             //selected Certificate Password
setDefine["newpassword"] = newpassword;             //new index Password
```

window.plugins.magicxsing.certchagepassword(certchagepasswordSuccess, certchagepasswordFail, JSON.stringify(setDefine));

window.plugins.magicxsign.makesign

- Creates signature data.

window.plugins.magicxsing.makesign(success return, fail return, jsonString);

Option	Description	Remark
success return	Function returned upon success	
fail return	Returned upon failure	
jsonString	Certificate Index, sends old/new passwords via jsonString	

```
var setDefine = {};
setDefine["plaintext"] = encodeURIComplnent(MagicXSign_makeQueryString(form));
setDefine["url"] = sURL;
setDefine["certindex"] = document.getElementById("xsigncertindex").value;
setDefine["certpassword"] = document.getElementById("xsigncertpassword").value;
window.plugins.magicxsign.makesign(makesign_ok, makesign_fail, JSON.stringify(setDefine));
```

window.plugins.magicxsign.mrs_makecode

- Accesses authentication server to receive authentication number

window.plugins.magicxsign.mrs_makecode(success return, fail return, jsonString);

Option	Description	Remark
success return	Function returned upon success	
fail return	Returned upon failure	
jsonString	Json Strings of phoneno, serverip, serverport, serviceid format	

```
var setDefine = {};
setDefine["phoneno"] = "123456789";
setDefine["serverip"] = "125.141.204.173";
setDefine["serverport"] = "10001";
setDefine["serviceid"] = "dreamAPP";
window.plugins.magicxsign.makesign(makesign_ok, makesign_fail, JSON.stringify(setDefine));
```

window.plugins.magicxsign.mrs_waitcert

- Receives and saves Certification, returns Certification information.

window.plugins.magicxsign.mrs_makecode(success return, fail return, jsonString);

Option	Description	Remark
success return	Function returned upon success	
fail return	Returned upon failure	
jsonString	For viewing received Certification's information Refer to certlist function	

```
var setDefine = ["oidname","issuer","name","subjdn","start","end"]
window.plugins.magicxsign.mrs_waitcert(getcertlistSuccess, getcertlistFail,
JSON.stringify(setDefine));
```

window.plugins.magicxsign.mrs_stopcertmove

- Stops transfer of Certification.

```
window.plugins.magicxsign.mrs_stopcertmove();
```

window.plugins.magicxsign.ollecert_check

- Checks if Olleh Certificate is installed.

Option	Description	Remark
success	return Olleh Certificate is installed	
fail	return Olleh Certificate is not installed	

```
window.plugins.magicxsign.ollecert_check(ollecert_Check_Success, ollecert_Check_Fail);
```

window.plugins.magicxsign.ollecert_install

- Calls Olleh Certificate installation screen at the AppStore for Olleh Certificate installation

```
window.plugins.magicxsign.ollecert_install();
```

window.plugins.magicxsign.ollecert_getcert

- Requests Certificate from Olleh Certificate, and saves selected Certifiacte.

Option	Description	Remark
Yes return	Return when successful	JsonString – Return HandShakeinformation created by handshake field
No return	Return when error occurs	
jsonString	For viewing received Certification's information	Refer to certlist function

