# PhoneGap APIs

## Làm việc với Device, Network, và Notifications

### Lấy thông tin từ thiết bị

PhoneGap có một *runtime object*được gọi là đối tượng *device,* để lưu giữ những thông tin hữu ích về thiết bị. Sau đây là những thuộc tính của đối tược *device*:

* device.name: tên của thiết bị (ví dụ như *my iPhone*)
* device.phonegap: phiên bản của PhoneGap
* device.platform: loại của thiết bị (ví dụ như *iPhone*)
* device.uuid: số UUID của thiết bị
* device.Version: phiên bản của hệ điều hành đang chạy

Một lưu ý quan trọng rằng, *device* được gán thành đối tượng *windows* (windows object), do vậy nó có phạm vi toàn cục. Để làm rõ, ta có khai báo hai biến khác nhau như sau và chúng đều tham chiếu tới *device*:

// Cả hai biến đều tham chiếu tới device như nhau

var myPhoneName = window.device.name;

var myPhoneName = device.name;

Trong mục này, ta sẽ học cách lấy các thông tin như: tên thiết bị, phiên bản của PhoneGap, số UUID của thiết bị và phiên bản của hệ điều hành đang dùng.

<ChuwxNTC.On-going>

## Accelerometer

### Accelerometer là gì?

Accelerometer (có nghĩa là gia tốc kế) là một API cho phép PhoneGap theo dỏi sự thay đổi các hướng của thiết bị trong không gian ba chiều (theo ba hướng X, Y và Z).

Ví dụ, với thiết bị nằm song song trên mặt bàn, mặt thiết bị hướng lên trên thì kết quả sẽ là X:0; Y:0; Z:10 và nếu mặt thiết bị hướng xuống thì X:0; Y:0; Z:-10. Khi thiết bị được dựng đứng đúng chiều X:0; Y:10; Z:0 và ngược chiều X:0; Y:-10; Z:0. Khi thiết bị nằm quay về bên trái X:10; Y:0; Z:0, bên phải X:-10; Y:0; Z:0. Như vậy, từ đó ta có thể hình dung sự thay đổi hướng X, Y và Z của thiết bị phụ thuộc vào sự thay đổi chiều của thiết bị khi xét đến như là một khối hộp 6 mặt. Nên nhớ rằng, khả năng của gia tốc kế có thể là xác định chuyển động, độ quay và gia tốc, giúp bạn trong việc tạo ra những ứng dụng sáng tạo và thú vị.

### Sử dụng đối tượng Acceleration

Đối tượng Acceleration là một đối tượng chỉ có thể đọc và nó chứa dữ liệu gia tốc kế được chụp tại một thời điểm nhất định trong một thời điểm. Nó gồm các thuộc tính:

* X – đại lượng chuyển động theo trục X.
* Y – đại lượng chuyển động theo trục Y.
* Z – đại lượng chuyển động theo trục Z.
* Timestamp – mốc thời gian lúc tạo ra, được trình bày trong mili giây.

Đối tượng này được tạo ra và cư trú trong PhoneGap, cách xác định giá trị được minh họa qua ví dụ sau:

navigator.accelerometer.getCurrentAcceleration(onSuccess, onError);

// Phương thức trên khai báo và lấy đối tượng Acceleration

// Sau đây ta sẽ lấy dữ liệu

function onSuccess(acceleration) {

alert(‘Acceleration X: ‘ + acceleration.x + ‘\n’ +

‘Acceleration Y: ‘ + acceleration.y + ‘\n’ +

‘Acceleration Z: ‘ + acceleration.z + ‘\n’ +

‘Timestamp: ‘ + acceleration.timestamp + ‘\n’);

};

function onError() {

alert(‘Sorry! Error!’);

};

### Cách sử dụng các phương thức trong Acceleration API

**Phương thức *getCurrentAcceleration***

Để lấy các giá trị của gia tốc kế X, Y và Z, ta sử dụng accelerometer .getCurrentAcceleration cụ thể như sau:

navigator.accelerometer.getCurrentAcceleration(accelerometerSuccess,

accelerometerError);

Dữ liệu sẽ trả thông qua hàm accelerometerSuccess:

function onSuccess(acceleration) {

alert(‘Acceleration X: ‘ + acceleration.x + ‘\n’ +

‘Acceleration Y: ‘ + acceleration.y + ‘\n’ +

‘Acceleration Z: ‘ + acceleration.z + ‘\n’ +

‘Timestamp: ‘ + acceleration.timestamp + ‘\n’);

};

function onError() {

alert(‘ooooops!’);

};

navigator.accelerometer.getCurrentAcceleration(onSuccess, onError);

Chú ý: trong các thiết bị iOS, phương thức này không hoạt động như ta nghĩ. Nó sẽ lấy và báo cáo giá trị lại của lần cuối mà Accelerometer API của PhoneGap gọi. Do vậy, trong iOS ta phải dùng phương thức watchAcceleration thay thế.

