http://www.egovframe.go.kr/wiki/doku.php?id=egovframework:hyb3.5:hrte:%EC%8B%A4%ED%96%89%ED%99%98%EA%B2%BD_%EC%98%88%EC%A0%9C

Outline

Using the runtime environment example for eGovFramework Device API, the developers have an opportunity to apply for sample application to execute Device API for PhoneGap by way of sample application.

Preconditions

Category	Description
Local Device Environments	eGovFramework Runtime Environment 3.5, Android SDK Revision 24 or better, X code 6.3, PhoneGap 4.3
Server-side Developmental Environment	N/A
Works in sync with Mash up Open API	N/A
Test Device	Galaxy S2, Galaxy S3, iPhone4, iPhone6
Test Platform	Android 2.2, Android 5.1, iOS 6.0, iOS7.1.2, iOS8.3
Libraries Added	N/A

Restriction

- N/A

Description

Using the runtime environment practices for eGovFramework Device API, the developers are given a total of 10 sample codings for use and reference when intending to implement the hybrid application.

Source

Type	Title	Remark
CSS	www/css/egovframwork/mbl/hyb/SampleTemplate.css	Cascading Style Sheets for Runtime Environment
IMAGE	www/images/egovframwork/mbl/hyb/	Image Folders for Runtime Environment
JS	www/js/egovframwork/mbl/hyb/SampleTemplate.js	JavaScript for Runtime Environment
HTML	www/SampleTemplate.html	Main Page for SampleTemplate
HTML	www/license.html	License Page for SampleTemplate
HTML	www/overview.html	Functions for SampleTemplate

Application API

Accelerator API

accelerometer.getCurrentAcceleration

- PhoneGap API for inquiry of Accelerometer Information

```
navigator.accelerometer.getCurrentAcceleration(accelerometerSuccess, accelerometerError);
function onSuccess(acceleration) {
    alert('Acceleration X: ' + acceleration.x + '\n' +
          'Acceleration Y: ' + acceleration.y + '\n' +
          'Acceleration Z: ' + acceleration.z + '\n' +
          'Timestamp: '      + acceleration.timestamp + '\n');
};
```

acceleratorOption

Option	Description	Remark
frequency	Frequency of inquiry of acceleration information	

GPS API

navigator.geolocation.getCurrentPosition

- PhoneGap API for geolocation information

```
navigator.geolocation.getCurrentPosition
navigator.geolocation.getCurrentPosition(successCallback, failCallback);
```

Option	Description	Remark
successCallback	Returned upon success	
failCallback	Returned upon failure	

Vibrator API

notification.beep

- Notifies beep in the device
- times: Beep counts repeted

```
navigator.notification.beep(times);
```

notification.vibrate

- Vibrates the device for the pre-determined period of time
- Lasting of vibration in milliseconds

```
navigator.notification.vibrate(milliseconds);
```

Camera API

camera.getPicture

- Loads picture shot or in album
- Loads images encoded in base64 or in album

```
navigator.camera.getPicture( cameraSuccess, cameraError, [ cameraOptions ] );
```

cameraOption

```
var cameraOption = {};
  cameraOption    = {
        quality: 50,
        destinationType : Camera.DestinationType.FILE_URI,
        sourceType : Camera.PictureSourceType.CAMERA,
        targetWidth: 200,
        targetHeight: 200
  };
```

Option	Description	Remark
quality	Defines quality (resolution) of image in percentage terms	
destinationType	Defines type of the destination value	navigator.camera.DestinationType
sourceType	Configures type of source for photo library and picture shot	nagivator.camera.PictureSourceType
allowEdit	Defines editability of the pre-selected image	
encodingType	Defines encoding type	navigator.camera.EncodingType
targetWidth	Configures target width (pixel) of image Rate fixed	
targetHeight	Configures target height (pixel) of image Rate fixed	
mediaType	Configures type of media when pictureSourceType is configured either for PHOTOLIBRARY or SAVEDPHOTOALBUM	nagivator.camera.MediaType
correctOrientation	Orientation of image for the images being loaded	
saveToPhotoAlbum	Saving image for the storage in the device	

