

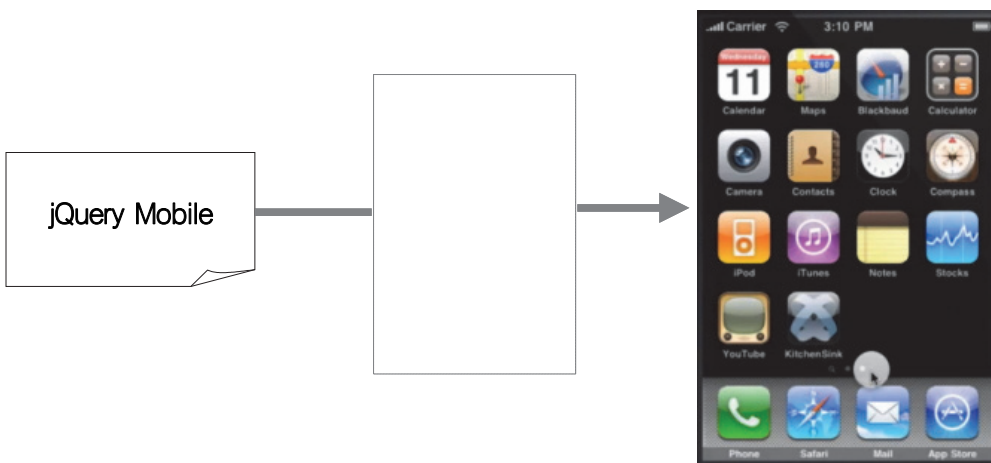
이 장에서는 jQuery Mobile로 웹앱을 완성하는 기술을 훑어볼 것입니다. 브라우저 기반의 모바일 웹 사이트만을 만들 것이라면 이 장은 보지 않아도 됩니다. 이 장에서는 폰갭(PhoneGap)을 기반으로 웹앱을 만드는 과정을 소개한 후, 폰갭의 데모 소스를 jQuery Mobile의 스마트 디자인 기술로 포장한 결과를 보여줍니다. 이를 통해 웹앱을 생성하는 방법과 jQuery Mobile의 전체적인 기능을 살펴보게 됩니다. 단순히 설치와 웹앱 테스트 과정을 익히는 것이므로 어렵지 않습니다.

jQuery Mobile은 "모바일 웹(Mobile Web)" 솔루션으로 활용될 수도 있고, "웹앱(Web App)" 솔루션으로 활용될 수도 있습니다. jQuery는 "jQuery Mobile"이라는 이름으로 HTML5를 기반으로 하는 화면(User Interface) 관련 솔루션을 제공하고 있습니다. jQuery Mobile은 웹앱 솔루션이 완성되기 전부터 모바일 웹의 웹 화면에 맞는 일명 스마트 디자인 솔루션을 제공하고 있었습니다.

본서를 집필하는 현재 시점까지 제시된 대부분의 jQuery Mobile 사용법들은 모바일 웹 브라우저에 표시되는 "모바일 웹"에 활용한 것이지, 스마트폰에서 실행되는 "웹앱"에 활용한 것은 아닙니다. jQuery Mobile만으로는 기존의 앱과 같은 웹앱을 만들 수 없습니다. 이는 "jQuery Mobile"의 경쟁 상대인 "센차터치(Sencha Touch)"도 마찬가지입니다. 아래 그림은 모바일 웹에 활용한 경우입니다.



하드웨어 제어까지 가능한 완전한 웹앱을 만들기 위해서는 폰갭과 같은 네이티브 브리지 솔루션이 필요합니다. 앱의 화면은 jQuery Mobile로 해결되지만, 이를 앱으로 완성시키는 네이티브 프로그램에 포팅하지 못하면 웹앱이 될 수 없습니다. jQuery Mobile은 폰갭을 만나 비로소 웹앱이라는 혁신적인 솔루션을 만들어 냈습니다. 아래 그림은 웹앱에 활용한 경우를 보여줍니다.



한편 폰갭은 웹앱을 만들 수 있지만, 화면 디자인에 대한 솔루션을 제공하지는 않기 때문에 스마트폰별로 특성을 살린 디자인을 하려면 일일이 수동으로 작업해야 하는 문제를 안고 있었습니다. 그런 면에서 jQuery Mobile과 폰갭의 만남은 절묘한 조화가 아닐 수 없습니다. 앱이 탄생한지 수년이 지났지만 그동안 앱을 만들기 위해서는 iOS, Android, Windows Phone 등과 같은 전문적인 네이티브 프로그램을 배워야 했습니다. 하지만 이제 “웹 기술을 활용하는 웹앱”이 등장함으로써 웹 디자이너나 웹 관련 일을 하던 사람들에게는 웹앱이 광활한 사막의 오아시스와 같은 해법이 되었습니다.

간단한 자바스크립트만으로도 네이티브 프로그램의 기능을 구현할 수 있는 폰갭으로 인해 프로그래밍에 대한 노력은 10분의 1 이상으로 줄었고, 오래 전부터 웹 페이지 디자인 솔루션으로 잘 알려진 jQuery가 jQuery Mobile을 제공함으로써 많은 시간을 소모해야 했던 스마트앱 디자인 작업 역시 10분의 1 이상으로 줄었습니다. 즉, 스마트폰의 하드웨어 제어 기능은 폰갭으로 해결하고, 스마트앱 디자인은 jQuery Mobile로 해결할 수 있게 된 것입니다. 특히 폰갭과 jQuery Mobile은 모두 아이폰, 안드로이드, 윈도우폰에서 거의 완벽하게 지원하기 때문에 진정한 “하이브리드 웹앱”을 실현했다고 할 수 있습니다.

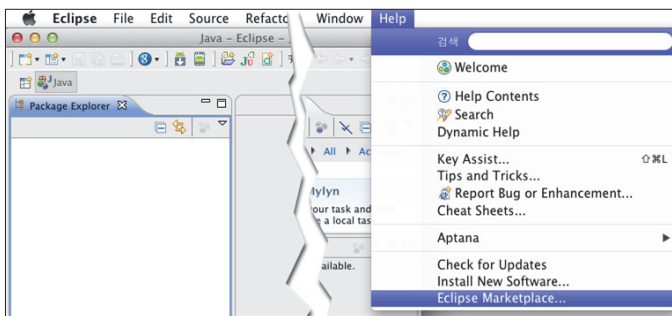
3.1 안드로이드용 웹앱 프로젝트

안드로이드에서 웹앱을 만들려면 수동으로 터미널을 이용하여 제작할 수도 있지만, 많은 시간과 노력이 필요합니다. 이를 편리하게 개발할 수 있도록 하는 개발 툴이 이클립스입니다. 이클립스는 이클립스 마켓플레이스를 통해 많은 개발 도구를 플러그인 방식으로 제공하고 있습니다. 폰갭 역시 이클립스 마켓플레이스에서 “안드로이드용 폰갭” 개발 도구를 찾아 설치할 수 있습니다.

이클립스 폰갭 플러그인 설치

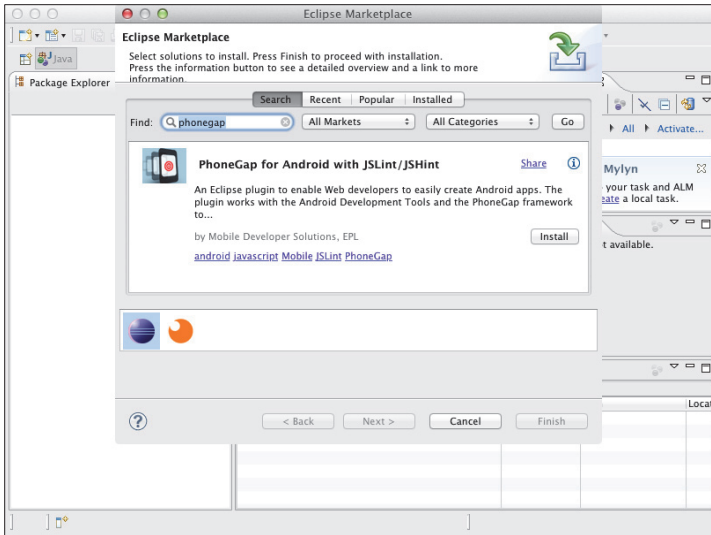
스텝 1

“이클립스 > Help > Eclipse Marketplace...” 메뉴를 실행하면 “이클립스 마켓플레이스”에 연결할 수 있습니다.



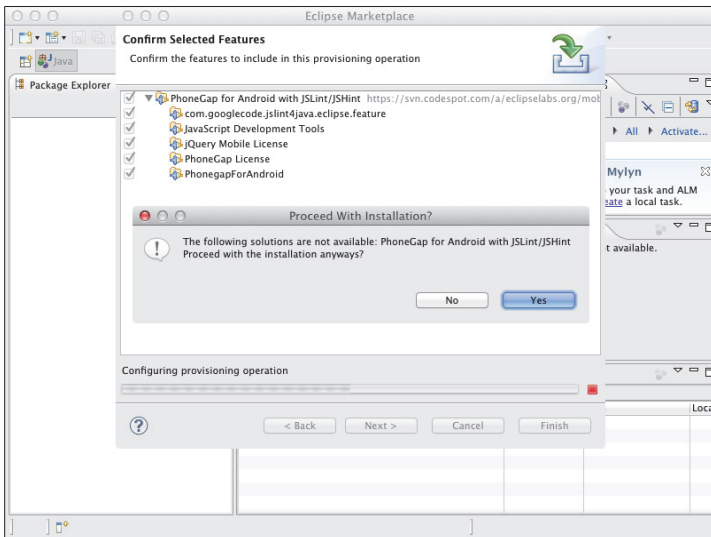
스텝 2

“Eclipse Marketplace” 창에서 “phonegap”으로 검색하여 “PhoneGap for Android ...” 플러그인을 찾습니다. 이 플러그인은 폰갭, jQuery Mobile, 센차터치 프레임워크를 모두 포함하고 있어 매우 실용적인 안드로이드 웹앱 프로젝트 생성 마법사를 제공합니다. 또한 “폰갭”, “jQuery Mobile”, “센차터치” 프레임워크에 대해 버전을 선택하여 프로젝트를 생성할 수 있는 유연성을 제공합니다. “Install” 버튼을 클릭하면 이 플러그인을 설치할 수 있습니다.



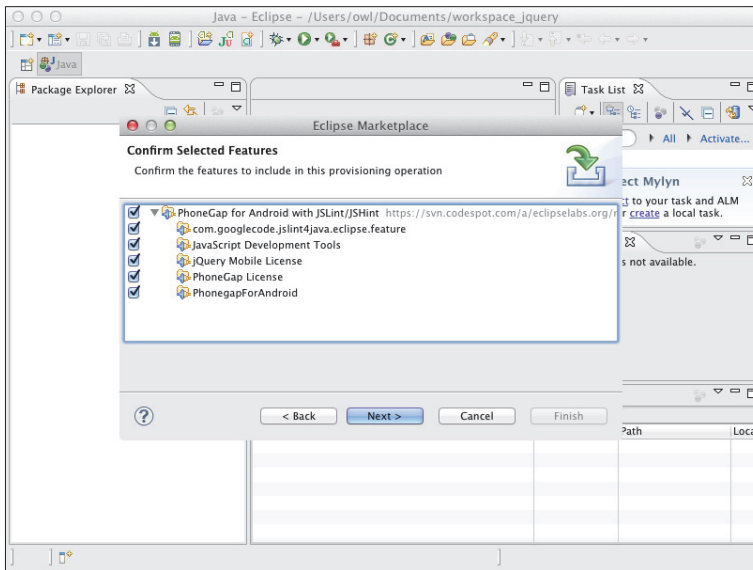
스텝 3

폰갭 플러그인의 설치가 시작되면 확인 대화상자가 나타날 수 있습니다. "Yes" 버튼을 클릭합니다.



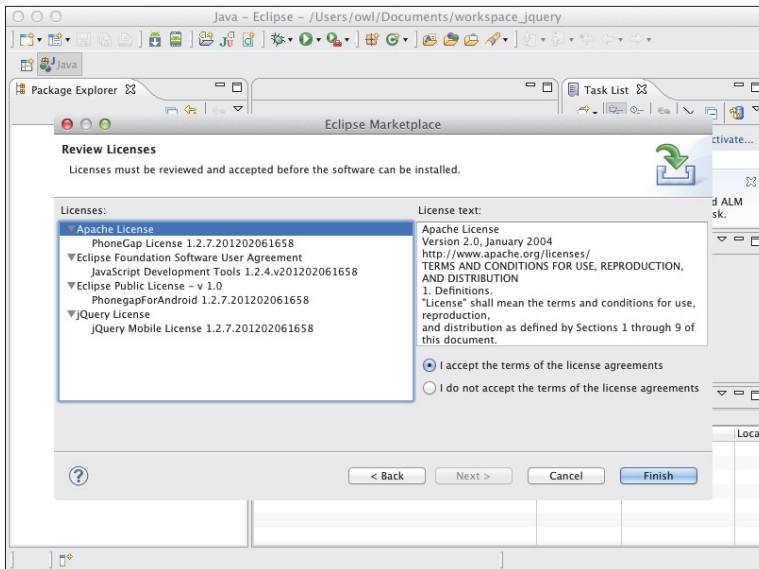
스텝 4

설치할 플러그인을 확인하고 "Next" 버튼을 클릭합니다.



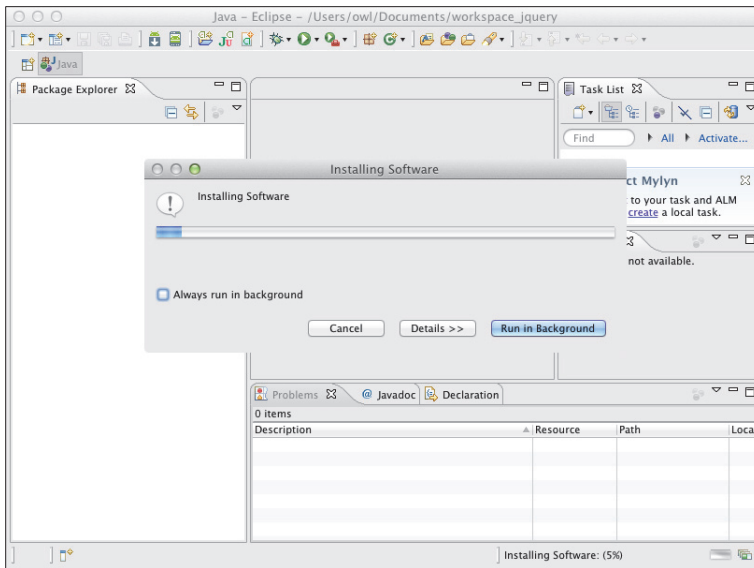
스텝 5

설치할 플러그인에 대한 라이선스에 동의하고 "Finish" 버튼을 클릭합니다.



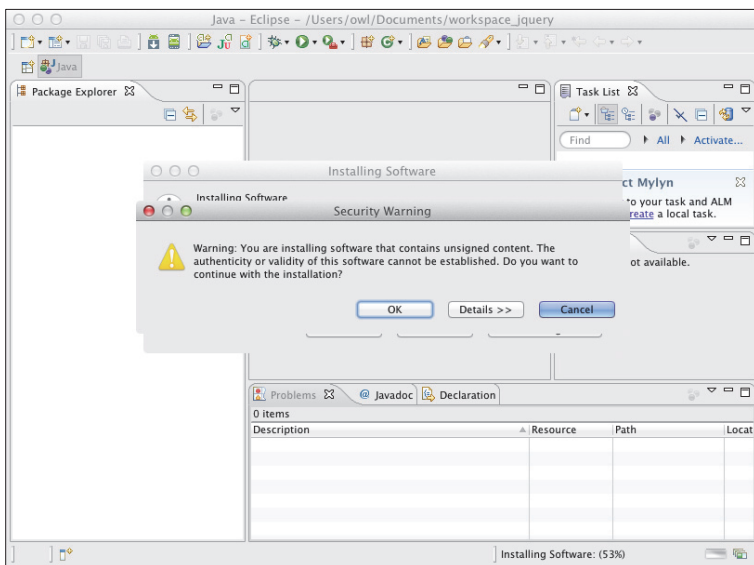
스텝 6

설치 과정이 시작되면 그림과 같이 나타납니다.



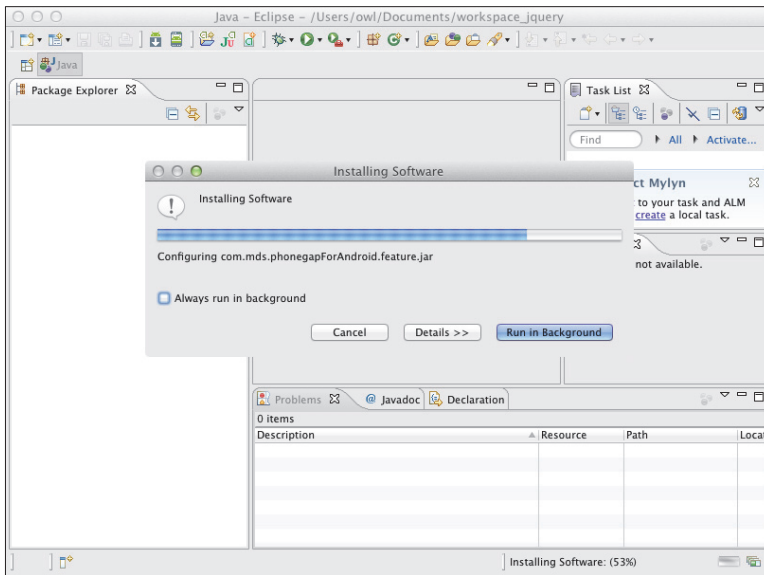
스텝 7

설치 과정에서 그림과 같이 보안에 대한 경고창이 나타날 수 있습니다. "OK" 버튼을 클릭하고 설치를 계속 진행합니다.