```
var setDefine = ["oidname","issuer","name","subjdn","start","end"];
window.plugins.magicxsign.ollecert_getcert(certGetFromOlleCert_Success,
certGetFromOlleCert_Fail, JSON.stringify(setDefine));
```

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
ServiceIim	egovframework.hyb.ios.pki.service.impl.EgovPKIiOSAPIServiceI	NPKIAPI Guide Program ServiceImpl C

pl	mpl.java	lass
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

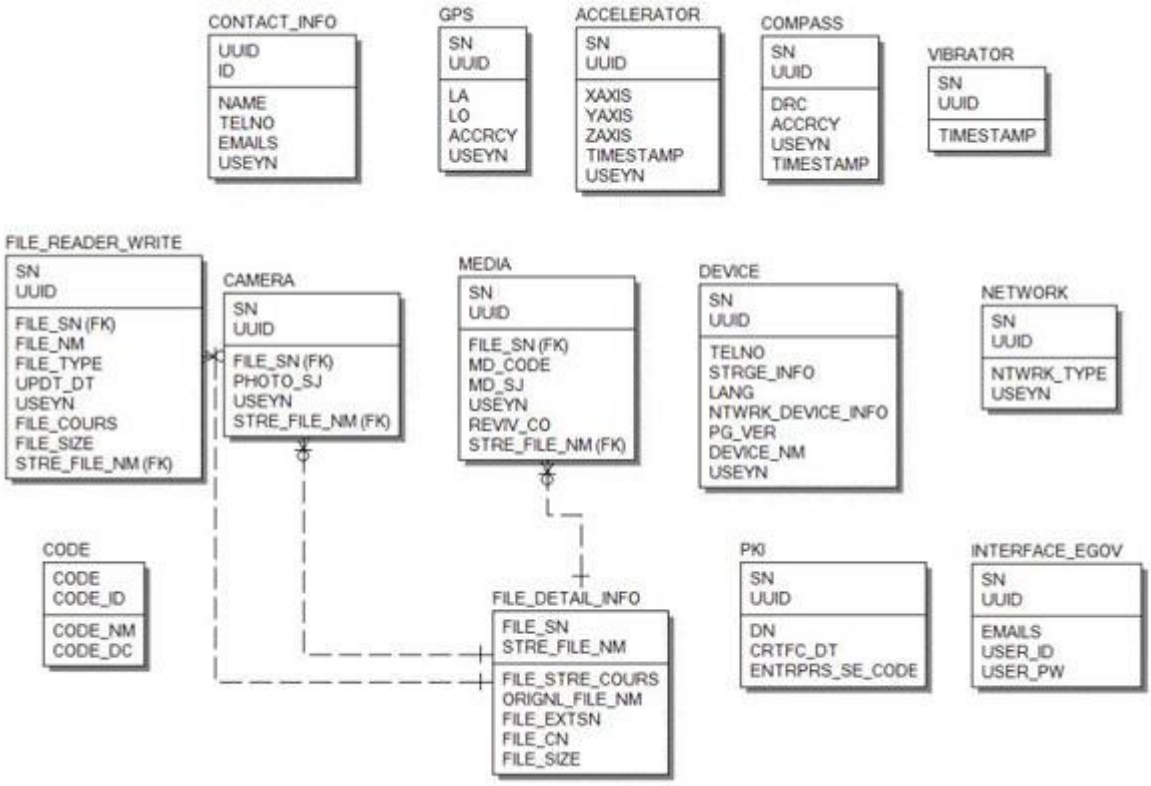
Title Table	Remark
PKI PKI	Certification Recognition Log Management

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	Certification Data	VARCHAR	255	Null
4	CFTFC_DT	Date and Time of Certification	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>
<!-- Phonegap Plugin class for using eGov NPKI Device API-->
<featurename="MagicXSignPlugin">
<paramname="ios-package"value="MagicXSignPlugin"/>
</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_show_certlist()
{
	console.log('DeviceAPIGuide fn_egov_show_certlist');
	$.mobile.showPageLoadingMsg('a');
	var setDefine = ["oidname","issuer","name","subjdn","start","end"];
	// request the data to the list designated in setDefine
	window.plugins.magicxsign.getcertlist(fn_egov_getcertlistSuccess,	fn_egov_getcertlistFail,
JSON.stringify(setDefine));
}

// signs the Certificate
function fn_egov_login_member() {
	console.log('fn_egov_login_member()');

	$.mobile.showPageLoadingMsg('a');

	var setDefine = {};
	setDefine["plaintext"]=encodeURIComponent("usrId=&password=&name=");	// form Data
	setDefine["certindex"]=document.getElementById("xsigncertindex").value;	// selected Cert
	setDefine["certpassword"]=$("#loginPasswd").val();
	window.plugins.magicxsign.makesign(fn_egov_makesign_ok, fn_egov_makesign_fail,