Media API

Media

- Object used for playing or recording of audio files in the device

```
var media = new Media(src, mediaSuccess, [mediaError], [mediaStatus]);
```

media.getCurrentPosition

- Gets the current position of a media

```
media.getCurrentPosition(mediaSuccess, [mediaError]);
```

media.getDuration

- Gets the duration of a media file

```
media.getDuration();
```

media.pause

- Pauses a media file

```
media.pause();
```

media.play

- Plays a media file

```
media.play();
```

media.release

- Releases an audio file regiatered in OS or memory.

```
media.release();
```

mediaError

- Media API error code

Error Code	Description	Remark
MEDIA_ERR_ABORTED	Plays aborted media	
MEDIA_ERR_NETWORK	Network error encountered	
MEDIA_ERR_DECODE	Decoding error (codec error)	
MEDIA_ERR_SRC_NOT_SUPPORTED	Media not supported	

Contacts API

contacts.create

- Create a new contact object

```
var myContact = navigator.contacts.create({"displayName": "Test User"});
```

contacts.find

- Finds a contact

```
navigator.contacts.find(contactFields, contactSuccess, contactError, contactFindOptions);
var options = new ContactFindOptions();
options.filter="Bob";
var fields = ["displayName", "name"];
navigator.contacts.find(fields, onSuccess, onError, options);
```

- Saves a contact

```
// create a new contact object
var contact = navigator.contacts.create();
contact.displayName = "Plumber";       // not support iOS
contact.nickname = "Plumber";           //specify both to support all devices
```

```
// populate some fields
var name = new ContactName();
name.givenName = "Jane";
name.familyName = "Doe";
contact.name = name;

// save to device
contact.save(onSuccess,onError);
```

- Clones a contact

```
var clone = contact.clone();
clone.name.givenName = "John";
console.log("Original contact name = " + contact.name.givenName);
console.log("Cloned contact name = " + clone.name.givenName);
```

- Removes a contact

```
function onSuccess() {
    alert("Removal Success");
};

function onError(contactError) {
    alert("Error = " + contactError.code);
};

// remove the contact from the device
contact.remove(onSuccess,onError);
```

- contactFields

Properties	Object Structure	Remark
ID	string	Global Unique Identifier (DOMString)
displayName	(DOMString)	Display name of a contact
name	**(ContactName)**	Object that contains the entire components
nickname	(DOMString)	Casual name where contact can be made
phoneNumbers	**(ContactField [])**	Phone contact field
email	**(ContactField [])**	E-mail contact field
addresses	**(ContactAddress [])**	Address field
ims	**(ContactField [])**	IM Address field
organizations	**(ContactOrganization [])**	Organization field

birthday	**Date**	Birth date of the contact (birthdate)
note	(DOMString)	Remark on the contact
photo	**(ContactField [])**	Photo field
categories	**(ContactField [])**	Customized array
urls	**(ContactField [])**	Web page array

- ContactName

Properties	Object Structure	Remark
formatted	(DOMString)	Full name of the contact
familyName	(DOMString)	
givenName	(DOMString)	
middleName	(DOMString)	
honorificPrefix	(DOMString)	Honorific prefix
honorificSuffix	(DOMString)	Honorific suffix

- ContactField

Properties	Object Structure	Remark
type	(DOMString)	Field category arrays
value	(DOMString)	Field value
pref	(Boolean)	Preferred value

- ContactAddress

Properties	Object Structure	Remark
pref	(boolean)	Representative value of ContactAddress
type	(DOMString)	Field types
formatted	(DOMString)	Address information for output
streetAddress	(DOMString)	Street information
locality	(DOMString)	City / County

region	(DOMString)	State / Region
postalCode	(DOMString)	Postal code
country	(DOMString)	Area code