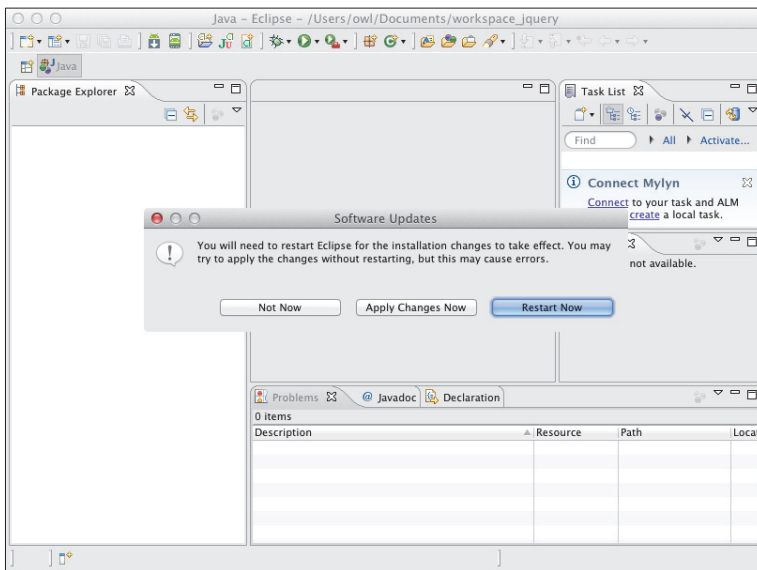
스텝 8

설치 과정이 계속 진행됩니다.



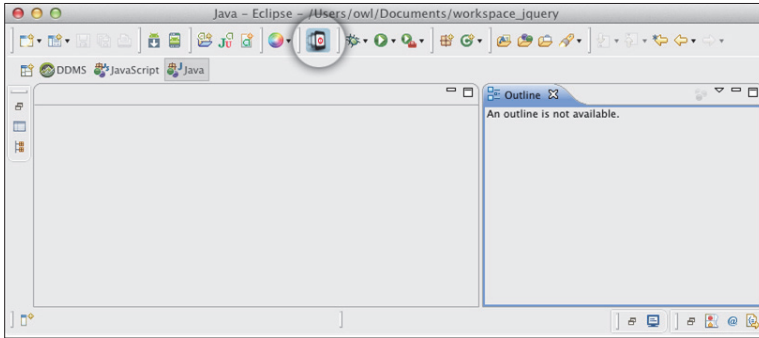
스텝 9

플러그인 설치를 완료하면 이 플러그인을 이클립스에 업데이트하기 위해 그림과 같은 대화상자가 나타납니다. "Restart Now" 버튼을 클릭하여 이클립스를 재시동합니다.



스텝 10

안드로이드용 폰갭 플러그인이 이클립스에 성공적으로 설치되었다면 그림과 같이 "폰갭 프로젝트 생성 아이콘"이 나타날 것입니다.

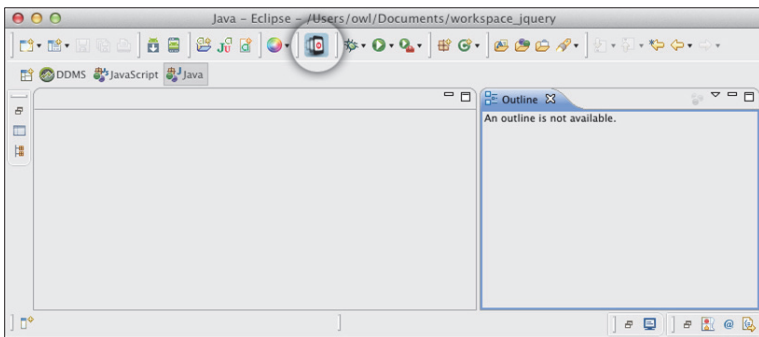


폰갭 프로젝트 생성

위에서 설치한 안드로이드용 폰갭 플러그인은 안드로이드용 폰갭 프로젝트 생성 마법사를 제공하는 것이 주요 기능입니다. 안드로이드용 폰갭 프로젝트를 생성하는 방법은 다음과 같이 "아이콘 버튼"을 이용하는 방법과 "메뉴"를 이용하는 방법이 있습니다.

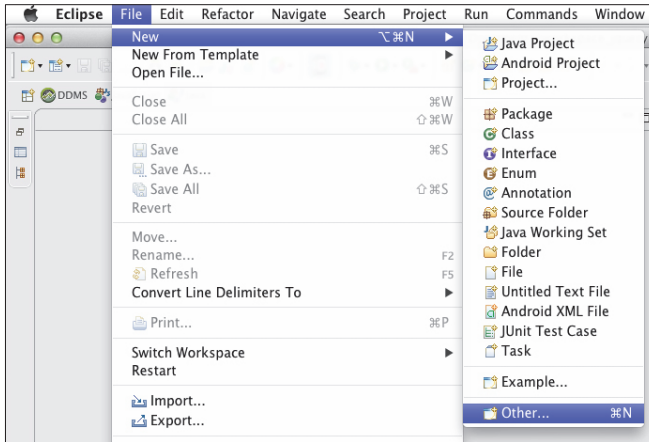
스텝 1

그림과 같이 "안드로이드용 폰갭 프로젝트 생성 아이콘 버튼"을 클릭하면 안드로이드용 폰갭 프로젝트 생성 마법사가 나타납니다.



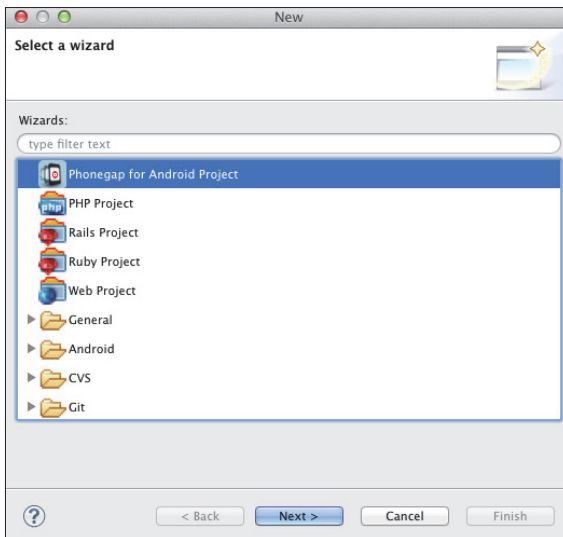
스텝 2

또 다른 방법으로 "File > New > Other..." 메뉴를 실행하여 안드로이드용 폰갭 프로젝트 생성 마법사 창을 호출할 수도 있습니다.



스텝 3

"File > New > Other..." 메뉴는 그림과 같이 이클립스에서 만들 수 있는 각종 프로젝트를 모두 탐색하고 선택할 수 있습니다. "Phonegap for Android Project"를 선택하고 "Next" 버튼을 클릭하면 "안드로이드용 폰갭 프로젝트 생성 마법사" 창이 나타납니다.



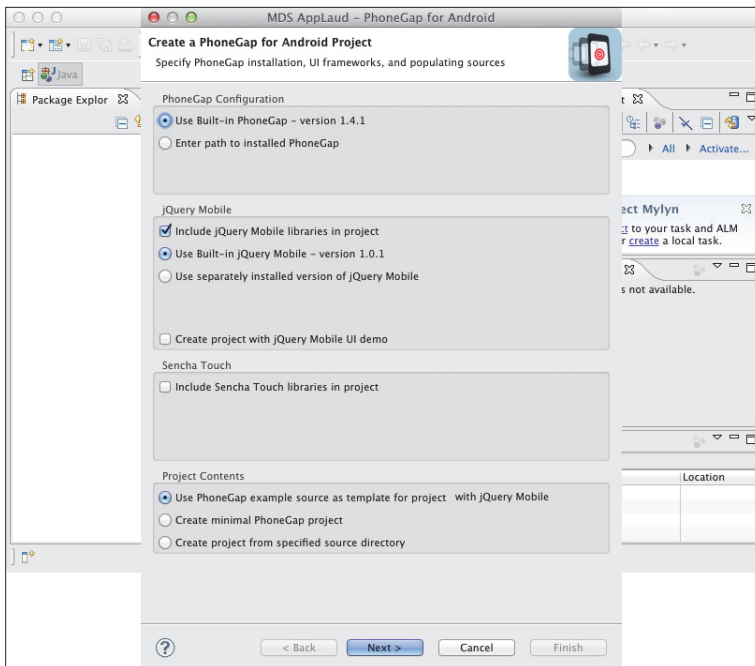
jQuery Mobile 라이브러리 포함 옵션

스텝 4

"MDS AppLaud - PhoneGap for Android" 마법사는 그림과 같이 "폰갭", "jQuery Mobile", "센차터치", "프로젝트 콘텐츠" 등 4개의 설정 영역으로 나뉩니다.

"폰갭", "프로젝트 콘텐츠" 영역은 필수 설정 사항이고, "jQuery Mobile", "센차터치"는 둘 중 하나만 선택해야 합니다. 그림에서는 "폰갭"은 플러그인에 기본으로 탑재된 폰갭 버전인 "1.4.1"을 선택하고 있고, "jQuery Mobile"도 플러그인에 기본으로 탑재된 "1.0.1" 버전을 선택하고 있습니다. "jQuery Mobile"을 스마트 디자인용으로 선택했기 때문에 "센차터치"는 선택할 수 없습니다. "프로젝트 콘텐츠" 영역에서는 이 프로젝트에 탑재한 웹 소스 샘플을 설정합니다. "jQuery Mobile"의 적용 사례와 함께 "폰갭"의 기능을 전반적으로 둘러보기 위해 "jQuery Mobile"로 디자인한 폰갭 샘플 소스를 선택한 것입니다.

폰갭 없는 웹앱은 속빈 강정과 같기 때문에 폰갭에 대해서 현재 잘 알지 못한다 할지라도 폰갭에 어떤 기능이 있는지 정도는 이해하고 경험해야 합니다. 또한 폰갭을 알고 있는 개발자도 현재 사용할 폰갭 버전이 어느 정도까지 지원하고 특성이 무엇인지를 파악할 필요가 있습니다. 그림과 같이 설정하고 "Next" 버튼을 클릭합니다.



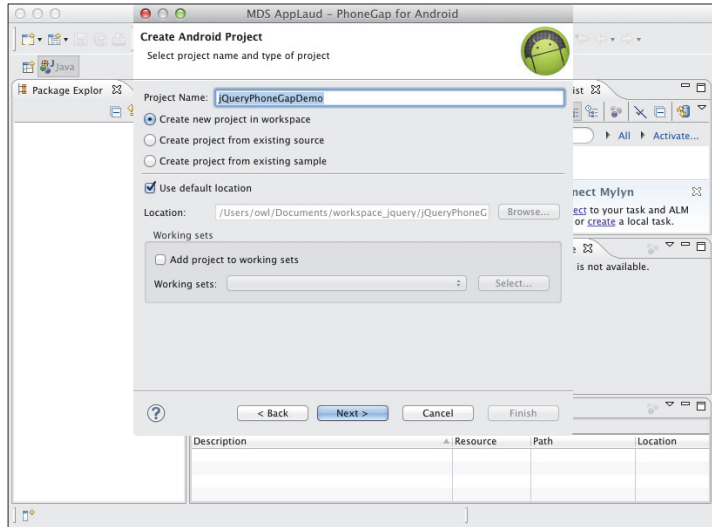
참고 MDS Applaud - PhoneGap for Android

안드로이드용 폰갭 프로젝트 생성 마법사는 "MDS Applaud - PhoneGap for Android"라는 이름으로 마법사 창이 나타나는데, "MDS"란 "Mobile Developer Solutions"의 약자이고 "<http://www.mobiledevelopersolutions.com>" 사이트를 의미합니다. 이 사이트를 찾아보면 "MDS Applaud"에 대한 상세한 정보를 찾아 볼 수 있습니다.

안드로이드 프로젝트 이름 정의

스텝 5

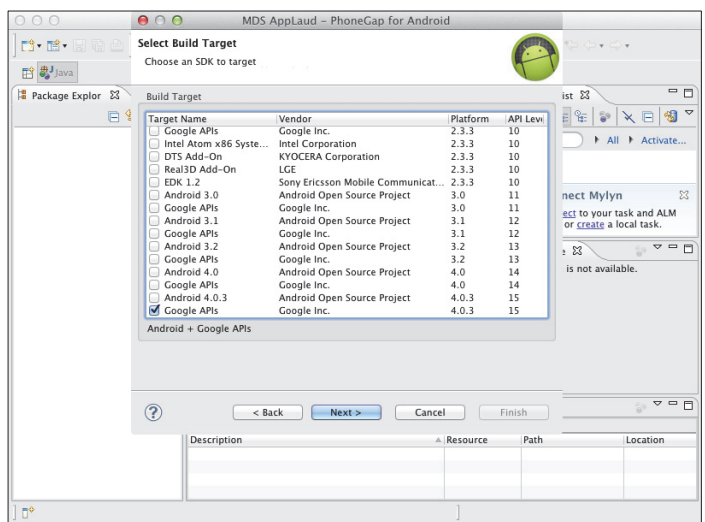
그림과 같이 프로젝트 이름을 정의하고 프로젝트 소스를 저장할 위치를 지정합니다. 본 사례는 이클립스를 시작할 때 정의하였던 워크스페이스(workspace) 위치를 소스 저장 위치로 설정하고 있습니다. "Next" 버튼을 클릭합니다.



안드로이드 프로젝트 SDK 버전 정의

스텝 6

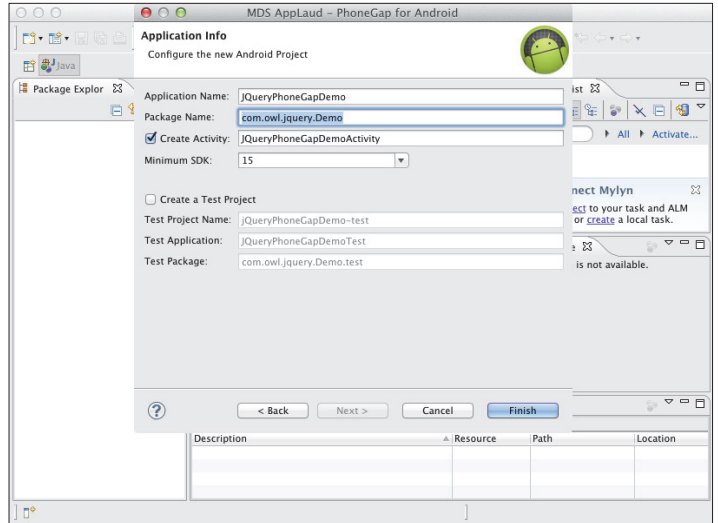
그림과 같이 이 프로젝트에서 사용할 "안드로이드 SDK 버전"을 선택합니다. 본 사례는 안드로이드 버전 중 코드명이 "아이스크림 샌드위치 (Ice Cream Sandwich)"인 "4.0.X 구글 API"를 선택하고 있습니다. 이 버전의 API Level은 "15"임을 기억하기 바랍니다. "Next" 버튼을 클릭합니다.



안드로이드 프로젝트 패키지 이름 정의

스텝 7

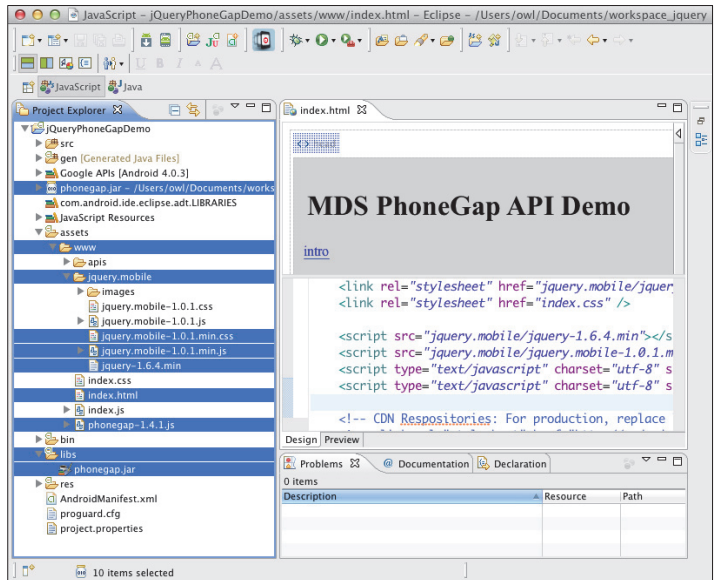
"Application Info"에서는 이 프로젝트가 사용할 자바 패키지명을 정의합니다. 나머지 항목들은 기본 설정에 따라 자동으로 작성됩니다. "Finish" 버튼을 클릭하면 "jQuery Mobile"로 디자인된 "안드로이드용 폰갭 프로젝트"가 생성됩니다.



안드로이드 jQuery 폰갭 프로젝트 생성 완료

스텝 8

위의 과정으로 생성된 웹앱 프로젝트입니다. "Project Explorer" 창에서 보는 바와 같이, 안드로이드 "이이스크립 샌드위치" 버전을 사용하고, "폰갭" 라이브러리가 장착되어 있으며, "jQuery Mobile" 자바스크립트 프레임워크가 탑재된 샘플 프로젝트라는 것을 알 수 있습니다.

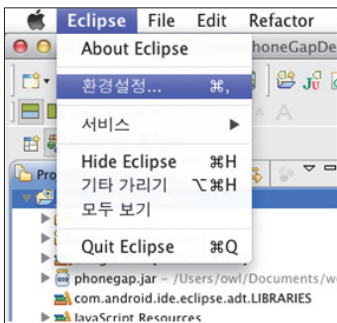


이클립스 설정 : workspace 기본 인코딩 설정

이클립스에 능숙한 개발자라면 언급할 필요가 없지만, 이클립스로 프로젝트를 개발할 때 꼭 알아야 할 사항이 다음의 워크스페이스 인코딩 문제입니다. 알면 당연하지만 모르면 매우 난감한 상황에 빠지는 팁입니다.