**Phương thức *watchAcceleration***

Phương thức watchAcceleration cho phép chúng ta lấy giá trị dữ liệu của gia tốc kế tại một khoảng thời gian cụ thể. Chúng ta sẽ thiết lập một biến chứa tần suất cập nhật và sẽ được một chu kỳ theo dỏi trong khoảng thời gian thiết lập đó. Theo dỏi ví dụ dưới đây:

function onSuccess(acceleration) {

alert(‘Acceleration X: ‘ + acceleration.x + ‘\n’ +

‘Acceleration Y: ‘ + acceleration.y + ‘\n’ +

‘Acceleration Z: ‘ + acceleration.z + ‘\n’ +

‘Timestamp: ‘ + acceleration.timestamp + ‘\n’);

};

function onError() {

alert(‘onError!’);

};

var options = { frequency: 1000 }; // Update every second

var watchID = navigator.accelerometer.watchAcceleration(onSuccess, onError, options);

Đoạn mã trên sẽ thông báo theo chu kỳ một giây.

Chú ý: trong các thiết bị iOS, gia tốc kết được quan tâm đặt biệt. Do đó, PhoneGap sẽ bị giới hạn mức đăng ký theo dỏi bởi chu kỳ từ 40 mili giây đến 1000 mili giây. Và khi bạn sử dụng một khoảng thời gian lớn hơn 1000 mili giây thì PhoneGap sử dụng trong iOS sẽ tự động cập nhật thành công trong 1000 mili giây.

**Phương thức *clearWatch***

Để dừng việc xem thay đỏi các giá trị của gia tốc kế được đăng ký ở phương thức watchAcceleration, ta dùng clearWatch như sau:

navigator.accelerometer.clearWatch(watchID);

Thông thường ta gắn đoạn mã này trong sự kiện của một nút lệnh Button hoặc cũng có thể tự động qua đoạn mã Javascript trong một khoảng thời gian chờ định sẳn.

## Camera

PhoneGap Camera API cung cấp cho một ứng dụng khả năng làm việc với một hình ảnh, khả năng chụp ảnh thông qua camera hay nhận tệp ảnh từ kho ảnh trong thiết bị. Khi lấy về một ảnh, thì API có thể trả về một URL chỉ tới ảnh trên thiết bị hoặc có thể là một chuổi base64-encoded biểu diễn nội dung bức ảnh.

API cung cấp một phương thức đơn là navigator.camera.getPicture, dùng để lấy về một ảnh hoặc đối tượng cameraOptions, dùng để định nghĩa các tham số như ảnh nhận về như thế nào, định dạng,…

Các ứng dụng cũng có thể sử dụng PhoneGap Capture API để chụp ảnh. Tuy nhiên, PhoneGap Capture API và Camera API là khác nhau. Do vậy, cần phải đánh giá trước khi sử dụng các nào cho phù hợp với mục đích ứng dụng.

### Truy cập sử dụng ảnh

Để nhận truy xuất một ảnh, ứng dụng nên thực thi các chức năng sau:

navigator.camera.getPicture(onCameraSuccess, onCameraError, cameraOptions);

Giống như các thư viện API khác của PhoneGap, hàm getPicture đòi hỏi phải đưa vào hai hàm để thực thi khi lệnh chạy thành công hay thất bại. Hàm onCameraError gọi khi người dùng hủy bỏ lệnh hoặc có bất cứ lỗi nào xãy ra trong tiến trình. Hàm onCameraSuccess thực thi với ảnh được chụp cùng những tham số cài đặt trong hàm cameraOptions.