- ContactOrganization

Properties	Object Structure	Remark
pref	(Boolean)	Representative value of ContactAddress
type	(DOMString)	Definition of field types (e.g. ‘work’)
name	(DOMString)	Name of the affiliation
department	(DOMString)	Department in the affiliation
title	(DOMString)	Title

- ContactFindOption

Properties	Object Structure	Remark
filter	(DOMString)	Keyword and search conditions (Default: ””)
multiple	(Boolean)	Loading of multiple contact information

- ContactError

Properties	Object Structure	Remark
ContactError.UNKNOWN_ERROR	(DOMString)	Unknown error
ContactError.INVALID_ARGUMENT_ERROR	(DOMString)	Wrong argument
ContactError.TIMEOUT_ERROR	(DOMString)	Timeout error
ContactError.PENDING_OPERATION_ERROR	(DOMString)	Invalid command
ContactError.IO_ERROR	(DOMString)	I/O error
ContactError.NOT_SUPPORTED_ERROR	(DOMString)	Not supported
ContactError.PERMISSION_DENIED_ERROR	(DOMString)	Permission denied

Compass API

compass.watchHeading

- Loads the heading information of the device

```
navigator.compass.getCurrentHeading(compassSuccess, compassError, compassOptions);
function onSuccess(heading) {
    alert('Heading: ' + heading.magneticHeading);
};

function onError(error) {
    alert('CompassError: ' + error.code);
};

navigator.compass.getCurrentHeading(onSuccess, onError);
```

compassOption

Option	Description	Remark
frequency	Frequency of inquiry of compass information	
frequency	Refers to the change required for initialization of the callback function	

FileReaderWriter API

LocalFileSystem

- Load the system information out of the mobile device

```
window.requestFileSystem(LocalFileSystem.PERSISTENT, 0, onSuccess, onError);
```

Parameter	Description	Remark
LocalFileSystem.PERSISTENT	Inquires the storages that cannot be removed by either user or application	
onSuccess,	Callback function called upon successful inquiry to the files system	
onError	Callback function called upon inquiry failure	

NetworkInfo API

navigator.connection.type

- Network status information for the current device

```
var states = {};
states[Connection.UNKNOWN]  = 'Unknown connection';
states[Connection.ETHERNET] = 'Ethernet connection';
states[Connection.WIFI]     = 'WiFi connection';
states[Connection.CELL_2G]  = 'Cell 3G connection';
states[Connection.CELL_3G]  = 'Cell 3G connection';
states[Connection.CELL_4G]  = 'Cell 4G connection';
states[Connection.NONE]     = 'No network connection';
```

```
var NowNetwork = states[navigator.connection.type];
```

Return	State(code)	NetworkInfo(string)
Connection.UNKNOWN	Unknown connection	
Connection.ETHERNET	Ethernet connection	
Connection.WIFI	WiFi connection	
Connection.CELL_2G	Cell 2G connection	
Connection.CELL_3G	Cell 3G connection	
Connection.CELL_4G	Cell 4G connection	
Connection.NONE	No network connection	

DeviceInfo API

device

- Inquiry to meta information related to both hardware and software of the mobile device

```
var name = device.name;
var cordova= device.cordova;
var platform = device.platform;
var uuid = device.uuid;
var version = device.version;
```

Parameter	Description	Remark
name	Returns the pre-configured name	
cordova	Returns the PhoneGap version information	
platform	Returns the platform information of the mobile device	
uuid	Returns UUID of the mobile device	
version	Returns the platform version of the mobile device	

Properties

Device properties required for use of Device API provided by the runtime environment practice program are as follows:

Device Application

Android

- app/res/xml/plugins.xml

```
<plugins>
```

<pluginname="App"value="org.apache.cordova.App"/>
<pluginname="Geolocation"value="org.apache.cordova.GeoBroker"/>
<pluginname="Device"value="org.apache.cordova.Device"/>
<pluginname="Accelerometer"value="org.apache.cordova.AccelListener"/>
<pluginname="Compass"value="org.apache.cordova.CompassListener"/>
<pluginname="Media"value="org.apache.cordova.AudioHandler"/>
<pluginname="Camera"value="org.apache.cordova.CameraLauncher"/>
<pluginname="Contacts"value="org.apache.cordova.ContactManager"/>
<pluginname="File"value="org.apache.cordova.FileUtils"/>
<pluginname="NetworkStatus"value="org.apache.cordova.NetworkManager"/>
<pluginname="Notification"value="org.apache.cordova.Notification"/>
<pluginname="Storage"value="org.apache.cordova.Storage"/>
<pluginname="Temperature"value="org.apache.cordova.TempListener"/>
<pluginname="FileTransfer"value="org.apache.cordova.FileTransfer"/>
<pluginname="Capture"value="org.apache.cordova.Capture"/>
<pluginname="Battery"value="org.apache.cordova.BatteryListener"/>
<pluginname="SplashScreen"value="org.apache.cordova.SplashScreen"/>
</plugins>

- app/AndroidManifest

<supports-screens
android:largeScreens="true"
android:normalScreens="true"
android:smallScreens="true"
android:resizeable="true"
android:anyDensity="true"/>
<uses-permissionandroid:name="android.permission.VIBRATE"/>
<uses-permissionandroid:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permissionandroid:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-
permissionandroid:name="android.permission.ACCESS_LOCATION_EXTRA_COMMANDS"/>
<uses-permissionandroid:name="android.permission.READ_PHONE_STATE"/>
<uses-permissionandroid:name="android.permission.INTERNET"/>
<uses-permissionandroid:name="android.permission.RECEIVE_SMS"/>
<uses-permissionandroid:name="android.permission.RECORD_AUDIO"/>
<uses-permissionandroid:name="android.permission.MODIFY_AUDIO_SETTINGS"/>
<uses-permissionandroid:name="android.permission.READ_CONTACTS"/>
<uses-permissionandroid:name="android.permission.WRITE_CONTACTS"/>
<uses-permissionandroid:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permissionandroid:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permissionandroid:name="android.permission.GET_ACCOUNTS"/>
<uses-permissionandroid:name="android.permission.BROADCAST_STICKY"/>

iOS

- App/Supporting Files/Cordova.plist

<dict>
<key>Logger</key>
<string>CDVLogger</string>
<key>Compass</key>
<string>CDVLocation</string>

```
        <key>Accelerometer</key>
        <string>CDVAccelerometer</string>
        <key>Camera</key>
        <string>CDVCamera</string>
        <key>NetworkStatus</key>
        <string>CDVConnection</string>
        <key>Contacts</key>
        <string>CDVContacts</string>
        <key>Debug Console</key>
        <string>CDVDebugConsole</string>
        <key>File</key>
        <string>CDVFile</string>
        <key>FileTransfer</key>
        <string>CDVFileTransfer</string>
        <key>Geolocation</key>
        <string>CDVLocation</string>
        <key>Notification</key>
        <string>CDVNotification</string>
        <key>Media</key>
        <string>CDVSound</string>
        <key>Capture</key>
        <string>CDVCapture</string>
        <key>SplashScreen</key>
        <string>CDVSplashScreen</string>
        <key>Battery</key>
        <string>CDVBattery</string>
        <key>InterfaceAPI</key>
        <string>EgovInterface</string>
        <key>StorageInfoAPI</key>
        <string>EgovStorageInfo</string>
</dict>
```

Functions

The runtime environment practice program comprises a set of simple examples to better understand Device API functions.

Inquiry of Acceleration Information

Business Logic

- In iOS platform practices, acceleration information is updated on a regular basis. Access the tab Accelerator to stop the inquiry manually.