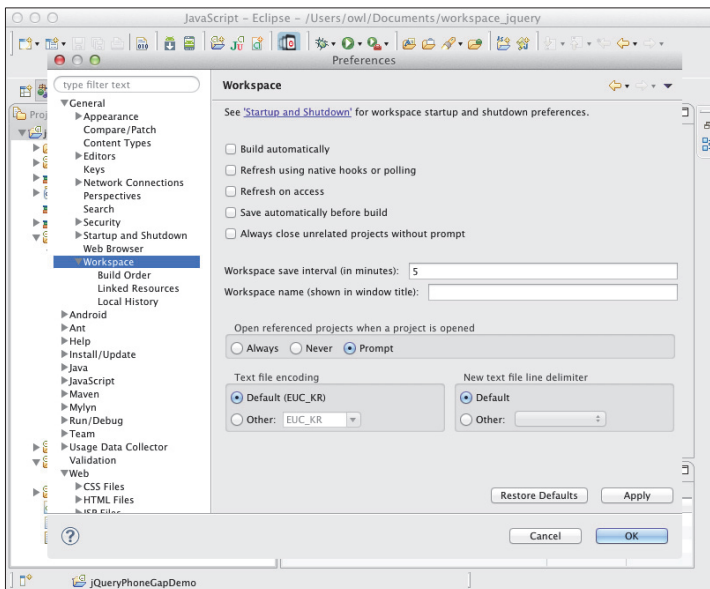
스텝 1

맥의 경우 "Eclipse > 환경설정..." 메뉴를 사용하고, MS Windows의 경우 "Window > Preferences" 메뉴를 사용하여 이클립스 개발 도구에 대한 환경 설정 창을 호출합니다.



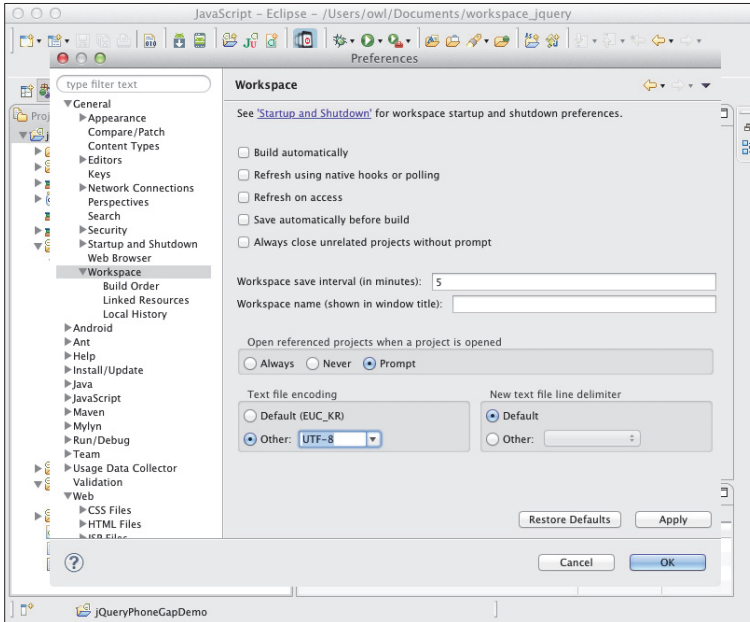
스텝 2

"General > Workspace > Text file encoding" 설정을 보면 그림과 같이 기본 설정이 "EUC-KR"로 설정되어 있는 경우가 많습니다.



스텝 3

이 설정을 "UTF-8"로 설정해야 합니다. 스마트폰은 기본 인코딩이 전 세계 표준 인코딩인 "UTF-8"을 사용하기 때문입니다.



이클립스 설정 : 웹 개발 플러그인 "애타나(Aptana)" 설치

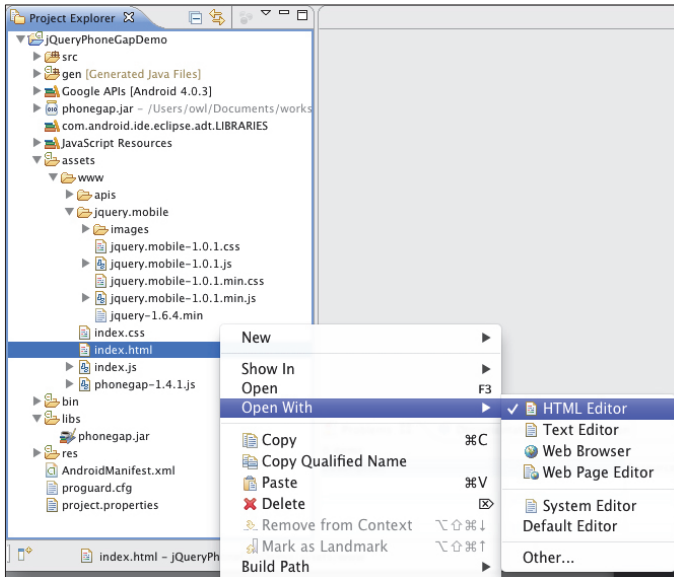
이클립스에서 웹 페이지를 개발할 때 기본 소스 편집기로도 충분히 작업할 수 있지만, 소스 작성 시 알면서도 손이 내 맘대로 움직여 주는 것이 아니라 오타나 기타 Syntax 오류를 만들어 낼 수 있습니다. 특히 HTML5와 자바스크립트, CSS의 경우 이런 문제를 네이티브 앱 개발 프로그램에서 검증하지 않기 때문에 디버깅하는데 어려움이 겪는 경우가 많습니다.

이클립스라는 GUI 개발 도구가 크게 성장한 이유도 여기에 있습니다. 오래된 자바 개발자라면 메모장이나 터미널에서 직접 코딩하고 수동으로 컴파일하는데 많은 불편을 겪었을 것입니다. 처음 이클립스가 나왔을 때 반신반의하는 하드코딩 개발자도 많았지만 지금은 모두 사용합니다. API는 계속 업그레이드되고, 그에 따라 소스 코드는 우리가 감당하기에 너무 방대해졌기 때문이지요.

웹앱은 웹 페이지 소스를 개발하는 작업이기 때문에 드림위버 같은 세계적인 완벽한 웹 개발 도구를 사용하는 방법도 있지만 고가의 유료 개발 도구라는 문제가 있습니다. 이에 반해 다음에서 소개하는 애타나(Aptana)는 현재 무료이고 이클립스 기반으로 만든 웹 개발 도구이어서 웹앱을 개발하는데 유용합니다.

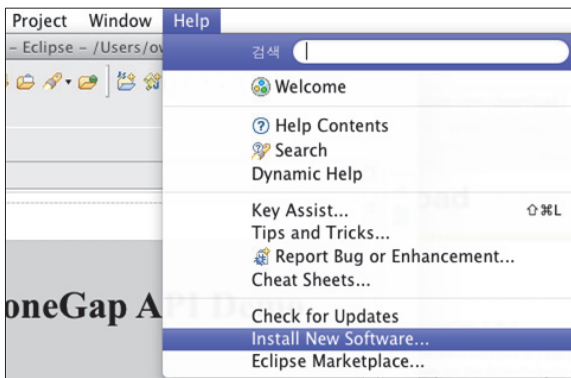
스텝 1

기본 이클립스만 설치된 상태라면 그림과 같이 이클립스에서 제공하는 기본 "HTML Editor"가 제공됩니다. 참고로 "Web Page Editor"의 경우 "Eclipse Marketplace"에서 추가 설치한 편집기인데 이 편집기는 소스와 디자인 미리보기를 동시에 살펴보면서 HTML 파일에 대한 작업을 할 수 있는 특징이 있습니다.



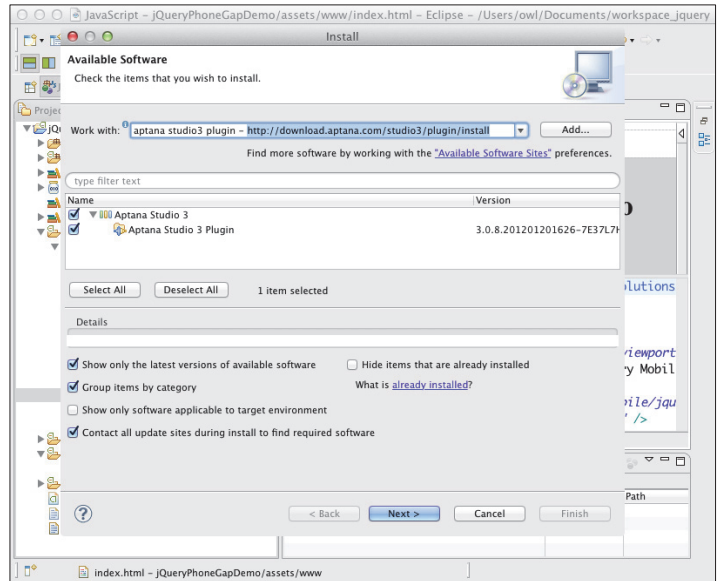
스텝 2

"Help > Install New Software..." 메뉴를 실행하면 "Eclipse Marketplace"에는 등록되어 있지 않지만 독자적으로 배포하는 이클립스 플러그인 프로그램을 다운받아 설치할 수 있습니다. 물론 이 방식은 배포하는 플러그인 프로그램에 대한 배포 서버 주소를 알고 있어야 합니다.



스텝 3

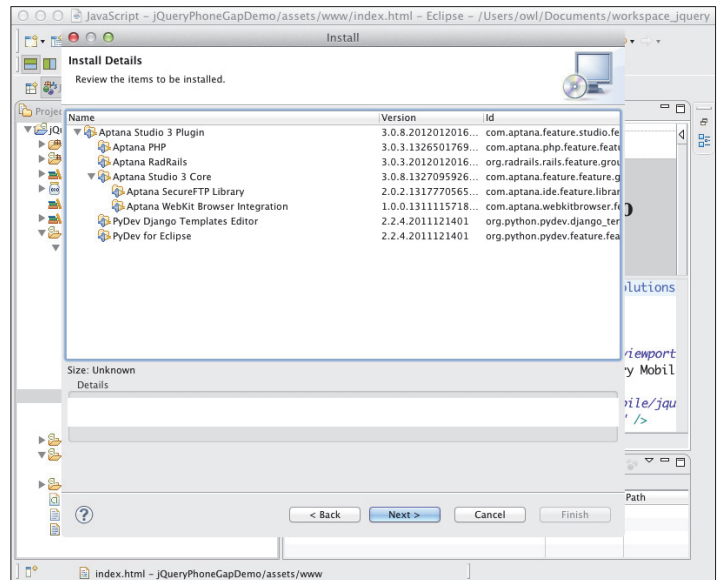
그림과 같이 플러그인 설치 창에서 배포 서버 주소를 입력하고 검색하면 앵타나 플러그인을 찾을 수 있습니다. 참고로 이 주소는 앵타나의 사정에 따라 바뀔 수도 있습니다. 그럴 때는 당연히 구글링에서 앵타나 배포 주소를 찾아 봐야 할 것입니다. 앵타나 패키지를 체크하여 선택하고 "Next" 버튼을 클릭합니다.



- 앵타나 배포 주소 : <http://download.aptana.com/studio3/plugin/install>

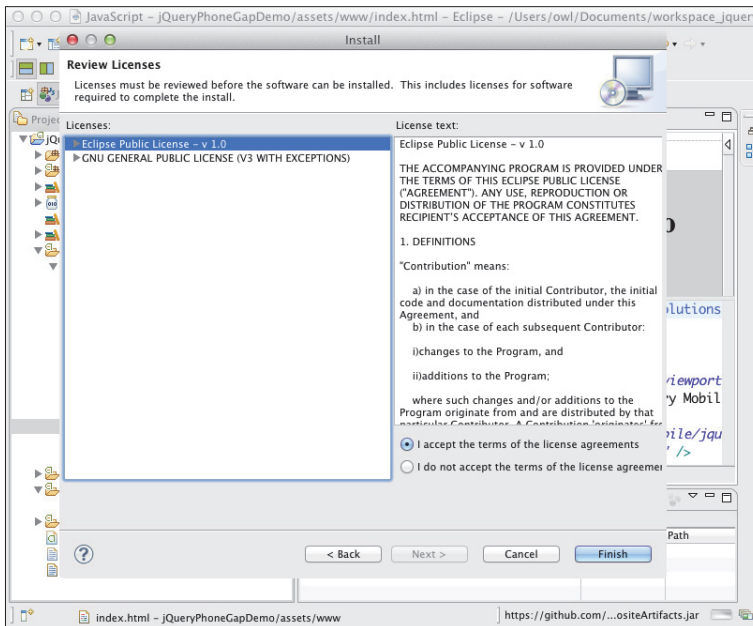
스텝 4

설치할 앵타나 패키지들을 확인하고 "Next" 버튼을 클릭합니다.



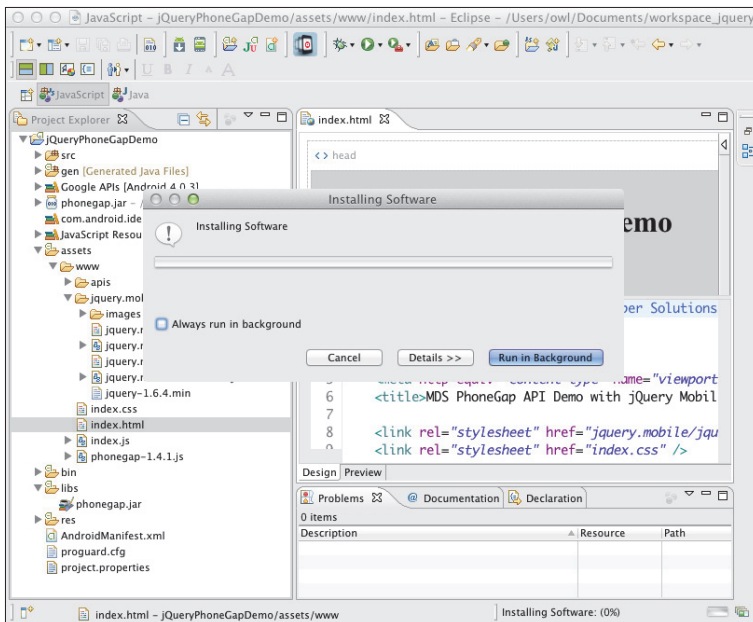
스텝 5

라이선스에 동의하고 "Finish" 버튼을 클릭하여 설치를 실행합니다.



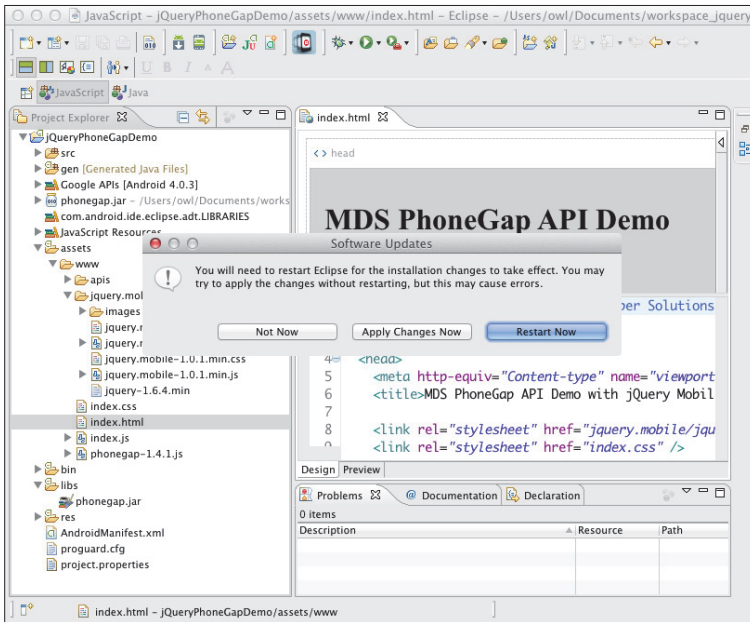
스텝 6

그림과 같이 앱타나 설치 과정이 진행됩니다.



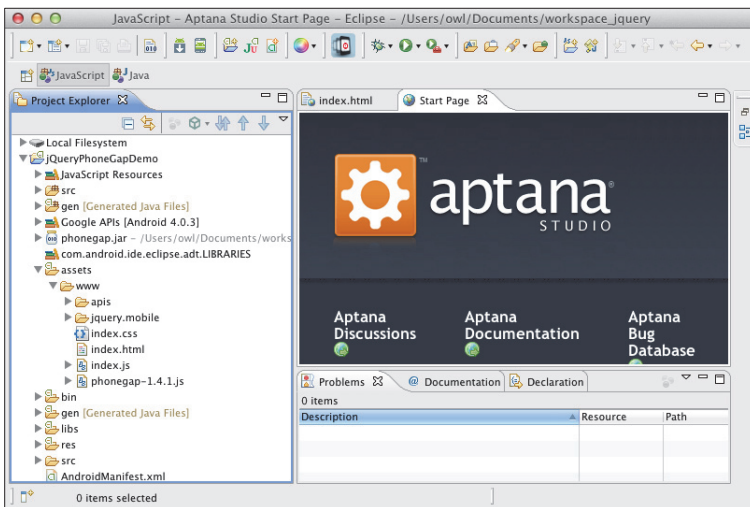
스텝 7

애플타나 설치가 완료되면 그림과 같이 대화상자가 나타나고 "Restart Now" 버튼으로 이클립스를 재시동합니다.