JSON.stringify(setDefine));
}

// requests authentication to Certificate signature data server
```

```
function fn_egov_makesign_ok(arg)
{
	console.log('DeviceAPIGuide fn_egov_makesign_ok');
	var jsonobj = JSON.parse(arg);

	var signedData = jsonobj.sign;
	var vidRandom = jsonobj.vidRandom;

	var acceptType = "json";
	var params = {    uuid :    device.uuid,
		sign : signedData,
		entrprsSeCode : "PKI01"};

	alert('Http Method:POST\nAcceptType:JSON\nSendData:' + JSON.stringify(params));
	$.mobile.showPageLoadingMsg('a');
	EgovInterface.submitAsynchronous(
						[params, "/pki/addPKIiOSInfo.do"],
						function(result) {
							console.log("PKIMagicXSignAPIGuide
fn_egov_makesign_ok 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("PKIMagicXSignAPIGuide
fn_egov_makesign_ok Failed");
							var str = '{';
							for (myKey in result){
								str += myKey + ' : ' + result[myKey] + '\n';
							}
							str += '}';
							$.mobile.hidePageLoadingMsg('a');
							alert('Response Method:RESTful\nSendType:json,
post\nParam:\n' + str);
						}
					);
	document.getElementById("innerhtmlArea").innerHTML= "SendResult : <br>"+str+"<br><br>";
}

// checks to see if Olleh Certificate is installed
function fn_egov_check_ollecert()
{
	console.log('fn_egov_check_ollecert()');
	window.plugins.magicxsign.ollecert_check(fn_egov_ollecert_Check_Success,
fn_egov_ollecert_Check_Fail);
}
```

```
// calls Olleh Certificate
function fn_egov_certGetFromOlleCert()
{
	console.log('fn_egov_certGetFromOlleCert()');
	var setDefine = ["oidname","issuer","name","subjdn","start","end"];
	window.plugins.magicxsign.ollecert_getcert( fn_egov_certGetFromOlleCert_Success,
fn_egov_certGetFromOlleCert_Fail, JSON.stringify(setDefine) );
}
```