Related Codes

Updating Acceleration Information

```
function fn_egov_update_Acceleration(accelInfo)
{
      var html = "<span>X : " + accelInfo.x + "<BR />" + "Y : " +    accelInfo.y + "<BR />" + "Z : "
+ accelInfo.z+"</span>";
            $("#infoDetail").html(html);
```

```
	if(accelInsertCheck)
	{
		if(accelInfo.x === 0 && accelInfo.y === 0 && accelInfo.z === 0)
		{

		}
		else
		{
			fn_egov_insert_table("ACCELERATOR",accelInfo);
			fn_egov_display_deviceAPIInfoMain("ACCELERATOR",html);

			accelInsertCheck = false;
		}
	}
}
```

Running Accelerator and Configuring Update Timer

```
function fn_egov_get_acceleration()
{
	if (accelerationWatch === null)
	{
		//navigator.notification.alert("acceleration start");
		toast("acceleration start");

		var options = {};
		options.frequency = 1000;
		accelInsertCheck = true;
		accelerationWatch =
navigator.accelerometer.watchAcceleration(fn_egov_update_Acceleration,
                                                                                function(ex)
                                                                                {

console.log("DeviceAPIGuide fn_egov_get_acceleration fail (" + ex.name + ": " + ex.message + ")");
                                                                                },
                                                                                options);
	}
	else
	{
		//navigator.notification.alert("acceleration stop");
		toast("acceleration stop");
		navigator.accelerometer.clearWatch(accelerationWatch);
		accelerationWatch = null;
	}
};
```

Related Screen and Implementation Manual

1. Inquiry for Acceleration Information

Inquiry of GPS Information

Related Codes

Received GPS information

```
function fn_egov_get_location()
{
	var suc = function(p)
	{
		var html = "<span>latitude : " + p.coords.latitude + "<BR />" + "longitude : " +
p.coords.longitude + "<BR />" + fn_egov_get_nowTime()+"</span>";
		fn_egov_display_deviceAPIInfoMain("GPS",html);
		fn_egov_insert_table("GPS",p);
	};
	var locFail = function()
	{
		jAlert("Cannot receive location information.", "Alert", "b");
	};

	navigator.geolocation.getCurrentPosition(suc, locFail);
};
```

Related Screen and Implementation Manual

1. Search Device latitude, longitude information

Vibration Alert

Related Codes

Device Vibration Alert

```
function fn_egov_get_vibrate()
{
	var html = "<span><img src='images/egovframework/mbl/hyb/ico_vibration.png' ></span><BR
/>" + fn_egov_get_nowTime();
	fn_egov_display_deviceAPIInfoMain("VIBRATE",html);
	fn_egov_insert_table("VIBRATOR","");
	//document.getElementById('showResult').value = str;
	navigator.notification.vibrate(2000);
};
```

Related Screen and Implementation Manual

1. Calls device vibration

Shoots photo

Related Codes

Calls device camera

```
function fn_egov_capture_photo()
{
	navigator.camera.getPicture(fn_egov_upload_photo,null,{sourceType:1,quality:60});
};
```

Callback function for successful shooting

```
function fn_egov_upload_photo(data)
{
	console.log("DeviceAPIGuide fn_egov_upload_photo success");
	var html = '<span class="camera"><img src="' + data + '" style="max-
width:100%;height:60px;"></img></span>';
	fn_egov_display_deviceAPIInfoMain("CAMERA",html);
	fn_egov_insert_table("CAMERA",data);

	toast("uploadPhoto success!");
};
```