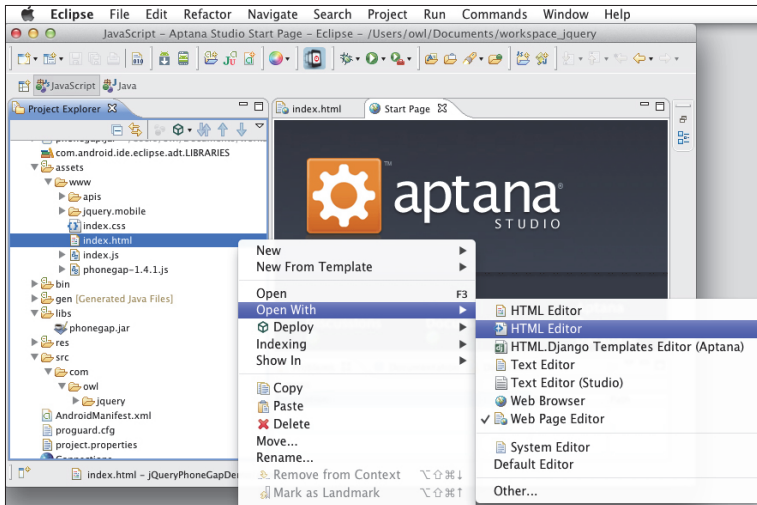
스텝 8

이클립스가 재시동되고 나면 그림과 같이 애플타나 안내 화면이 나타납니다.



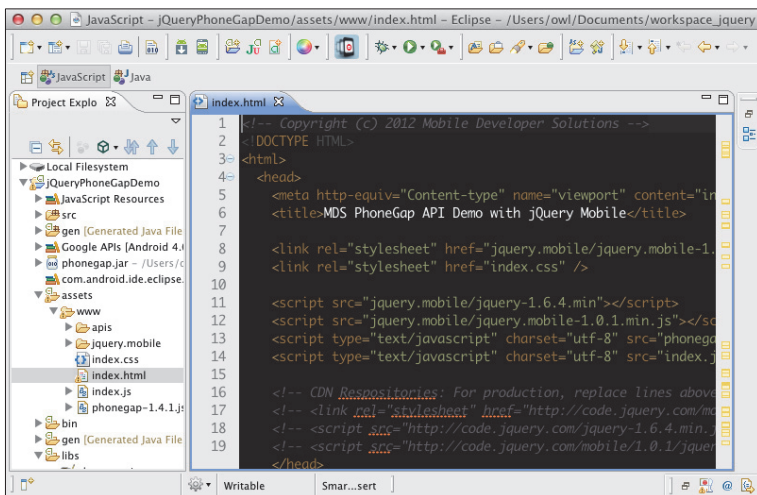
스텝 9

그림과 같이 index.html 파일을 선택하고 마우스 오른쪽 버튼으로 컨텍스트 메뉴를 열면 "애플타나용 HTML Editor" 메뉴를 찾을 수 있습니다.



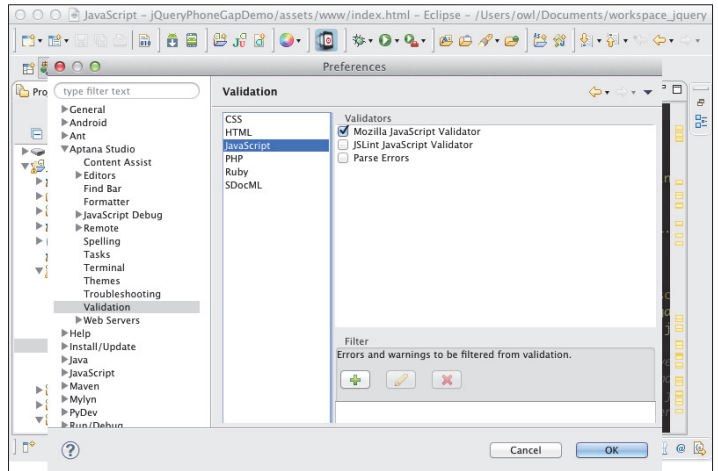
스텝 10

애플타나 HTML Editor로 index.html 파일을 열면 그림과 같이 나타납니다. 애플타나 편집기는 다른 편집기에 비해 월등히 편리한 기능들을 제공합니다.



이클립스 설정 : 웹 표준 코딩을 위한 Validation 설정

웹 페이지를 편리하게 개발하려면 앞서 언급한 바와 같이 웹 표준에 맞는 코딩을 할 수 있도록 도와주는 Validation 설정을 할 필요가 있습니다. 그림과 같이 앵타나의 Validation은 HTML, CSS, Javascript 등에 대한 웹 표준 서비스를 제공합니다.

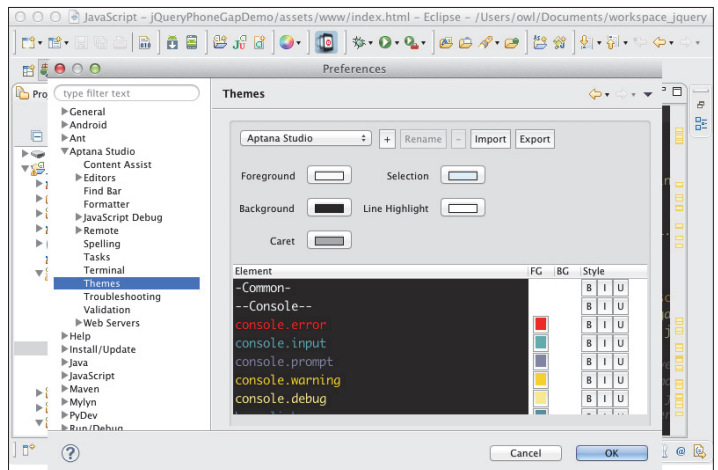


이클립스 설정 : 웹 편집기 테마 설정

앵타나 30은 기본 편집기 테마가 터미널형으로 되어 있습니다. 터미널에 익숙한 개발자라면 좋아하겠지만 그렇지 않는 경우가 대부분입니다. 좀 어렵게 보이는 선입견이 있는 것이지요. 이 테마를 다음과 같이 이클립스 환경 설정에서 변경할 수 있습니다.

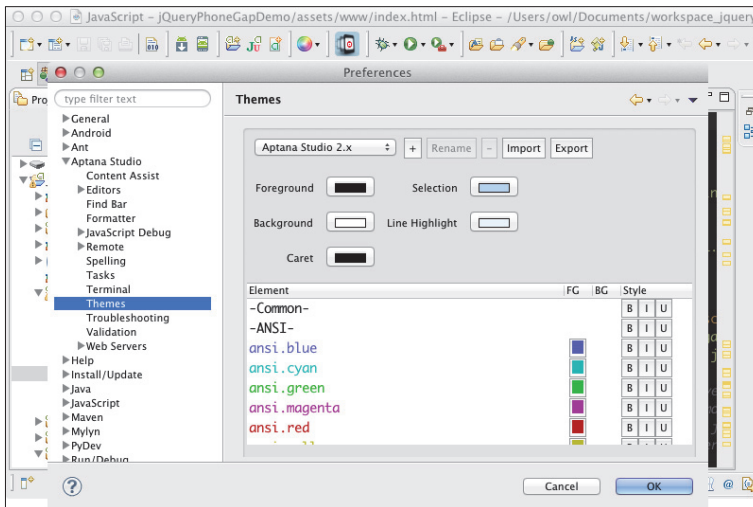
스텝 1

다음 그림은 앵타나 기본 테마로 "Aptana Studio"가 설정되어 있습니다.



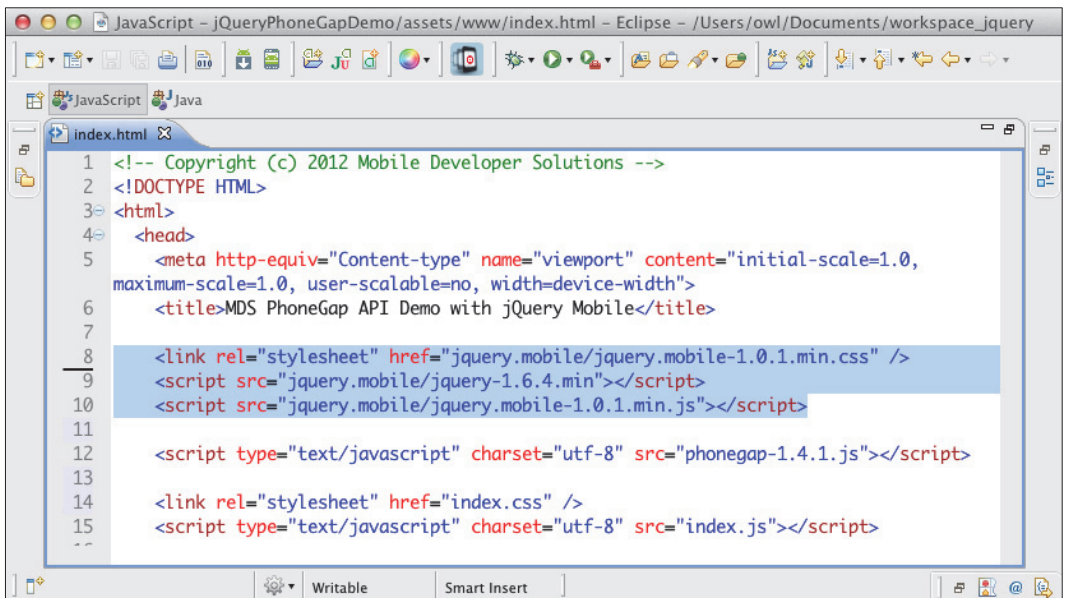
스텝 2

엡타나 테마를 "Aptana Studio 2.x"로 변경 선택하고 "OK" 버튼을 클릭합니다.



jQuery Mobile 라이브러리 호출 사례

이 데모 프로젝트의 index.html 파일을 열어보면, 그림과 같이 jQuery Mobile 라이브러리를 호출하는 구문이 있습니다. 이 구문이 jQuery Mobile 프레임워크를 호출하는 구문입니다. 각 라이브러리 파일의 기능은 다음과 같습니다.



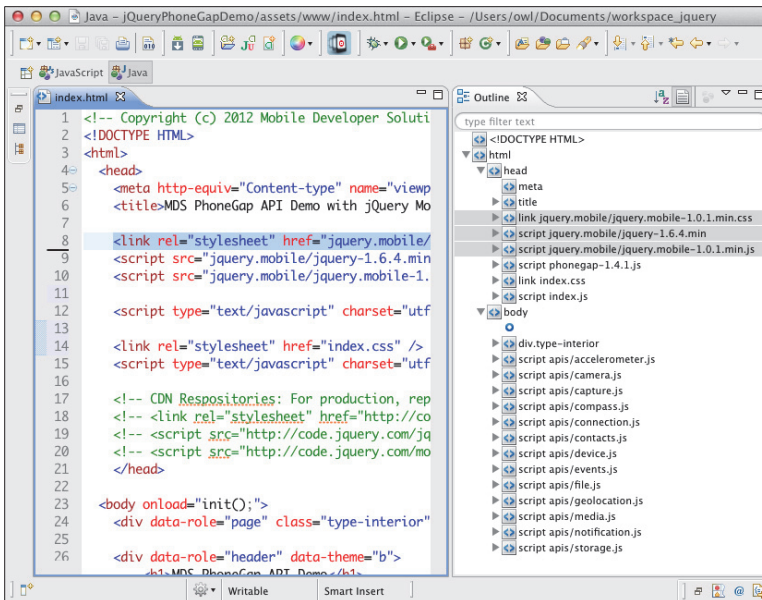
소스라인 8 : "jQuery Mobile CSS 라이브러리"를 호출합니다.

소스라인 9 : "jQuery 기본 라이브러리"를 호출합니다.

소스라인 10 : "jQuery Mobile 라이브러리"를 호출합니다.

HTML 구성을 한눈에 보는 이클립스 "Outline" 창

이클립스에서는 "Outline" 창을 통해 HTML 파일에 배치된 객체들을 한눈에 탐색할 수 있는 기능을 지원합니다. 이 창을 통해 HTML 문서 뿐 아니라 각종 소스의 객체들을 탐색할 수 있습니다.

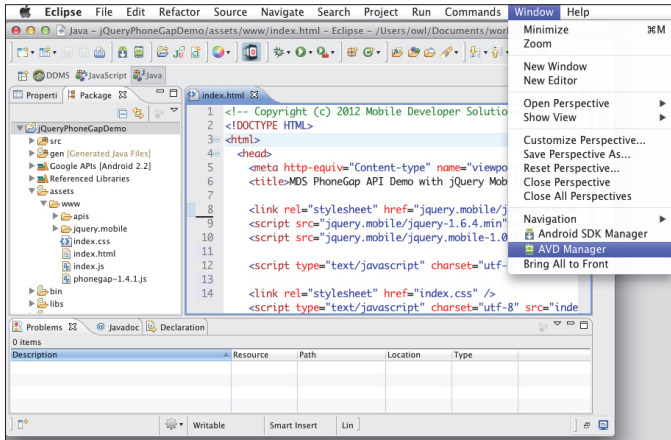


가상기기 준비하기 : SDK 4.0.X 아이스크림 샌드위치 버전

이미 데모 웹앱 프로젝트는 완성된 상태입니다. 이 프로젝트를 안드로이드 가상기기에서 실험하려면 다음과 같이 이 프로젝트의 안드로이드 버전에 알맞은 가상기기를 준비해야 합니다.

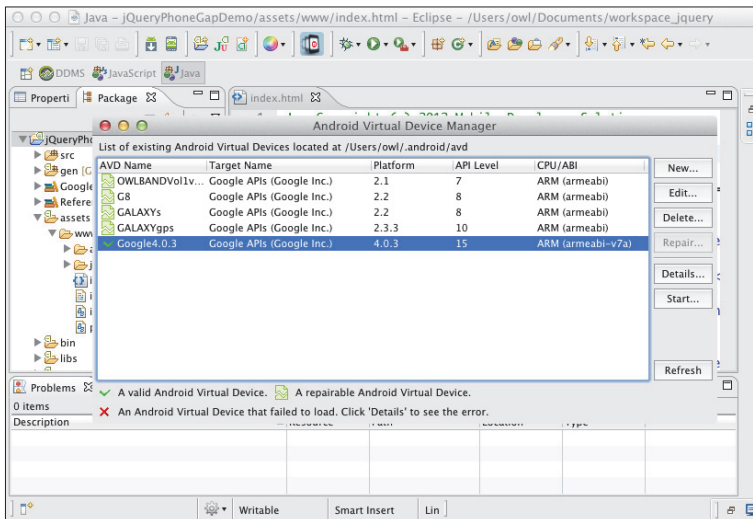
스텝 1

“Window > AVD Manager” 메뉴를 실행합니다.



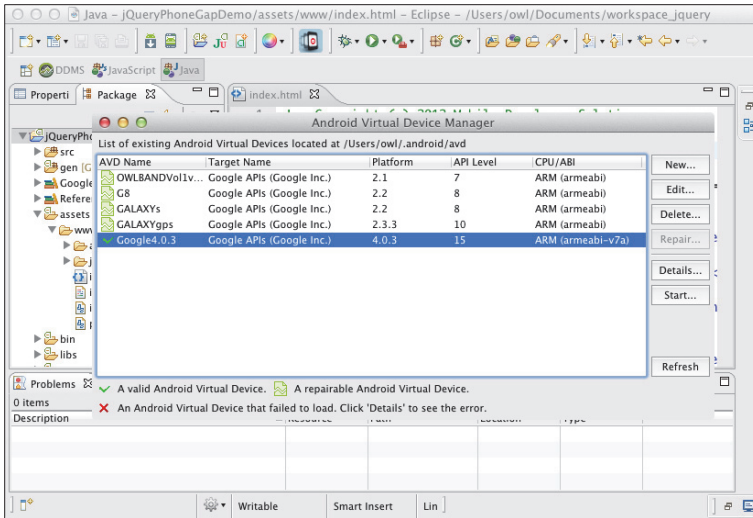
스텝 2

"Android Virtual Device Manager" 창에서 그림과 같이 Google APIs 4.0.X 버전용 가상기기를 준비합니다. 가상기기는 "New..." 버튼 또는 "Edit..." 버튼으로 생성, 변경할 수 있습니다.



스텝 3

본 사례에서는 그림과 같이 가상기기에서 지원 가능한 모든 옵션을 추가했습니다.

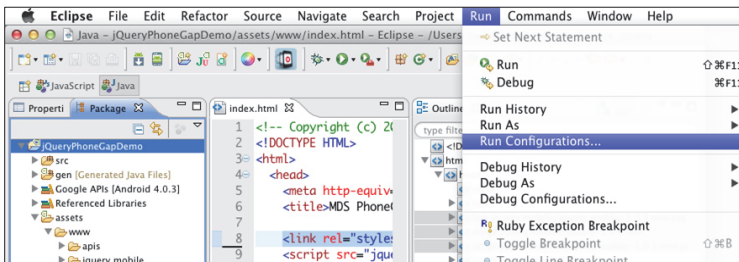


안드로이드 프로젝트 실행 환경 설정

안드로이드용 웹앱을 가상기기 또는 실물 단말기에서 선별적으로 실험하려면 다음과 같이 실행 환경을 "Manual(수동 방식)"으로 설정할 필요가 있습니다.

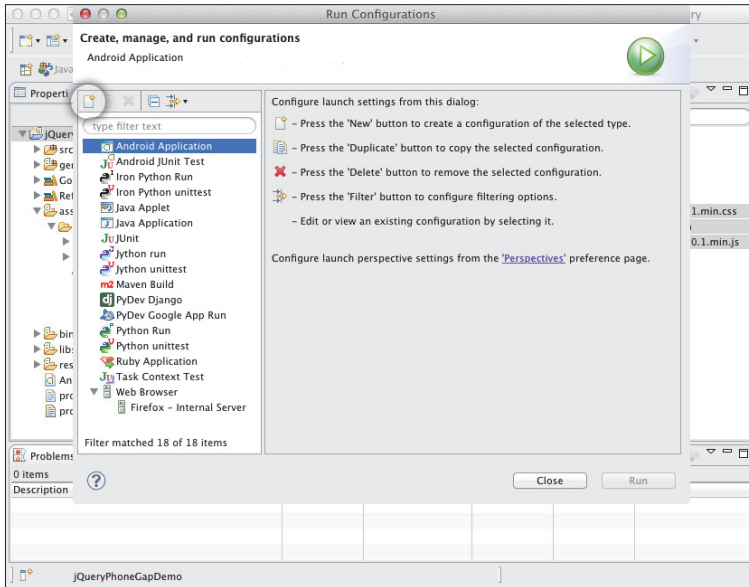
스텝 1

"Run > Run Configurations..." 메뉴를 실행합니다.