### Cấu hình tùy chỉnh Camera

PhoneGap Camera API cũng hỗ trợ những cài đặt cho Camera để tiến hành chụp ảnh theo ý muốn như sau:

* quality: chất lượng của ảnh.
* destinationType
* sourceType
* allowEdit: cho phép chỉnh sửa.
* encodingType: loại mã hóa tệp ảnh.
* targetWidth: kích thước chiều rộng.
* targetHeight: kích thước chiều cao.
* mediaType

Tuy nhiên, một số cấu hình không hỗ trợ trên một vài nền tảng nhất định. Ví dụ sau đây cho biết các cấu hình có thể sử dụng trên mọi nền tảng:

var cameraOptions = { quality : 75,

sourceType : Camera.PictureSourceType.CAMERA,

destinationType : Camera.DestinationType.FILE\_URI,

allowEdit : true,

encodingType: Camera.EncodingType.JPEG,

targetWidth: 1024,

targetHeight: 768 };

## Capture

PhoneGap Capture API cho phép ứng dụng ghi lại âm thanh, video và hình ảnh. PhoneGap triển khai Capture API nền tảng trên W3C Media Capture API (www.w3.org/TR/media-capture-api). Và vì nhiều lý do nào đó, mà đội phát triển PhoneGap không thể triển khai hoàn toàn các chuẩn này. Do vậy, khi sử dụng PhoneGap Capture API có thể có vài tính năng không làm việc hoặc không được cung cấp.

### Capture video

Để ghi video trên thiết bị, ta sử dụng phương thức device.capture.captureVideo cụ thể như sau:

navigator.device.capture.captureVideo(onSuccess, onError, options);

Hai hàm onSuccess, onError sẽ được gọi khi thực thi thành công hoặc thất bại và options để tùy chỉnh cài đặc camera.

***Sử dụng các tham số tùy chỉnh:***

Tham số options ở trên gồm:

* limit – số lượng đoạn video mà người dùng có thể ghi tại một thời điểm (mặc định là 1).
* duration – thời gian tối đa ghi video (tính bằng giây).
* mode – chế độ ghi video (video/quicktime, video/3gpp, …)

Ví dụ, để ghi video trong thời gian 30 giây, ta có thể làm như sau:

var options = { duration: 30 };

navigator.device.capture.captureVideo(onSuccess, onError, options);

Để ghi video lấy 4 clip:

// start video capture – grab 4 clips

navigator.device.capture.captureVideo(onSuccess,onError, {limit:4});

Hàm thực thi sau khi ghi video thành công:

function **onSuccess**(media\_files) {

var i, len;

for (i = 0, len = media\_files.length; i < len; i += 1) {

processFiles(media\_files[i]);

}

}

function processFiles(media\_file){

//do something interesting here

}

Một số lưu ý khi sử dụng:

* Trong các thiết bị Android và BlackBerry:
  + Tham số duration không hỗ trợ, nó được thiết đặt tự động.
  + Tham số mode không hỗ trợ.
* Trên nền tảng iOS:
  + Tham số limit không hỗ trợ.
  + Tham số duration không hỗ trợ, không thể thiết đặt giá trị này.
  + Tham số mode không hỗ trợ, định dạng mặc định là MOV (video/quicktime).

### Capture audio

Tương tự như ghi video, ta thay vào đó là phương thức navigator.device

.capture.captureAudio() như sau:

navigator.device.capture.captureAudio(onSuccess, onError, options);

Và đi cùng với các tham số options được định nghĩa như trên.

### Capture images

Ở Camera API, chúng tôi đã trình bày cách chụp ảnh sử dụng ứng dụng camear mặc định trên thiết bị, hoặc chọn ảnh từ thư viện ảnh, đường dẫn. Capture API sẽ cho phép chúng ta tạo một chương trình để chụp ảnh, xử lý và lưu chúng mà không cần phải thông qua ứng dụng có sẳn đó.

Để tiến hành chụp một bức ảnh, ta chi đơn giản gọi phương thức navigator.device.capture.captureImage() như sau:

navigator.device.capture.captureImage(onSuccess, onError, options);

Ta gọi hàm onSuccess khi đối đã chụp ảnh thành công, như sau:

function onSuccess(media\_files) {

var i, len;

for (i = 0, len = media\_files.length; i < len; i += 1) {

processFiles(media\_files[i]);

}

}

function processFiles(media\_file){

//do something interesting here

}

// start video capture – grab 2 clips

navigator.device.capture.captureImage(onSuccess,onError, {limit:2});

Tùy chỉnh tham số options:

* limit – số lượng ảnh chụp.
* mode – loại ảnh chụp (chẳn hạn như image/jpeg).

Ghi chú quan trọng khi sử dụng Capture images:

* Trên thiết bị Android, BlackBerry thì tham số mode không hỗ trợ, mặc định là image/jpeg.
* Trên nền tảng iPhone:
  + Tham số limit không hỗ trợ, ta chỉ chụp được một ảnh trên một lần.
  + Tham số mode không hỗ trợ, mặc định là image/jpeg.