Related Screen and Implementation Manual

1. Shoots photo

Plays media

Business Logic

- Clicks on media button to play / stop media

Related Codes

Plays media

```
function fn_egov_play_audio()
{
	if(audioCheck)
	{
		fn_egov_stop_audio();
		audioCheck = false;
		return;
	}

	// Create Media object from src
	mediaHandle = new Media("SleepAway.mp3", fn_egov_on_audioSuccess,
fn_egov_on_audioError);
	// Play audio
	mediaHandle.play();
	// Update mediaHandle position every second
	if (mediaTimer === null)
```

```
	{
		audioCheck = true;
		fn_egov_insert_table("MEDIA","");
		var html = "<span><img
src=\'images/egovframework/mbl/hyb/ico_movie.png\'></span><BR />" + fn_egov_get_nowTime();
		fn_egov_display_deviceAPIInfoMain("MEDIA",html);

		mediaTimer = setInterval(function()
					{
						// get mediaHandle position
						mediaHandle.getCurrentPosition(
								// success callback
								function(position)
								{
									if (position > -1)
{

fn_egov_set_audioPosition((position) + " %");
									}
								},
								// error callback
								function(e)
								{

console.log("DeviceAPIGuide fn_egov_play_audio Error "+e.code);

fn_egov_set_audioPosition("Error: " + e);
								});
					},
					1000);
	}
}
```

Pauses media

```
function fn_egov_pause_audio()
{
	if (mediaHandle)
	{
		mediaHandle.pause();
	}
}
```

Stops media

```
function fn_egov_stop_audio()
{
	if (mediaHandle)
	{
		mediaHandle.stop();
	}
	clearInterval(mediaTimer);
```

```
	mediaTimer = null;
}
```

Current audio position

```
function fn_egov_set_audioPosition(position)
{
	var html = "<span><img src=\'images/egovframework/mbl/hyb/ico_movie.png\'></span><BR />Play position : "+position+"<BR />" + fn_egov_get_nowTime();
	$("#infoDetail").html(html);
}
```

Related Screen and Implementation Manual

1. Plays media
2. Stops media

Inquiry of contact information

Related Codes

Request for getting contacts

```
function fn_egov_get_contacts()
{
	var obj = new ContactFindOptions();
	obj.filter = "";
	obj.multiple = true;
	navigator.contacts.find(
					[ "displayName", "name" ],
```

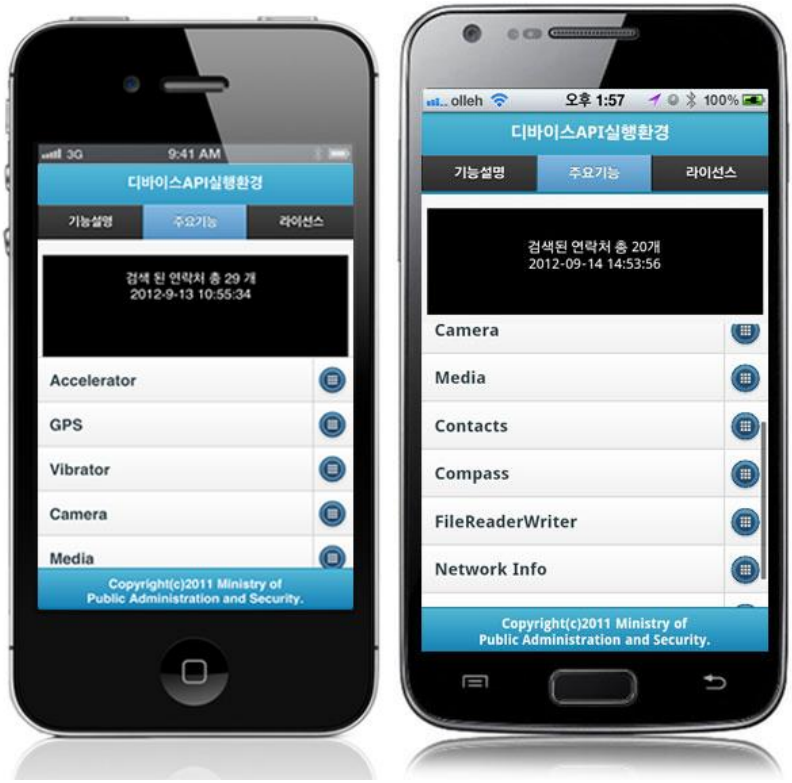
```
                                    fn_egov_get_contactsRead,
                                    fn_egov_get_contactsFail,
                                    obj);
}
```