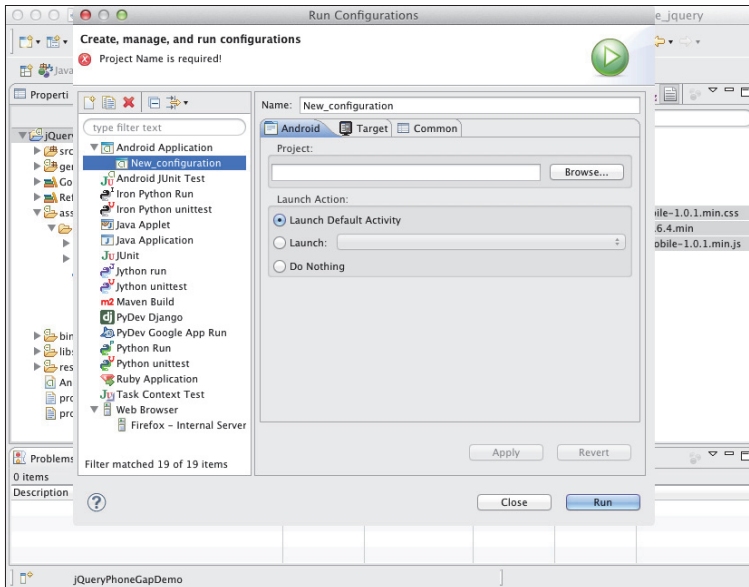
스텝 2

"Run Configurations > Android Application"을 선택하고, 왼쪽 위에 있는 "New Launch Configuration" 아이콘 버튼을 클릭합니다.



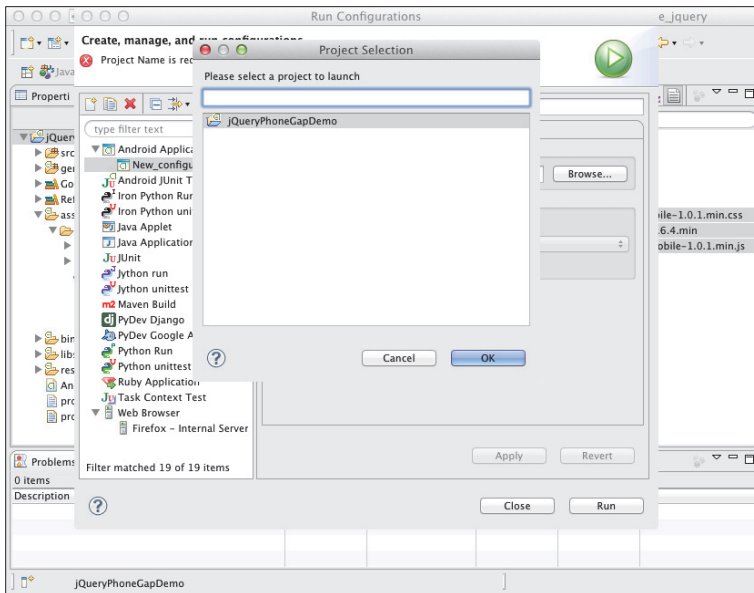
스텝 3

그림과 같이 "Android Application"에 "New_configuration"이 생성되었습니다. 필요에 따라 이 이름을 변경할 수 있습니다. "New_configuration > Android > Project > Browse..." 버튼을 클릭하여 이 환경에서 사용할 프로젝트를 선택합니다.



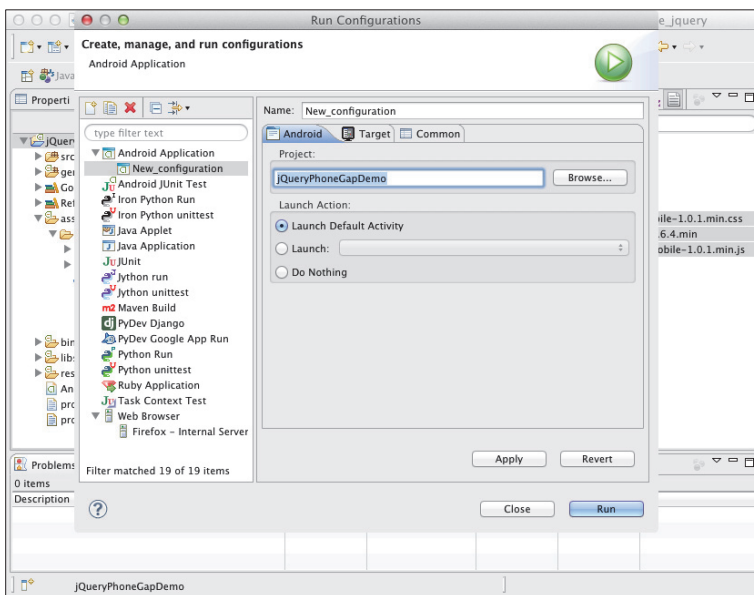
스텝 4

그림과 같이 "Project Selection" 창에서 실행할 프로젝트를 선택하고 "OK" 버튼을 클릭합니다.



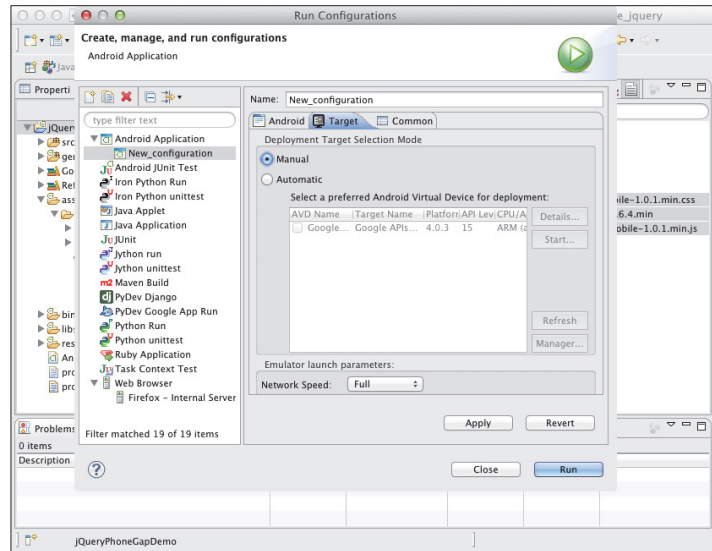
스텝 5

그림과 같이 이 실행 환경에서 사용할 안드로이드 프로젝트를 설정했습니다.



스텝 6

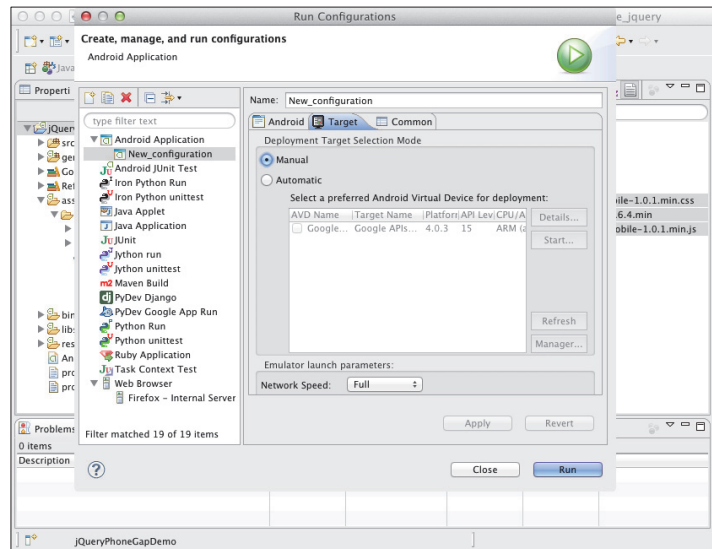
"New_configuration > Target > Deployment Target Selection Mode"를 "Manual"로 선택합니다. 이 설정은 실험 의도에 따라 원하는 단말기를 선택할 수 있어 개발자에게는 필수적인 설정입니다. "Apply" 버튼을 클릭하여 설정을 적용합니다.



가상기기 실행하기 : SDK 4.0.X 아이스크림 샌드위치 버전

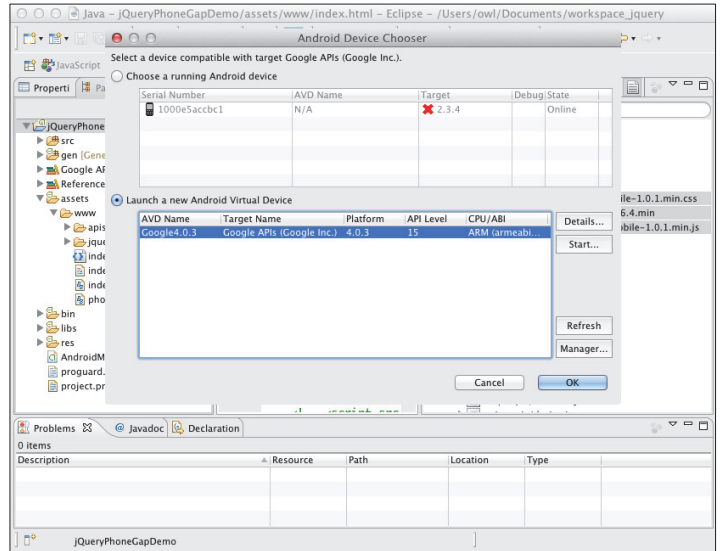
스텝 1

이 창에서 "Run" 버튼을 클릭하거나 "Run > Run As > Android Application" 메뉴를 실행하면 프로젝트를 실행할 수 있습니다.



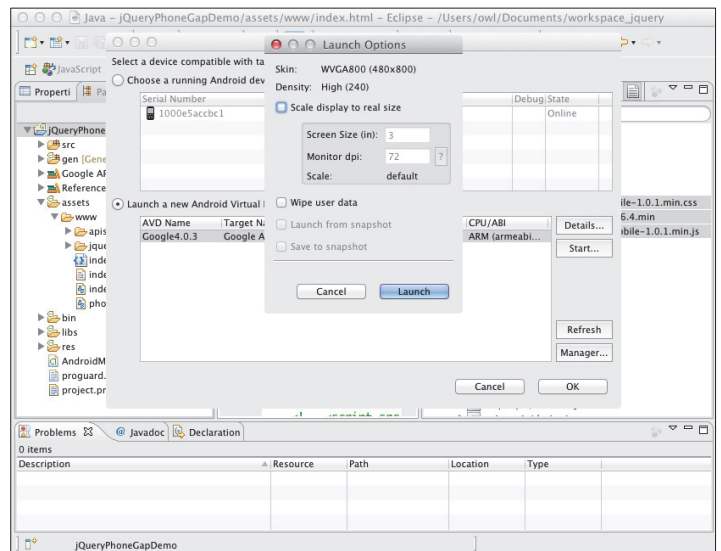
스텝 2

프로젝트에 실행에 앞서 그림과 같이 "Android Device Chooser" 창이 나타납니다. 필자는 실물 단말기 한대를 개발 컴퓨터에 연결한 상태인데, 이 실물 단말기는 2.3.4 버전이고 프로젝트는 4.0.X 버전이기 때문에 실물 단말기에서 실험할 수 없습니다. 앞서 준비한 가상기기를 사용해야 합니다. 먼저 앞서 준비한 가상기기를 선택하고 "Start..." 버튼을 클릭하여 가상기기를 실행합니다.



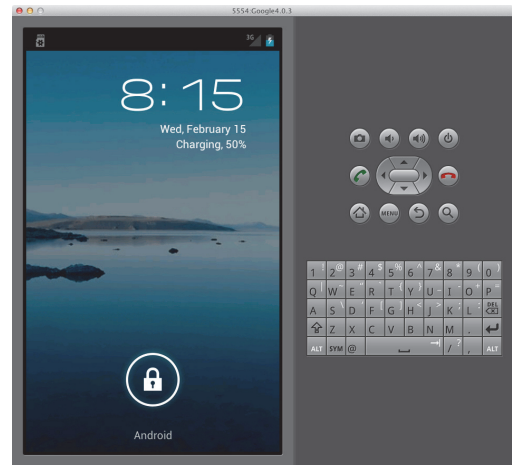
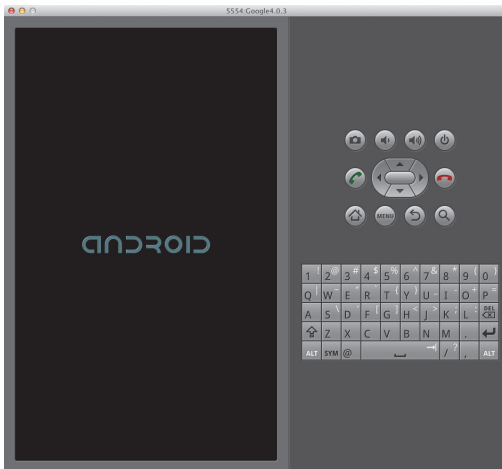
스텝 3

안드로이드 가상기기를 실행하면 "Launch Options" 창이 나타납니다. 가상기기를 실행하는 몇 가지 옵션들이 있는데 상세한 설명은 생략합니다. 그림과 설정하고 "Launch" 버튼을 클릭합니다.



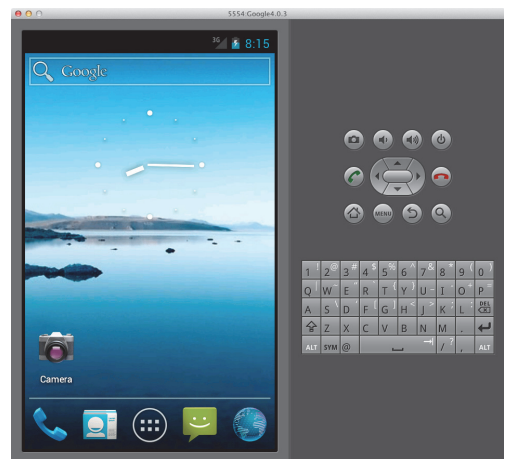
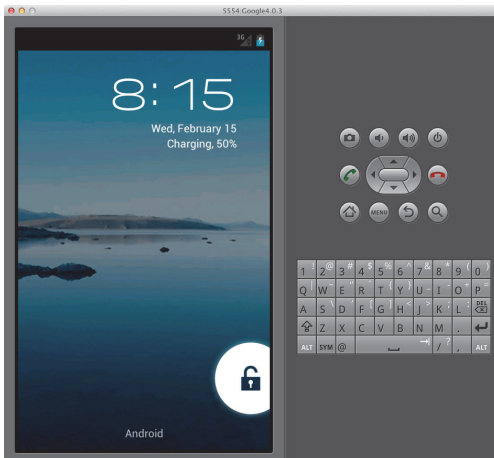
스텝 4

그림과 같이 가상기기가 실행되면서 부팅을 시작합니다.



스텝 5

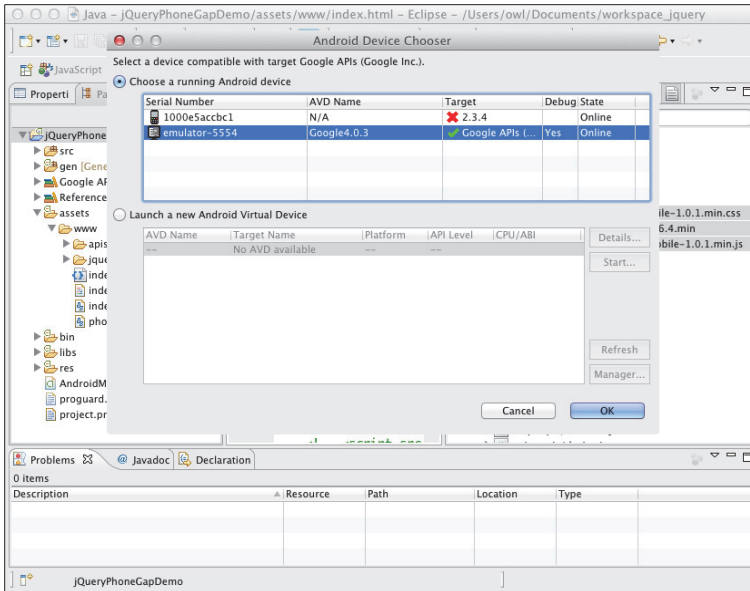
부팅이 완료되면 잠금을 해제합니다.



가상기기에서 jQuery Mobile 폰갭 데모 앱 실행하기

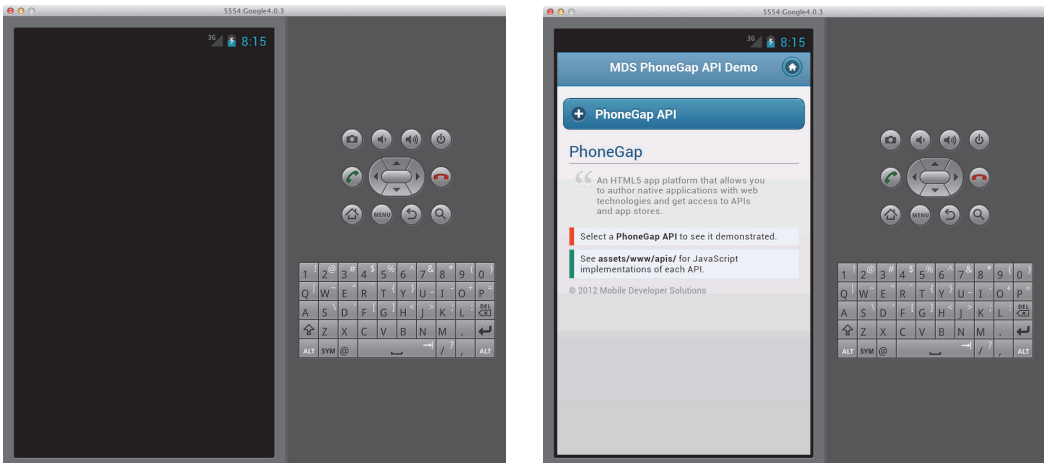
스텝 6

이제 그림과 같이 실행중인 가상기기를 선택하고 "OK" 버튼을 클릭하면 준비한 안드로이드 웹앱 프로젝트를 실행해볼 수 있습니다.