Related Screen and Implementation Manual

Action	URL	Controller method	QueryID
Certificate authentication	/pki/addPKIiOSInfo.do	addPKIInfo	"PKIiOSAPIDAO.insertPKIInfo"

Certificate list

Certificate authentication

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_goLoginInfoList()
{
	console.log('fn_goLoginInfoList()');
	var accept_type = "json";
	// get the data from server
	$.mobile.showPageLoadingMsg('a');
	EgovInterface.submitAsynchronous(
                                                ["/pki/pkiInfoList.do"],
                                                function(result) {
                                                    console.log("PKIMagicXSignAPIGuide
fn_goLoginInfoList 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("PKIMagicXSignAPIGuide
fn_goLoginInfoList Failed");
                                        var str = '{';
                                        for (var myKey in error){
                                            str += myKey + ' : ' + error[myKey] + '\n';
                                        }
                                        str += '}';
                                        $.mobile.hidePageLoadingMsg('a');
                                        alert('Response Method:RESTful\nSendType:json,
post\nParam:\n' + str);
                                    }
                                );
}
```

Related Screen and Implementation Manual

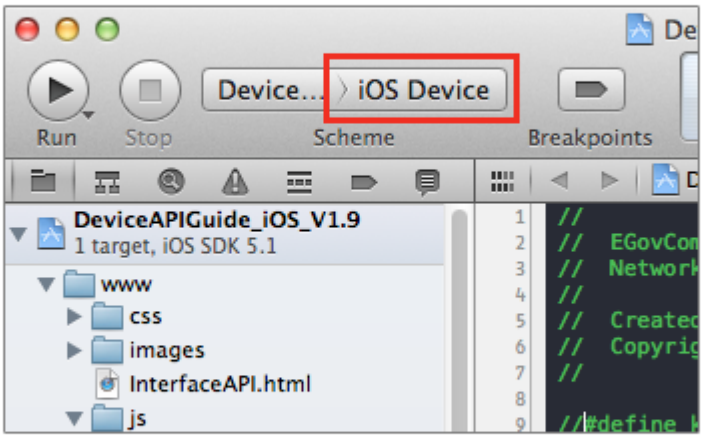
Function	URL	Controller	method	QueryID
Inquire Certificate authentication results log	/pki/pkiInfoList.do	EgovPKIiOSAPIController	selectPKIInfoList	PKIiOSAPIDAO.selectPKIInfoList

Compiling, debugging, distributing

Compiling

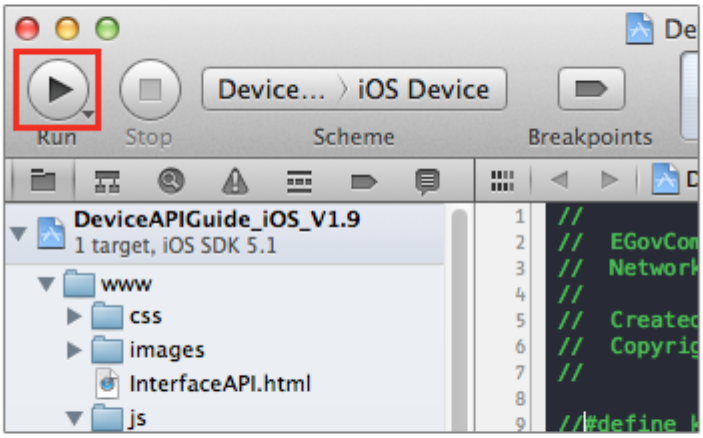
How to compile NPKI Device Applicaton

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



2. Select device or simulator.

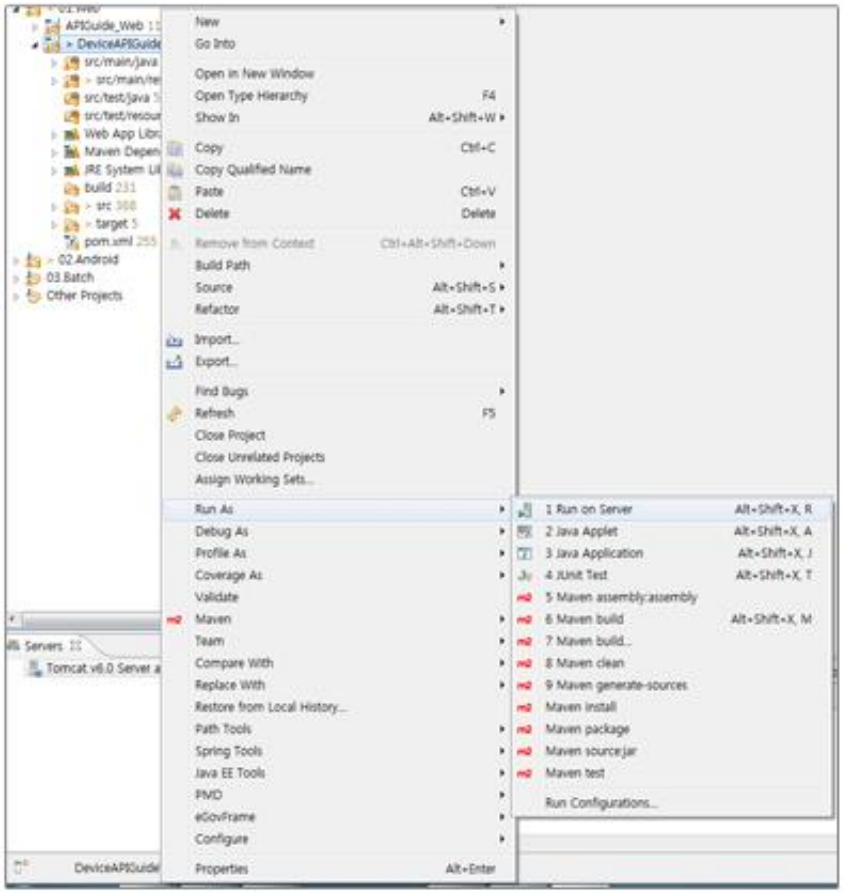
3. Click on "Execute."



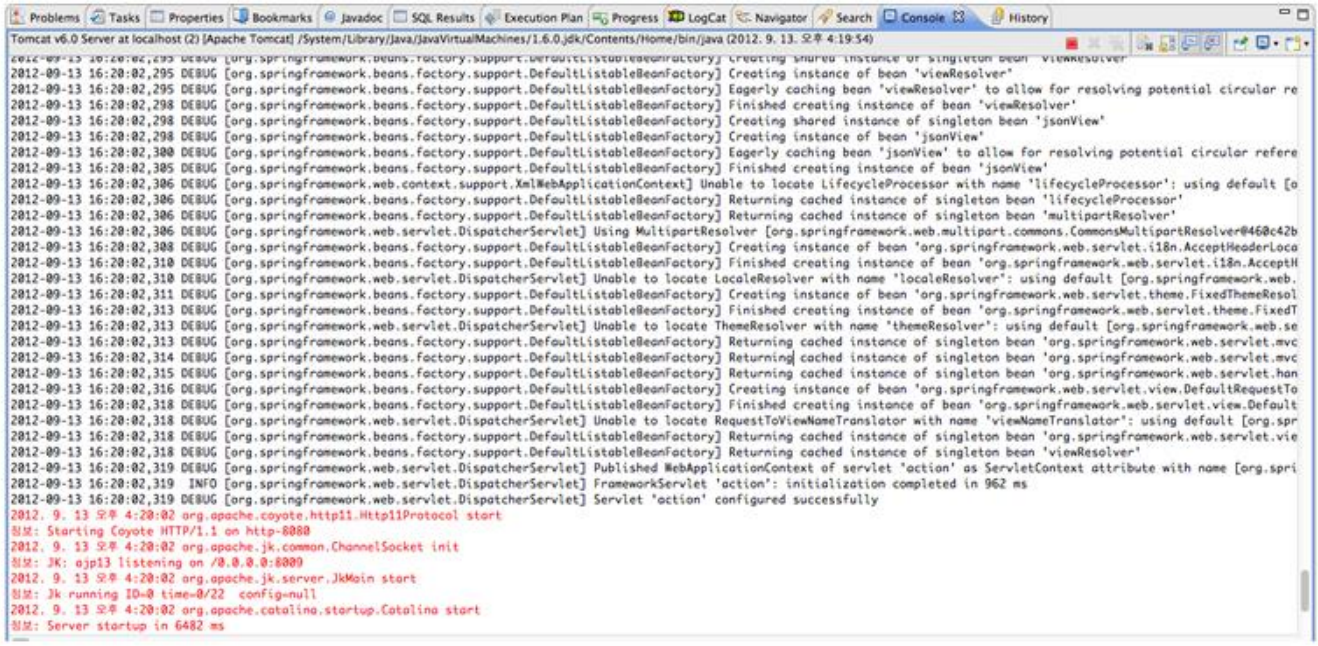
4. Check intro and main screen.

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.



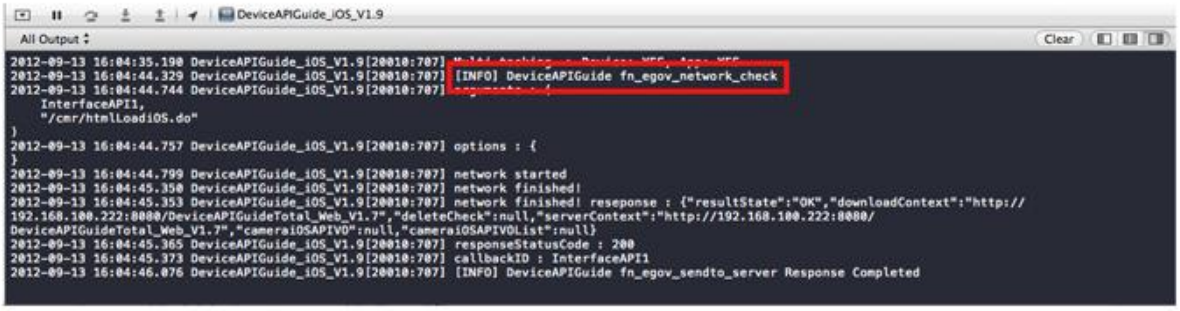
Debugging

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.

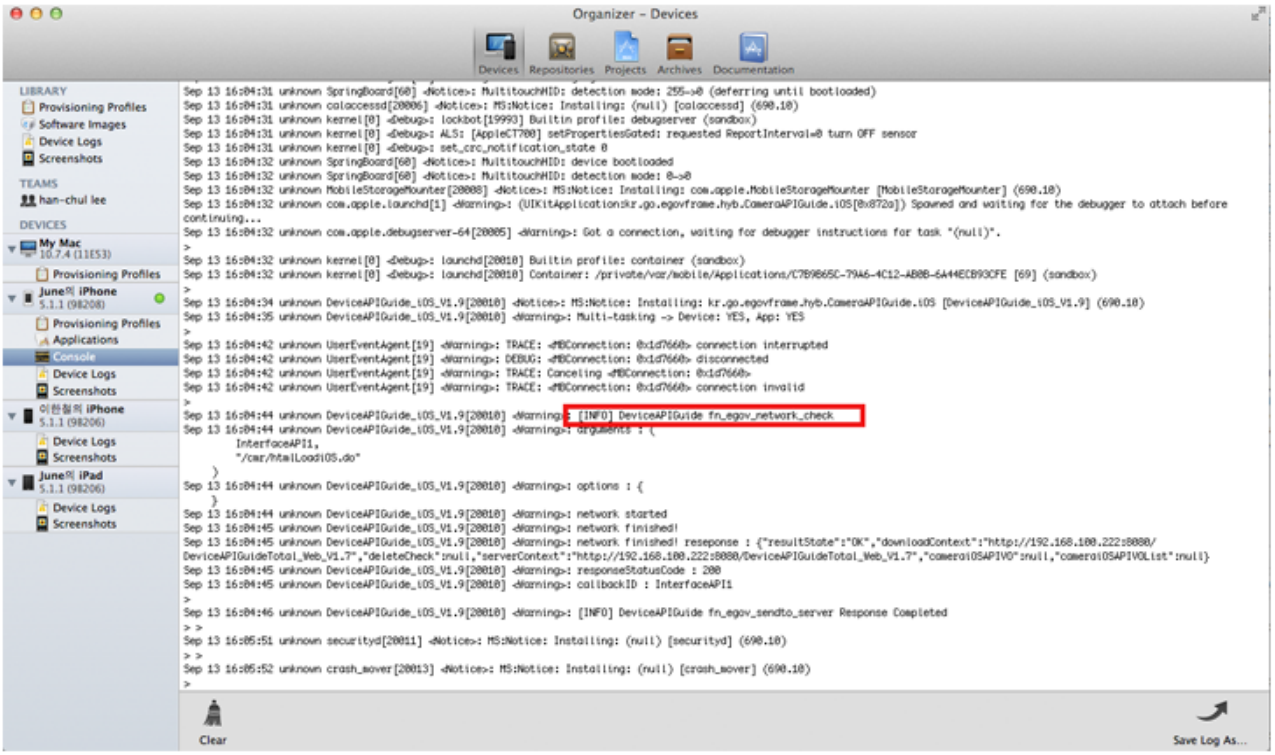
- Example of actual console log

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

- xCode console window



- Organizer log window



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

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

PKIMagicXSignAPIGuide	fn_egov_makesign_ok	Success	Certificate signing successful
PKIMagicXSignAPIGuide	fn_egov_makesign_fail	Failed	Certificate signing failed
PKIMagicXSignAPIGuide	fn_egov_getcertlistSuccess	Success	Certificate list inquiry successful
PKIMagicXSignAPIGuide	fn_egov_getcertlistFail	Failed	Certificate list inquiry failed
PKIMagicXSignAPIGuide	fn_egov_ollecert_Check_Fail	Failed	Olleh Certificate not installed
PKIMagicXSignAPIGuide	fn_egov_ollecert_Check_Fail	Completed	Olleh Certificate installed
PKIMagicXSignAPIGuide	fn_egov_certGetFromOlleCert_Fail	Failed	Olleh Certificate call failed
PKIMagicXSignAPIGuide	fn_egov_certGetFromOlleCert_Success	Success	Olleh Certificate call succssful

Distribution

Download NPKI Device API Guide: Click

References

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