Callback function for successful inquiry

```
function fn_egov_get_contactsRead(contacts)
{
    console.log("DeviceAPIGuide fn_egov_get_contactsRead Success");
    var html = "<span>Contacts searched" + contacts.length + "Number of contacts" + "<BR />" +
fn_egov_get_nowTime() + "</span>";
    fn_egov_display_deviceAPIInfoMain("CONTACTS",html);

    fn_egov_insert_table("CONTACTS","total contacts : "+contacts.length);
}
```

Related Screen and Implementation Manual



1. Inquires total number of contacts available in the device

Updates compass information

Business Logic

- In iOS platform practices, compass information is updated on a regular basis. Access the tab Accelerator to stop the inquiry manually.

Related Codes

Updates compass information

```
function fn_egov_get_compass()
{
	if (CompasswatchID === null)
	{
		fn_egov_display_deviceAPIInfoMain("COMPASS","");

		toast("Compass start");
		CompassInsertCheck = true;
		var options = { frequency: 1000 };
		CompasswatchID = navigator.compass.watchHeading(fn_egov_update_heading,
									function(e)
									{

console.log("DeviceAPIGuide fn_egov_get_compass Error "+e.code);
									},
									options);
	}
	else
	{
		navigator.compass.clearWatch(CompasswatchID);
		CompasswatchID = null;
		fn_egov_update_heading({ magneticHeading : "Off"});
		toast("Compass stop");
	}
}
```

Callback function for successful update

```
function fn_egov_update_heading(h)
{
	var html = "azimuth: " + h.magneticHeading + "<BR />" + fn_egov_get_nowTime();
	$("#infoDetail").html(html);
	if(CompassInsertCheck)
	{
		fn_egov_insert_table("COMPASS",h);
		CompassInsertCheck = false;
	}
}
```

Related Screen and Implementation Manual

1. Updating azimuth information

Inquires for files

Related Codes

Updates the local file system

```
function fn_egov_get_localStorageInfo()
{
	window.requestFileSystem(LocalFileSystem.PERSISTENT, 0,
					// success get file system
					function(fs)
					{
						console.log("DeviceAPIGuide
fn_egov_get_localStorageInfo Success");
						fileSystem = fs;
						dirEntry = fs.root;

					},
					// error get file system
					function(evt)
					{
						console.log("DeviceAPIGuide
fn_egov_get_localStorageInfo Error "+evt.target.error.code);
					});
}
```

Gets directory information

```
function fn_egov_go_directory(directoryEntry)
{
	console.log("DeviceAPIGuide fn_egov_go_directory Success");
		dirEntry = directoryEntry;
		fn_egov_read_directory();
}
```

Gets location information in the current file system

```
function fn_egov_next_chdir(dir)
{
		if (dir == "../")
	{
				dirEntry.getParent(fn_egov_go_directory, fn_egov_get_fileError);
		}
	else if (dir == "root")
	{
				dirEntry = fileSystem.root;
				fn_egov_read_directory();
		}
	else
	{
				dirEntry.getDirectory(dir, {}, fn_egov_go_directory, fn_egov_get_fileError);
		}
}
```