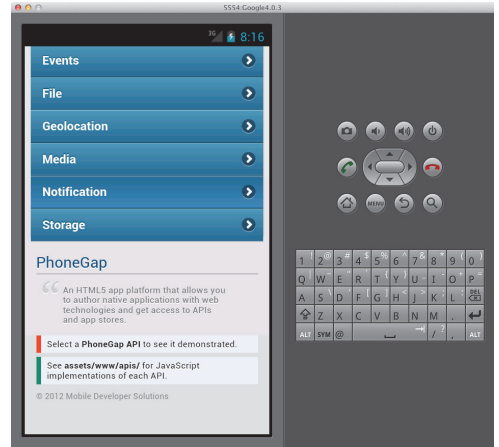
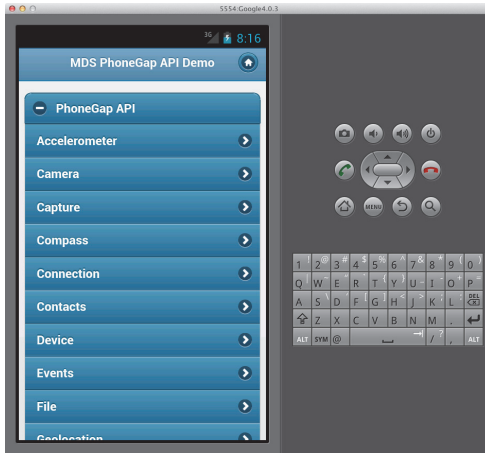
스텝 7

이클립스는 지정한 가상기기에 컴파일한 안드로이드 패키지를 전송하고 설치한 후 실행합니다. 그림과 같이 jQuery Mobile로 디자인된 폰갭 데모 앱이 가상기기에 나타납니다.



스텝 8

“PhoneGap API” 버튼을 클릭하면 폰갭의 기능들을 실험해볼 수 있는 메뉴들이 펼쳐집니다.

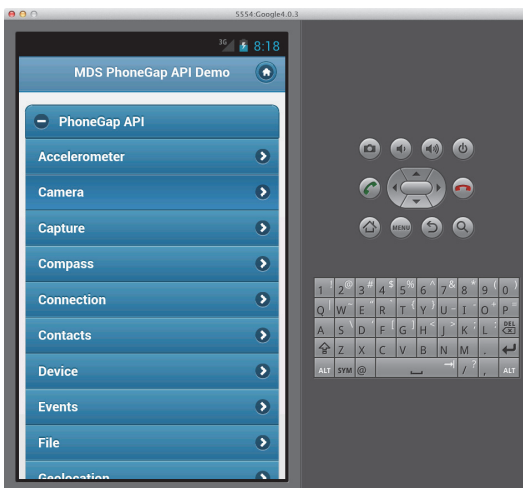


가상기기 폰갭 데모 : 가속 센서(Accelerometer)

jQuery Mobile에 대해 논하기 전에 웹앱의 근간인 폰갭의 기능들을 전반적으로 파악할 필요가 있습니다. 본서에서 폰갭의 기술들을 상세히 소개할 수는 없지만 다음의 실험을 통해 폰갭의 기능들을 개략적으로 파악하기를 바랍니다. 또한 이 장 집필 당시 최신 버전인 폰갭 1.4.1 버전의 특성과 가상기기에서 실험할 수 있는 기능과 실물 단말기에서 실험해야 할 기능을 파악해보기 바랍니다.

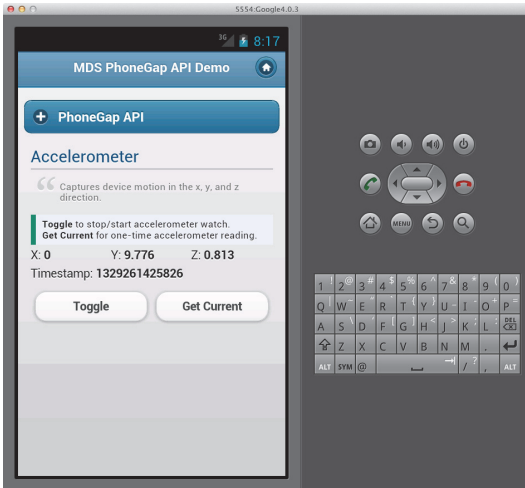
스텝 1

Accelerometer는 폰갭이 단말기의 가속 센서와 연동하여 가속 정보를 X, Y, Z 좌표 값으로 표현합니다. 이 데모는 현재의 가속 센서를 감지하는 "Get Current" 버튼과 주기적 가속 정보를 감지하는 "Toggle" 버튼으로 구성되어 있습니다.



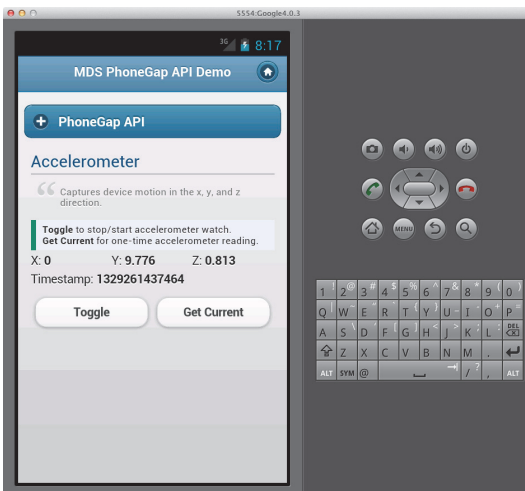
스텝 2

"Get Current" 버튼을 클릭하면 그림과 같이 가상기기의 현재 가속 정보를 화면에 출력하는데, 사실상 가상기기는 단말기의 움직임을 감지하지 못하지만, 앞서 이 가상기기에 가속 센서 옵션을 추가했기 때문에 이 실험이 가능합니다. 실제 개발에서는 이와 같은 센서 관련 실험은 실물 단말기에서 실행하는 것이 적합합니다.



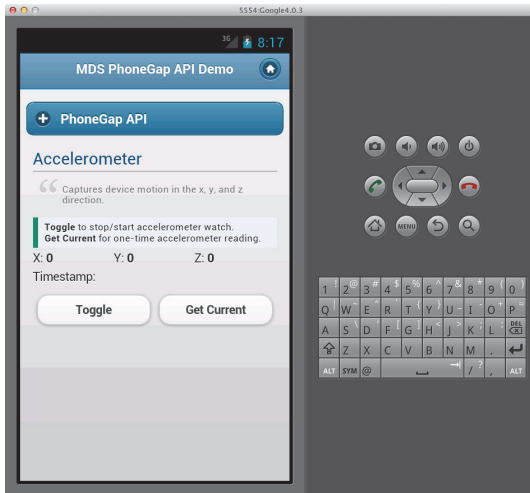
스텝 3

"Toggle" 버튼을 클릭하면 특정 시간을 주기로 단말기의 가속 정보를 감지하여 화면에 출력하는데 이 실험은 가상기기에서 하는 것이기 때문에 그림과 같이 똑같은 가속 값만 출력합니다. 따라서 이 실험은 실물 단말기에서 하는 것이 올바르다 할 수 있습니다. 물론 이클립스의 DDMS 기능을 이용하여 가상 가속 값을 설정하여 실험할 수도 있습니다.



스텝 4

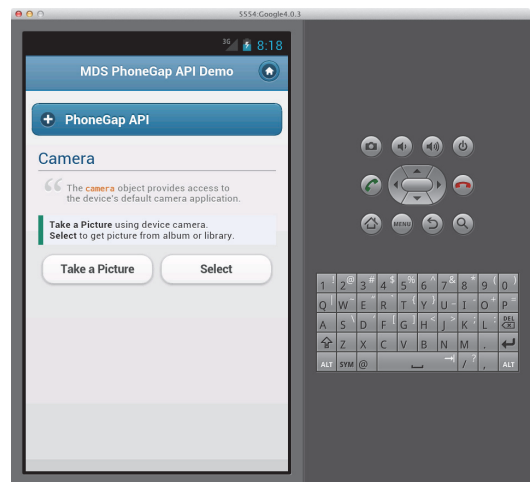
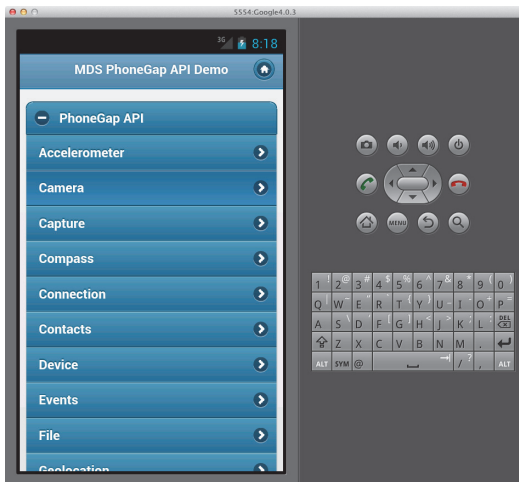
"Toggle" 버튼을 다시 클릭하면 주기적 가속 센서 감지를 종료합니다.



가상기기 폰갭 데모 : 카메라(Camera)

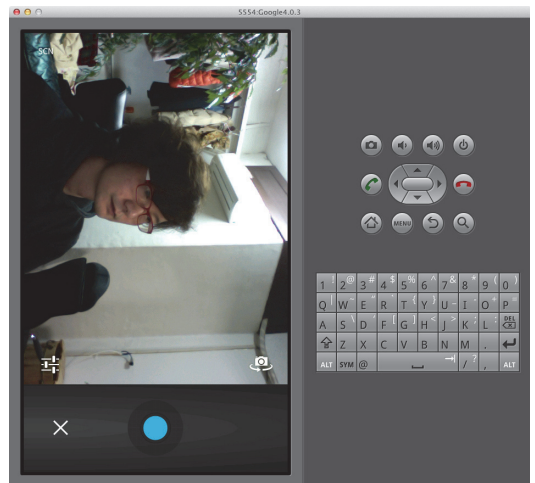
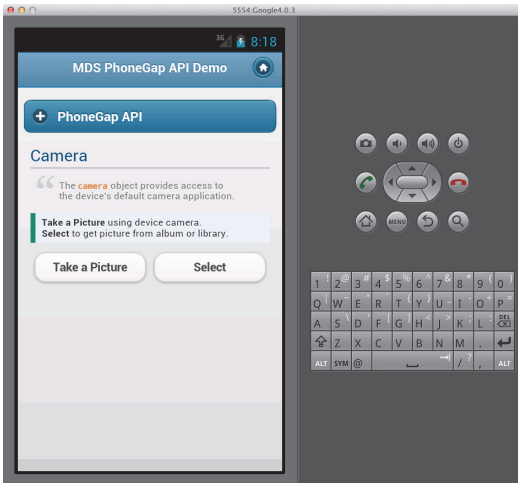
스텝 1

"PhoneGap API" 버튼을 클릭하면 다시 목록 버튼이 펼쳐지고, 이번엔 "Camera" 버튼을 클릭하여 폰갭의 카메라 기능을 실험합니다. 폰갭의 카메라 API에는 단말기 카메라를 이용하여 사진을 찍거나 동영상을 녹화하여 결과 파일을 받아오는 기능이 있고, 이미 찍어놓은 사진이나 동영상을 갤러리에서 선택해오는 두 가지 방식이 있습니다.



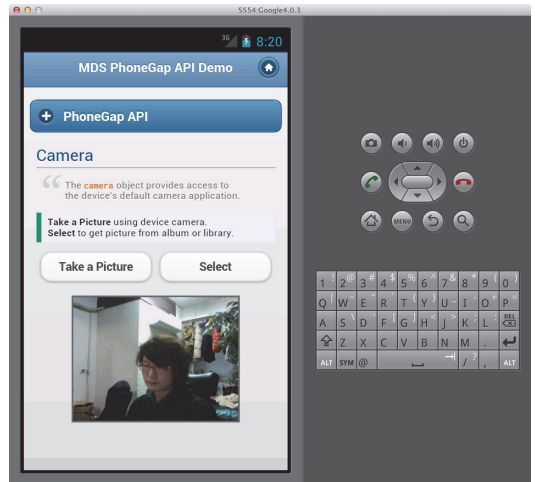
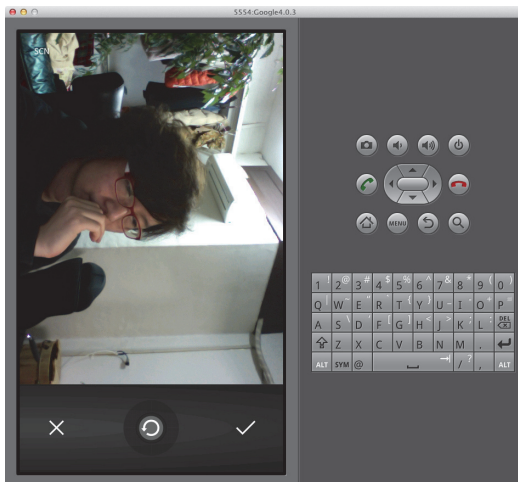
스텝 2

"Take a Picture" 버튼을 클릭하면 카메라 미리보기 화면이 나타납니다. 이 역시 필자가 실험중인 가상기기에 카메라 옵션을 추가했기 때문이고 필자의 개발 컴퓨터인 맥북에 카메라가 장착되어 있기 때문입니다.



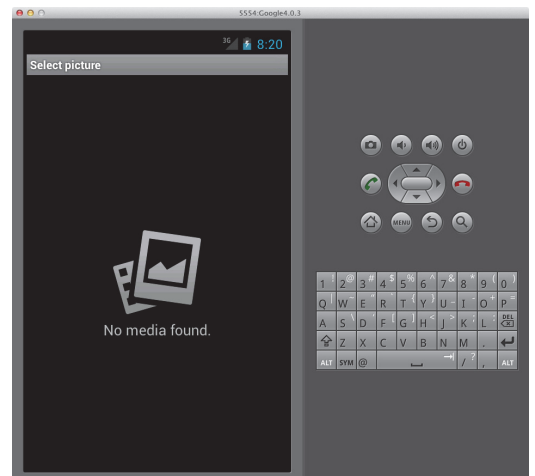
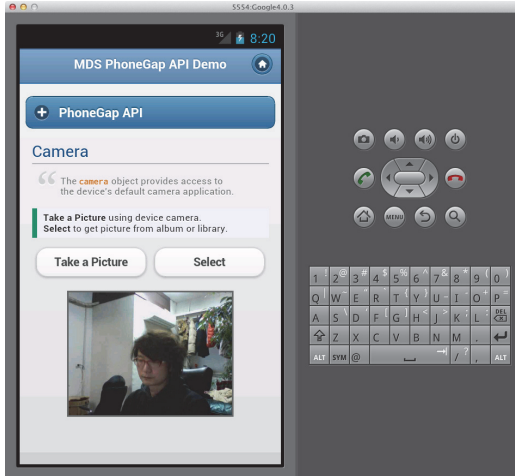
스텝 3

카메라 "셔터" 버튼을 클릭하여 사진을 찍고, "확인" 버튼을 클릭하면 찍은 사진을 가져옵니다.



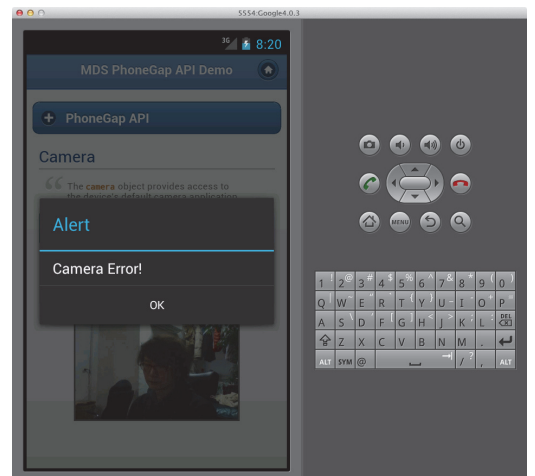
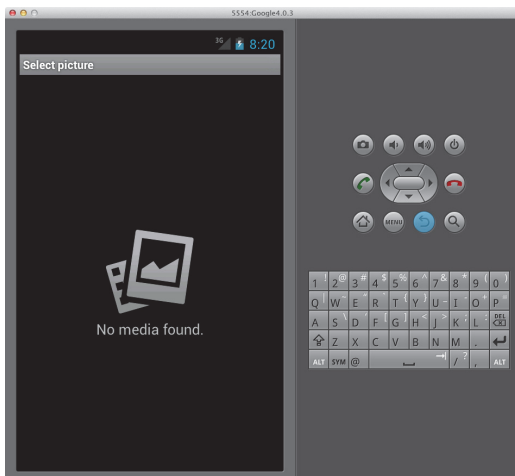
스텝 4

"Select" 버튼을 클릭하면 앨범(갤러리)에서 사진을 가져올 수 있는데, 최초에는 찍어둔 사진이 없기 때문에 그림과 같이 사진이 없다는 안내문을 볼 수 있습니다.