## Media

Media API cung cấp cho ứng dụng khả năng ghi âm, chơi các tập tin âm thanh. Các chức năng trong Media API chồng chéo và nhiều hạn chế so với Capture API. Đã có những cảnh báo đến các nhà phát triển, Media API không thực sự tuân thủ đặc điểm kỷ thuật theo chuẩn W3C và tương lai hướng phát triển sẽ xoay quay theo Capture API. Tuy nhiên, nó vẫn thực sự còn hữu ích khi có một số tính năng mà Capture API không hỗ trợ. Media API bị giới hạn hỗ trợ trên Android, iOS và Windows Phone. Điều đó có nghĩa khi chúng ta sự dụng trên BlackBerry và các nền tảng khác nó sẽ không hoạt động.

### On-going

## File

File API trong PhoneGap được dựa trên File API qui định trong W3C. Một API dùng để đọc, viết và điều hướng các hệ thống phân cấp hệ thống tập tin.

.. to be continue…

## Storage

Storage API được dựa trên đặc tả kỹ thuật W3C Web SQL Database[[1]](#footnote-1) và W3C Web Storage API[[2]](#footnote-2). Một vài nền tảng di động đã hiện thực đặc tả này, do đó có thể xây dựng thay chúng thay vì phải dùng Cordova. Với những thiết bị nền tảng chưa được hỗ trợ, Việc hiện thực Cordova cần được tương thích với các đặc điểm kỹ thuật của W3C.

Web Database (về bản chất là một hệ thống của SQLite3) rất phù hợp với các lưu trữ dữ liệu phức tạp.

### Lcal Storage

Local Storage cho phép một ứng dụng lưu dữ liệu lại với cặp key/value, giá trị này được lưu giữ với ứng dụng. W3C Web Storage API bao gồm cả hai phiên bản Session và Local Storage. Điểm khác nhau giữa Session Storage và Local Storage là các đối tượng Session Storage sẽ được xóa khi ta đóng trình duyệt còn Local Storage thì không.

Ví dụ để trả về *key* của cặp dữ liệu ở vị trí thứ 5:

*var myKeyName = window.localStorage.key(5);*

Để ghi một lưu trữ (gồm khóa và giá trị) vào trong bảng Local Storeage thì ta sử dụng mã như sau:

*window.localStorage.setItem("key\_name", value);*

Để lấy một giá trị trong bảng sử dụng mã như sau:

*window.localStorage.getItem("key\_name");*

Để xóa một lưu trữ trong bảng Local Storage:

*window.localStorage.removeItem("key\_name");*

Để xóa toàn bộ:

*window.localStorage.clear();*

### SQL Database

#### Mở một Database

Trong SQLite , phương thức mở database sẽ mở một database có sẳn hoặc tự động tạo mới nếu nó chưa tồn tai. Để mở một database ta dùng phương thức *window.openDatabase()* như sau:

var myDB = window.openDatabase(name, version, displayname,size);

Ý nghĩa các thông số:

* name – Tên của database
* version – Phiên bản (hai database trùng tên nhưng khác phiên bản được xem là khác nhau)
* displayname – Mô tả database
* size – Kích thức của database (tính bằng đơn vị byte)

Ngoài ra việc mở một database còn có thêm một thông số tùy chọn *databasecallback* – là phương thức *callback* được thực thi sau khi gọi database được mở/tạo.

Ví dụ: *var myDB = window.openDatabase(“photos”, “1.0”, “Photos DB”,10000000);*

Lệnh này sẽ tạo một database tên photos, phiên bản 1.0 với kích thước là 1 MB để trả về myDB (sẽ được sử dụng tiếp ở ví dụ dưới). Lệnh mở/tạo database thường được đính kèm vào hàm *onDeviceReady()* như sau:

*document.addEventListener(“deviceready”, onDeviceReady, false);*

*function onDeviceReady() {*

*var myDB = window.openDatabase(“photos”, “1.0”, “Photos DB”, 1000000);*

*}*

#### Chạy một truy vấn SQL

Định nghĩa tổng quát của đối tượng SQLTransaction – đối tượng này không chỉ sử dụng một cách độc lập mà phải được nhúng trong các phương thức như: transaction(), readTransaction(), changeVersion().

Khi chạy một chuỗi truy vấn SQL, chúng ta phải sử dụng nó trong một đối tượng SQLTransaction. Đối tượng này bao hàm các phương thức cho phép ta thực thi các câu lệnh SQL dựa vào một database đã được mở từ trước.

Đây là một dạng cơ bản của phương thức *executeSql()*:

*myDB.executeSQL(‘SELECT \* FROM table1’);*

Lưu ý là myDB được sử dụng vì nó đã được khai báo ở ví dụ trên.