Related Screen and Implementation Manual

1. Updates the file system

Updates network information

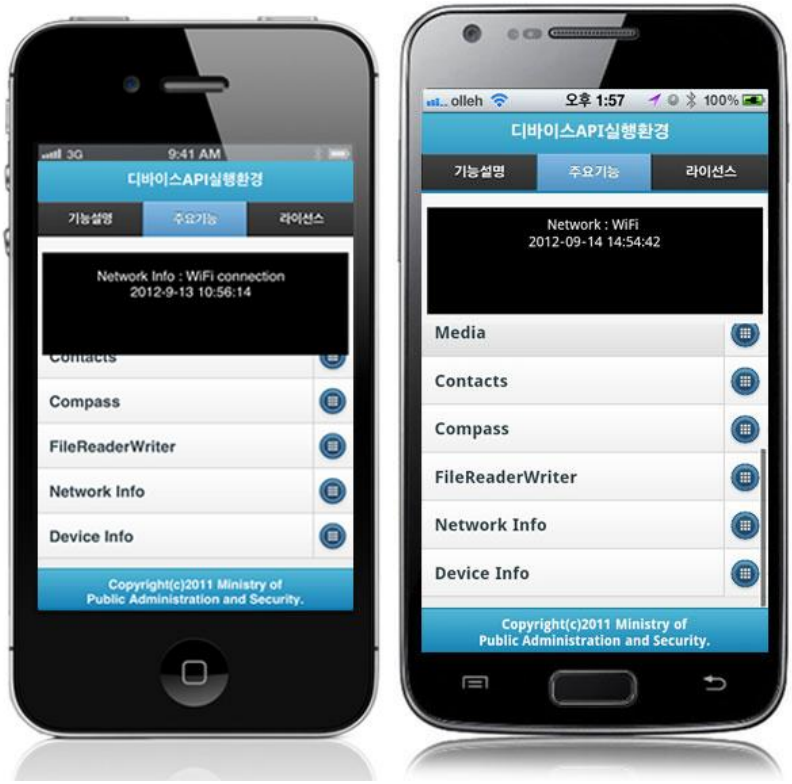
Related Codes

Checks network condition

```
function fn_egov_check_network()
{
	var networkState = navigator.network.connection.type;
	var states = {};
	states[Connection.UNKNOWN]  = 'Unknown connection';
	states[Connection.ETHERNET] = 'Ethernet connection';
	states[Connection.WIFI]     = 'WiFi connection';
	states[Connection.CELL_2G]  = 'Cell 3G connection';
	states[Connection.CELL_3G]  = 'Cell 3G connection';
	states[Connection.CELL_4G]  = 'Cell 4G connection';
	states[Connection.NONE]     = 'No network connection';

	var html = "<span>Network Info : " + states[networkState] + "<BR />" +
fn_egov_get_nowTime()+"</span>";
	fn_egov_display_deviceAPIInfoMain("NETWORK",html);
	fn_egov_insert_table("NETWORK", states[networkState]);
}
```

Related Screen and Implementation Manual



1. Updates device network information

Updates device meta information

Related Codes

Updates device information

```
function fn_egov_get_deviceInfo()
{
	var html = "NAME : " + device.name + "<BR/>cordovaVersion : " + device.cordova
			+ "<BR/>platform : " + device.platform + "<BR/>uuid : " + device.uuid
			+ "<BR/>version : " + device.version + "<BR />" + fn_egov_get_nowTime();

	fn_egov_display_deviceAPIInfoMain("DEVICE",html);
	fn_egov_insert_table("DEVICE",device);
}
```

Related Screen and Implementation Manual

1. Updates device meta information

How to Use

Installations

Use the developmental environment tools for installations of the runtime environments for eGovFramework Device API. Refer to the following for more information:

- Runtime example installation link

Troubleshooting Guide

- Troubleshooting for device application: Use console.log of PhoneGap Framework to verify errors and debugging. Debug codes in console.log are available in JavaScript syntaxes that you can use in both Eclipse and Xcode.

See the following for how to code console.log:

```
console.log("[DeviceAPI Guide] fn_egov_delete_fileInfo : Completed");
```

When the debugging code is executed, check out the following console message appears:

```
)
2012-09-14 09:45:20.801 DeviceAPIGuide_iOS_V1.9[5765:707]
AppDelegate forcing status bar to: 1 from: 5
2012-09-14 09:45:22.183 DeviceAPIGuide_iOS_V1.9[5765:707]
[INFO] [DeviceAPI Guide] Debugging Message :
```