스텝 5

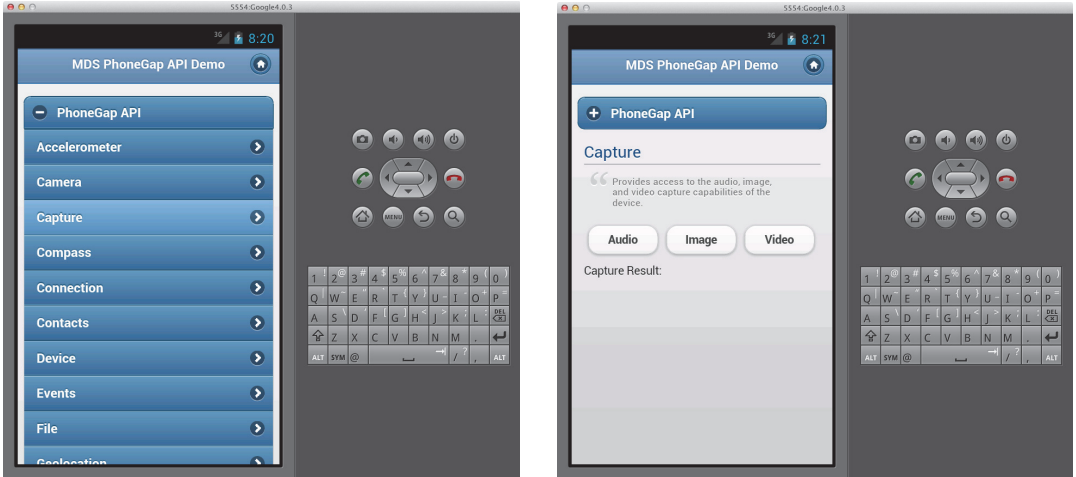
"Back" 버튼을 클릭하면 이전 화면으로 돌아오면서 오류 안내 대화상자가 나타납니다. 이는 프로그램의 오류가 아니라 사진을 선택하지 않았기 때문에 출력하는 안내문입니다. "OK" 버튼을 클릭하여 대화상자를 닫습니다.



가상기기 폰갭 데모 : 캡처(Capture)

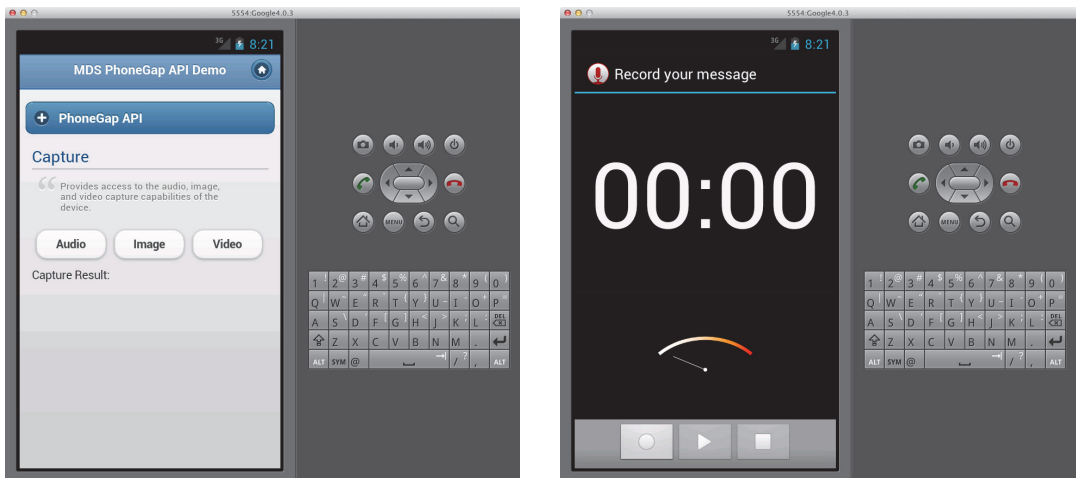
스텝 1

"PhoneGap API" 목록에서 "Capture" 버튼을 클릭하고 폰갭의 캡처 기능을 실험해봅니다.
폰갭의 캡처 API는 녹음, 사진, 동영상상을 캡처하는 것이 주요 기능입니다.



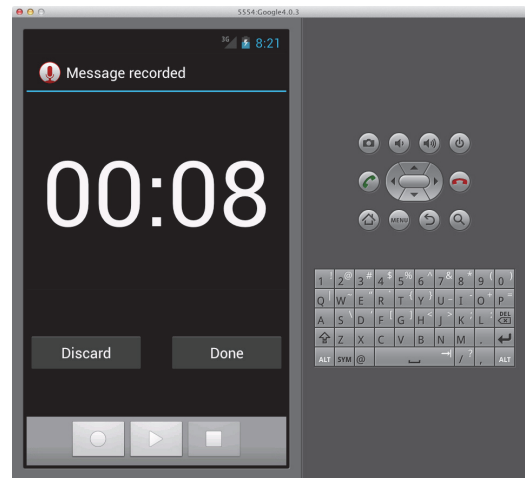
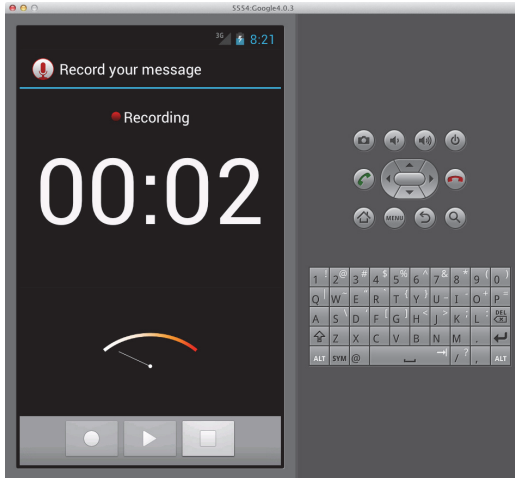
스텝 2

"Audio" 버튼을 클릭하면 녹음을 할 수 있는 화면으로 전환됩니다.



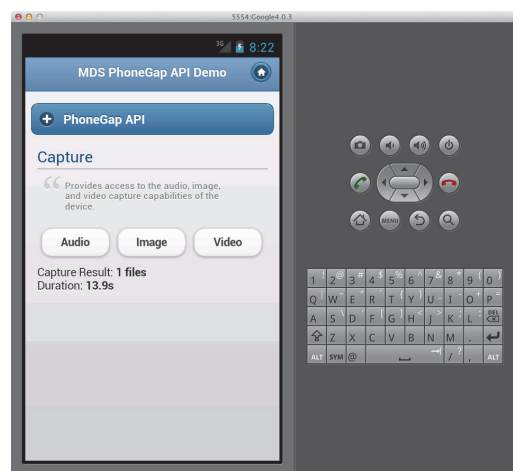
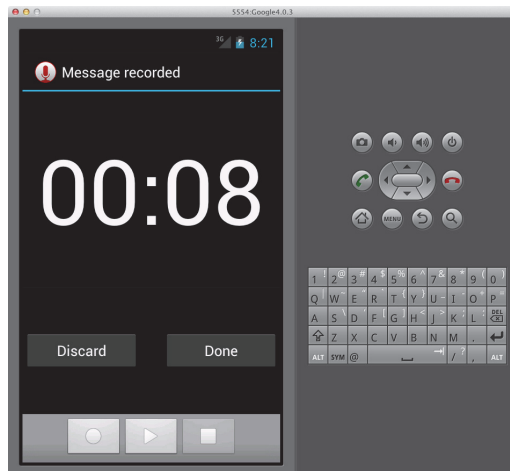
스텝 3

"녹음" 버튼을 클릭하면 녹음이 시작되고, "종료" 버튼을 클릭하면 이 녹음을 사용할 것인지 재녹음을 할 것인지 선택하는 버튼이 나타납니다.



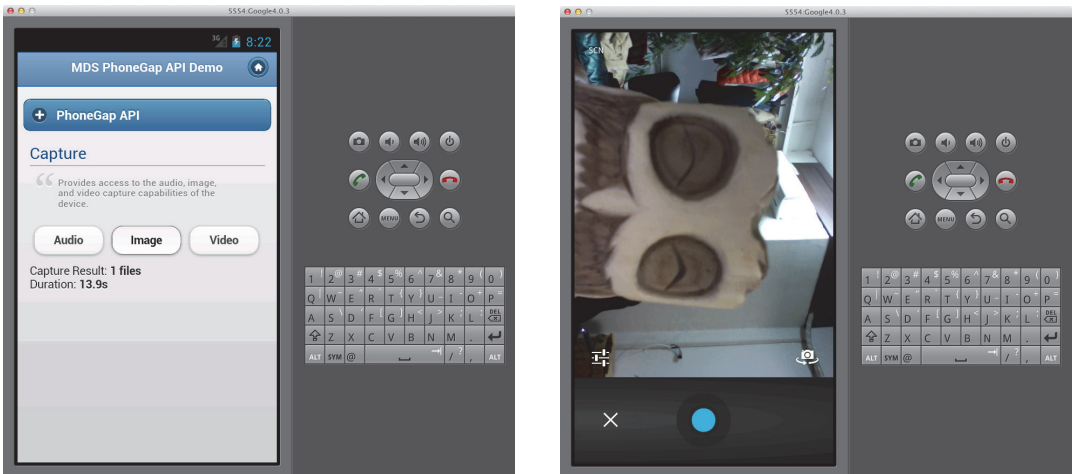
스텝 4

"Done" 버튼을 클릭하면 그림과 같이 녹음한 음원 파일 정보를 가져와 화면에 출력합니다.



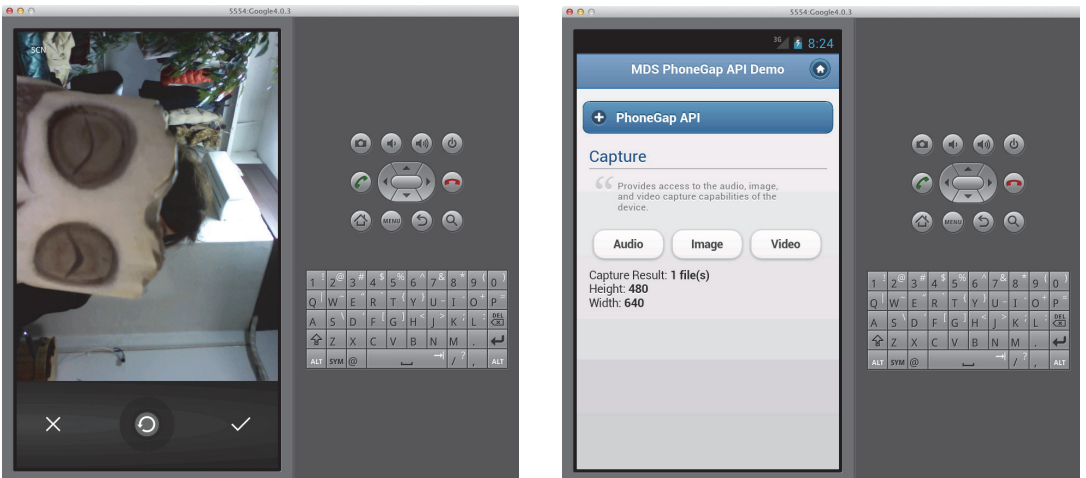
스텝 5

"Image" 버튼을 클릭하면 카메라 사진찍기 화면으로 전환됩니다.



스텝 6

사진을 찍고 확인 버튼을 클릭하면 그림과 같이 찍은 사진에 대한 정보가 화면에 출력됩니다.



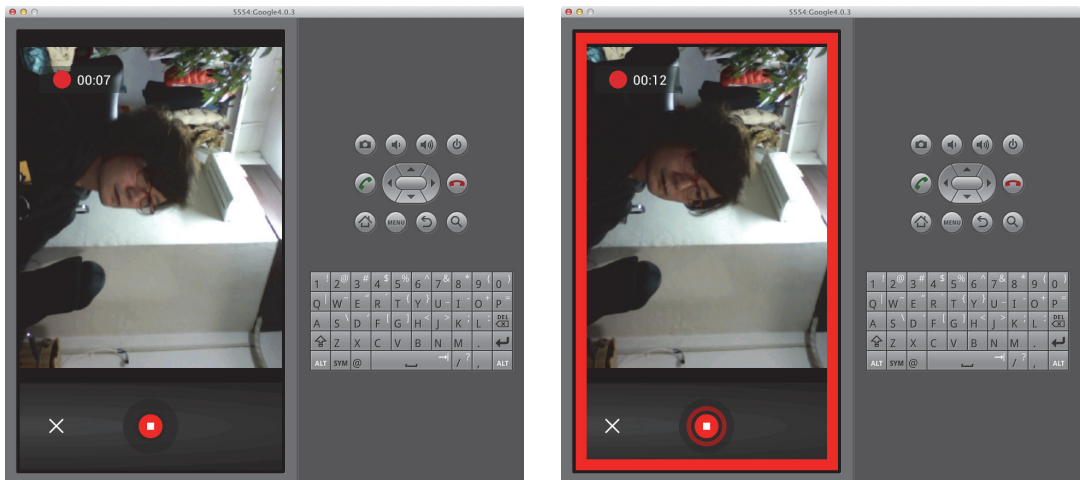
스텝 7

"Video" 버튼을 클릭하면 동영상을 녹화하는 화면으로 전환합니다.



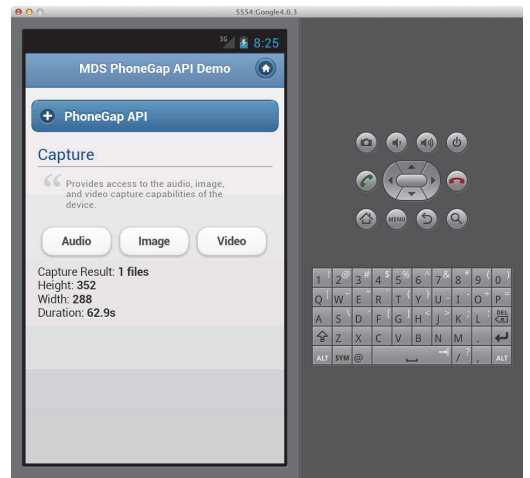
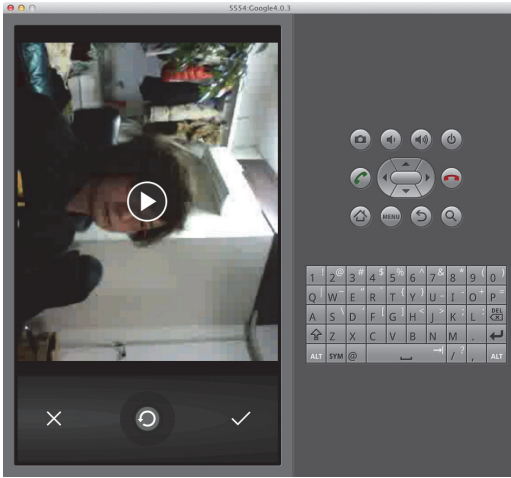
스텝 8

그림과 같이 녹화 버튼으로 녹화를 시작하고 종료 버튼으로 녹화를 마칩니다.



스텝 9

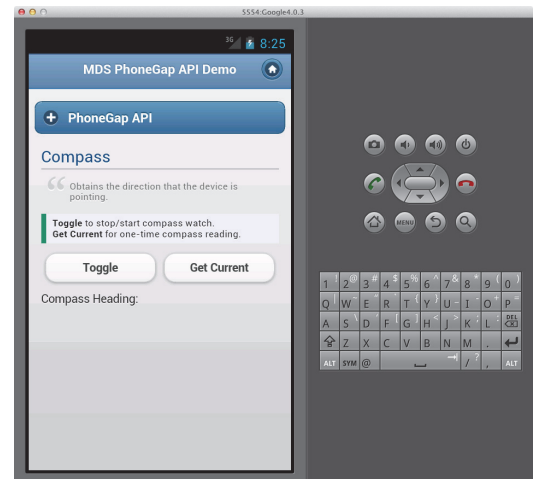
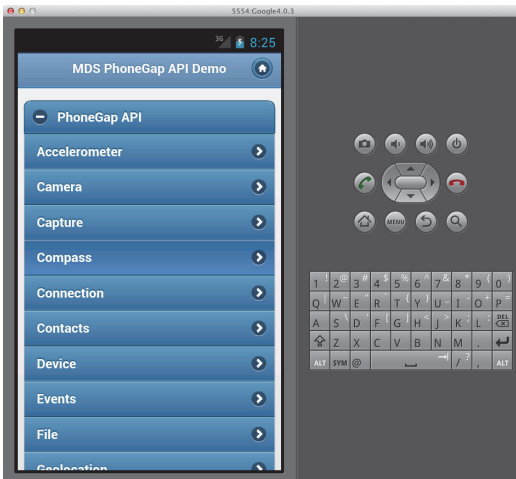
녹화된 동영상을 재생 또는 확인하고 "확인" 버튼을 클릭하면, 동영상의 크기 정보를 가져와 화면에 출력합니다.