Lệnh sau dùng để xóa một bảng:

*myDB.executeSql(‘DROP TABLE IF EXISTS table1)’);*

Lệnh sau dùng để tạo một bảng:

*myDB.executeSql(‘CREATE TABLE IF NOT EXISTS table1 (id unique, firstname varchar, lastname varchar)’);*

Thêm dữ liệu vào bảng:

*myDB.executeSql(‘INSERT INTO TABLE (id, firstname, lastname) VALUES (1, “Thomas “, “Myer “)’);*

Lệnh sau dùng để xóa dữ liệu:

*myDB.executeSql(‘DELETE FROM TABLE where id=1’);*

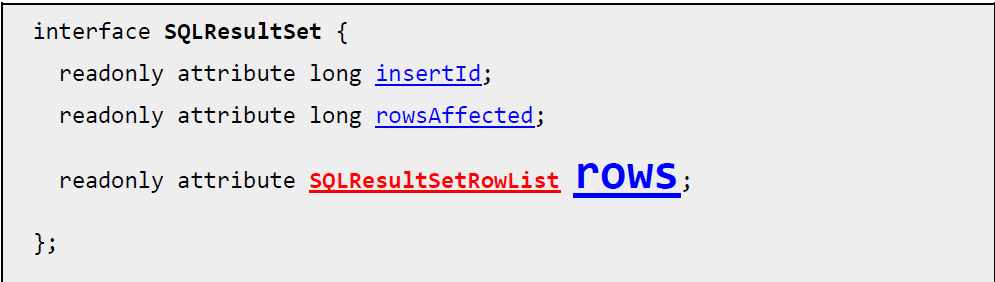
#### Xem kết quả SQLResultSet

Phương thức transaction cung cấp hàm gọi lại success khi thực hiện thành công. Phương thức executeSql() cũng hỗ trợ hai hàm gọi lại đó là success và error. Với hàm success sẽ được dùng để xem kết quả truy vấn.

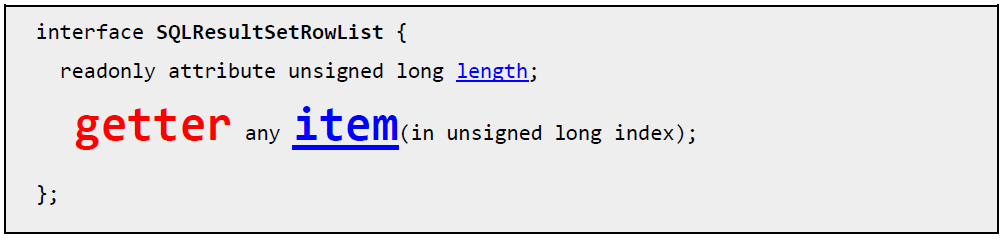
Đối tượng SQLResultSet chưa ba thuộc tính sau:

* insertId – Được xem là số thứ tự hàng mà câu truy vấn SQL của đối tượng SQLResultSet được thêm vào trong cơ sở dữ liệu (chỉ được áp dụng nếu ta chạy câu lệnh insert).
* rowAffected – Đây là số dòng được thay đỗi bởi câu truy vấn SQL (kết quả sẽ trả về 0 nếu là câu truy vấn select).
* rows – Là danh sách các kết quả được lưu dưới dạng đối tượng SQLResultSetRowList.

Định nghĩa chung cho đối tượng SQLResultSet:



Định nghia chung cho đối tượng SQLResultSetRowList:



#### Kiểm soát lỗi

Nếu có một lỗi xãy ra thì PhoneGap sẽ văng ra một đối tượng SQLError. Đối tượng SQLError chứa hai thuộc tính sau:

* code – Đây là một mã lỗi được định nghĩa trước.
* message – Thông điệp mô tả lỗi.

Mã lỗi bao gồi các loại sau:

* SQLError.UNKNOWN\_ERR
* SQLError.DATABASE\_ERR
* SQLError.VERSION\_ERR
* SQLError.TOO\_LARGE\_ERR
* SQLError.QUOTA\_ERR
* SQLError.SYNTAX\_ERR
* SQLError.CONSTRAINT\_ERR
* SQLError.TIMEOUT\_ERR

1. W3C Web SQL Database – Tài liệu thao khảo tại: <http://dev.w3.org/html5/webdatabase> [↑](#footnote-ref-1)
2. W3C Web Storage API – Tài liệu thao khảo tại: <http://dev.w3.org/html5/webstorage> [↑](#footnote-ref-2)