가상기기 폰갭 데모 : 방위 센서(Compass)

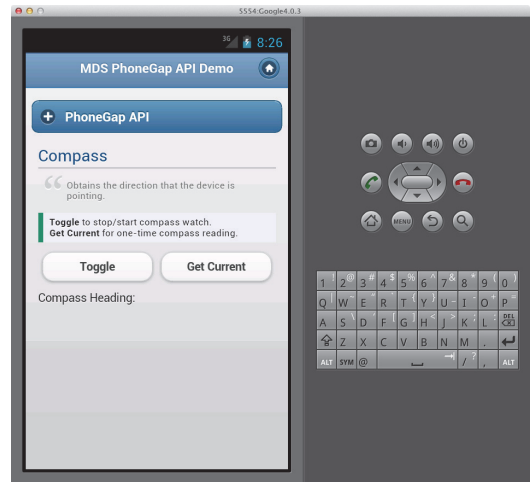
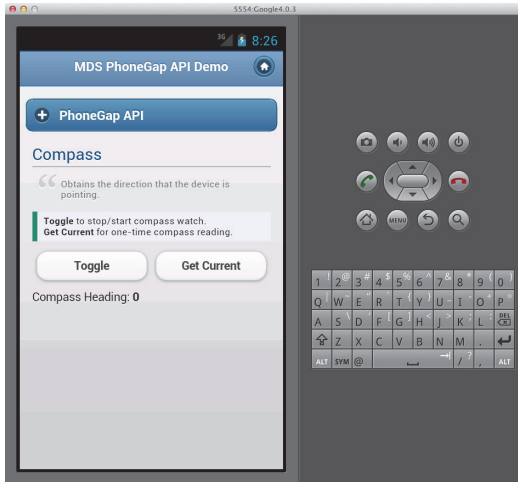
스텝 1

Compass는 방위 센서를 의미합니다. 단말기의 현재 방위 값을 가져오는 "Get Current" 버튼과 주기적으로 감지하는 "Toggle" 버튼이 제공됩니다.



스텝 2

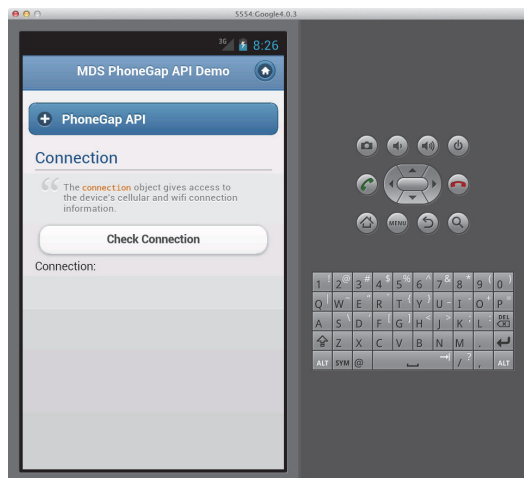
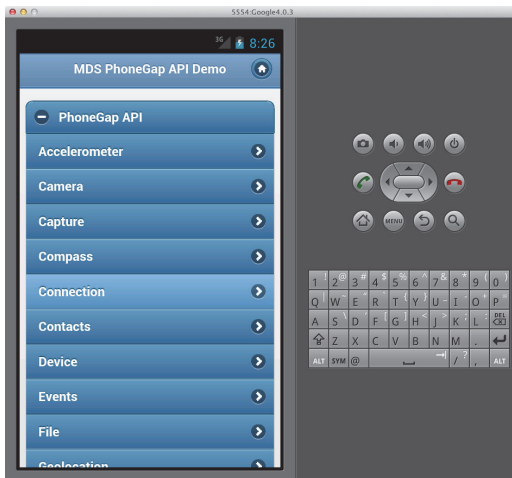
방위 센서는 불행히도 필자의 가상기기에서는 지원하지 않았습니다. 그러나 프로그램 상의 오류는 발생하지 않았습니다. 이 기능은 나중에 실물 단말기에서 실험하도록 하겠습니다.



가상기기 폰갭 데모 : 네트워크 연결(Connection)

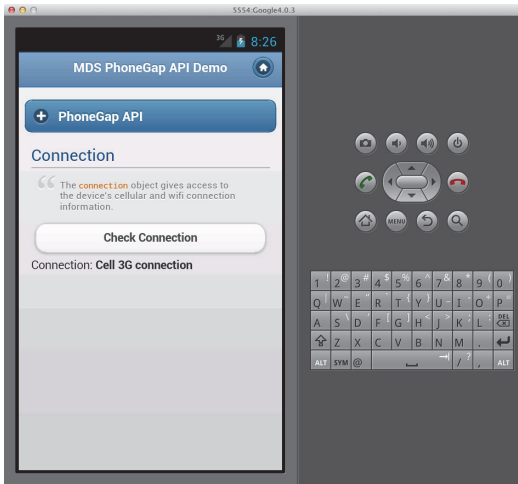
스텝 1

폰갭의 Connection API는 단순히 단말기가 사용하는 네트워크 정보를 출력하는데 그칩니다.



스텝 2

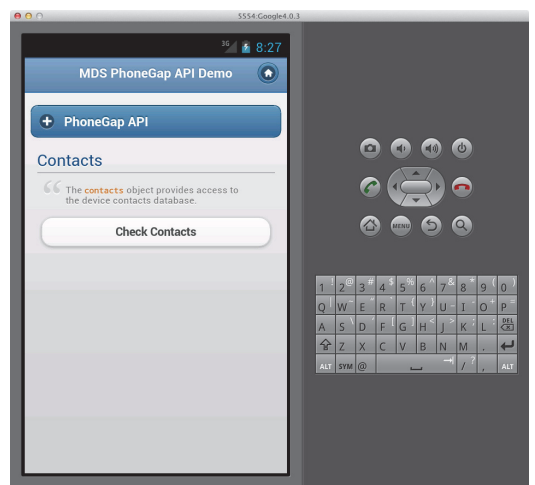
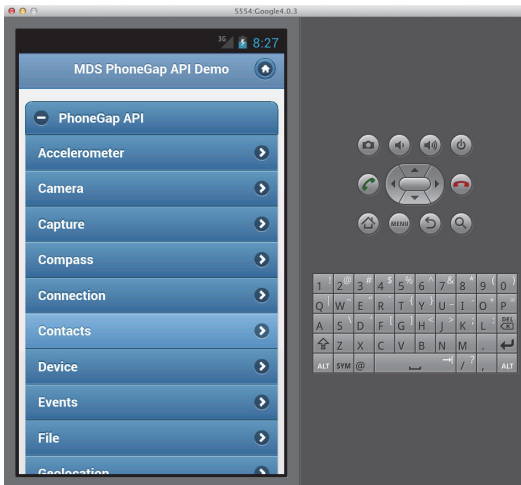
"Check Connection" 버튼을 클릭하면 그림과 같이 이 가상기기가 사용하고 있는 네트워크 정보를 출력해줍니다.



가상기기 폰갭 데모 : 연락처(Contacts)

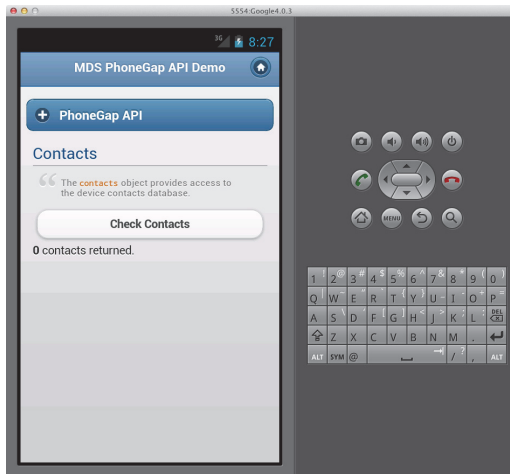
스텝 1

폰갭의 Contacts API는 단말기에 기본으로 탑재되어 있는 연락처(주소록) 데이터베이스와 연동하는 기능을 합니다.



스텝 2

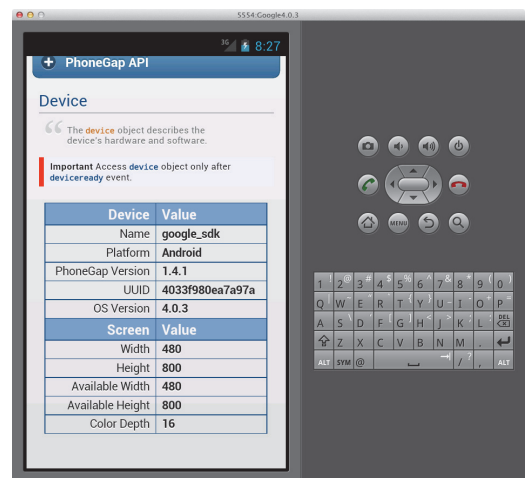
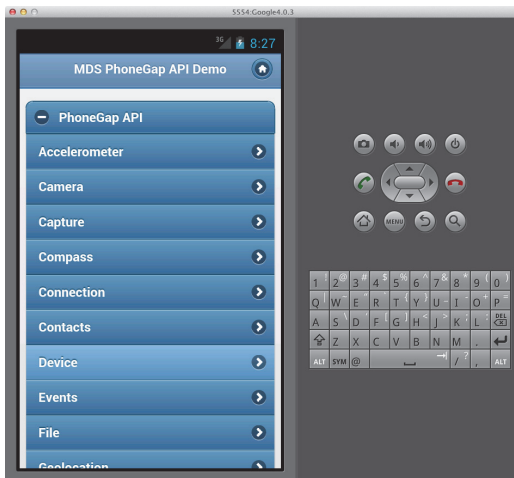
필자의 실험 가상기기는 주소록에 연락처를 등록한 적이 없기 때문에 그림과 같이 "Check Contacts"을 클릭하면 0개의 연락처가 있다는 메시지가 나타납니다. 이 부분도 실물 단말기에서 나중에 추가로 실험해보겠습니다.



가상기기 폰갭 데모 : 단말기 정보(Device)

스텝 1

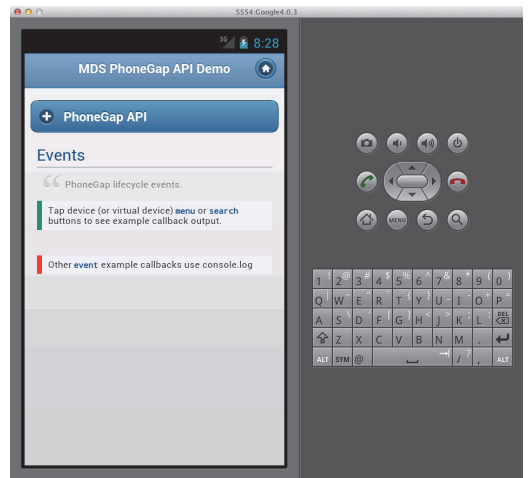
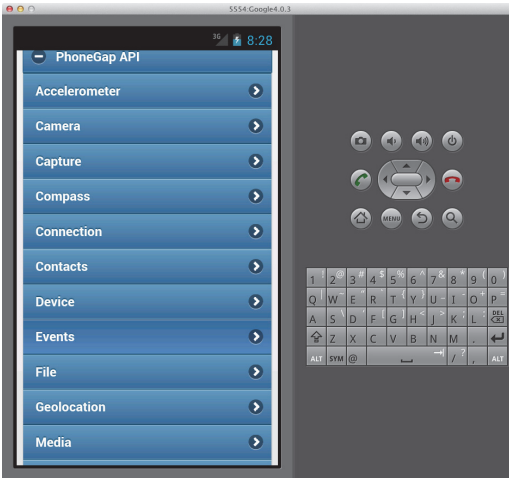
폰갭의 Device API는 단말기의 정보를 가져오는 기능을 합니다. 그림과 같이 가상기기의 정보를 가져와 화면에 출력하고 있습니다.



가상기기 폰갯 데모 : 이벤트(Event)

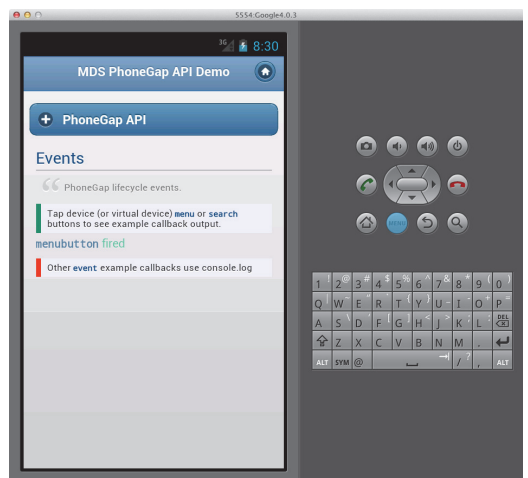
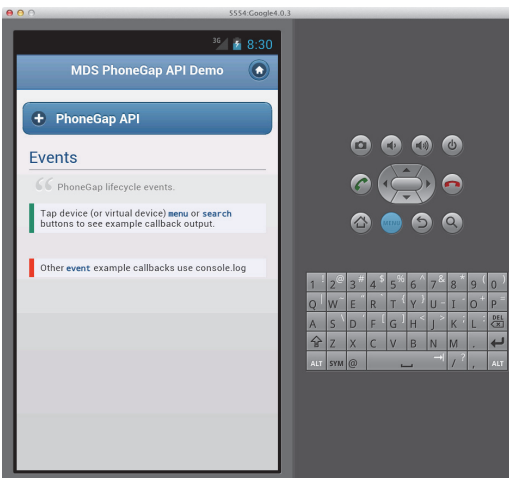
스텝 1

폰갯의 Event API는 폰갯의 라이프 사이클, 네트워크 온/오프, 각종 안드로이드 버튼, 배터리 관련 이벤트 등에 대한 솔루션을 제공합니다. 본 데모에서는 이러한 이벤트들 중 메뉴 버튼과 검색 버튼에 대한 이벤트를 사용할 수 있도록 지원하고 있습니다.



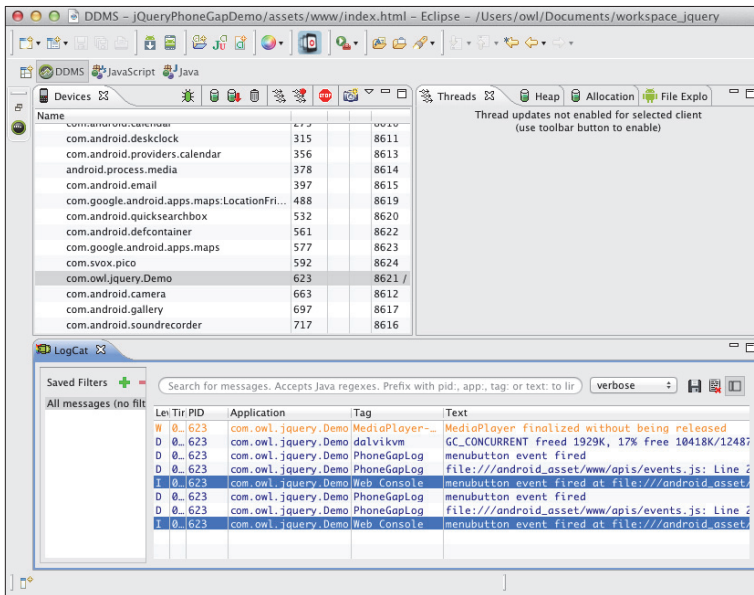
스텝 2

그림과 같이 단말기의 "MEMU" 버튼을 클릭하면 폰갯은 이를 감지하여 메뉴 버튼을 클릭했다는 메시지를 화면에 출력합니다.



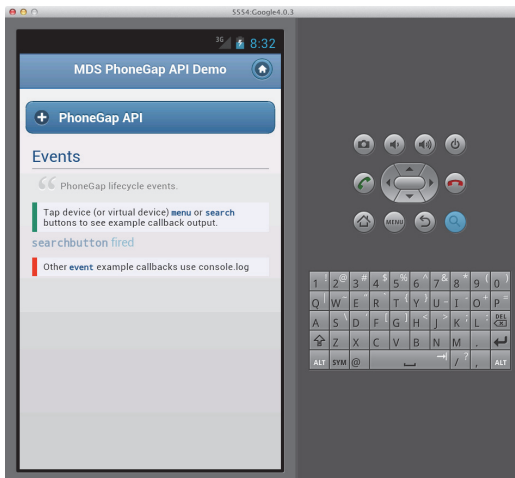
스텝 3

이 데모 앱에서는 버튼 이벤트를 감지하면 화면에도 안내문을 출력하지만 개발자를 위해 안드로이드 LogCat이라는 로그 시스템에도 로그를 출력하도록 프로그래밍되어 있습니다. 이클립스에서는 그림과 같이 DDMS Prespective 화면에서 LogCat 창을 볼 수 있는데 이 창을 보면 앞서 클릭했던 메뉴 버튼에 대한 로그가 나타나는 것을 확인할 수 있습니다.



스텝 4

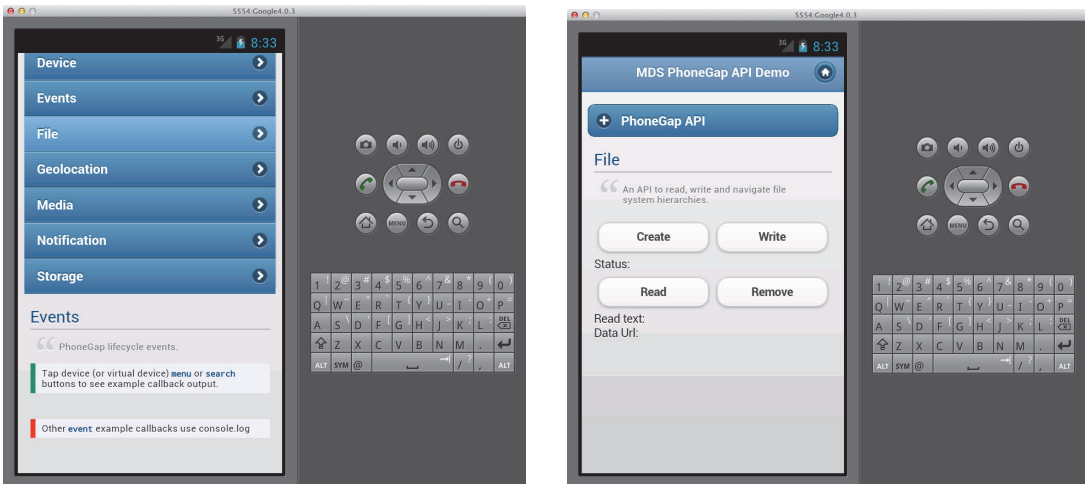
같은 방법으로 단말기의 검색 버튼을 클릭하면 화면에 검색 버튼을 실행했다는 안내문이 나타납니다.



가상기기 폰갭 데모 : 파일 시스템(File)

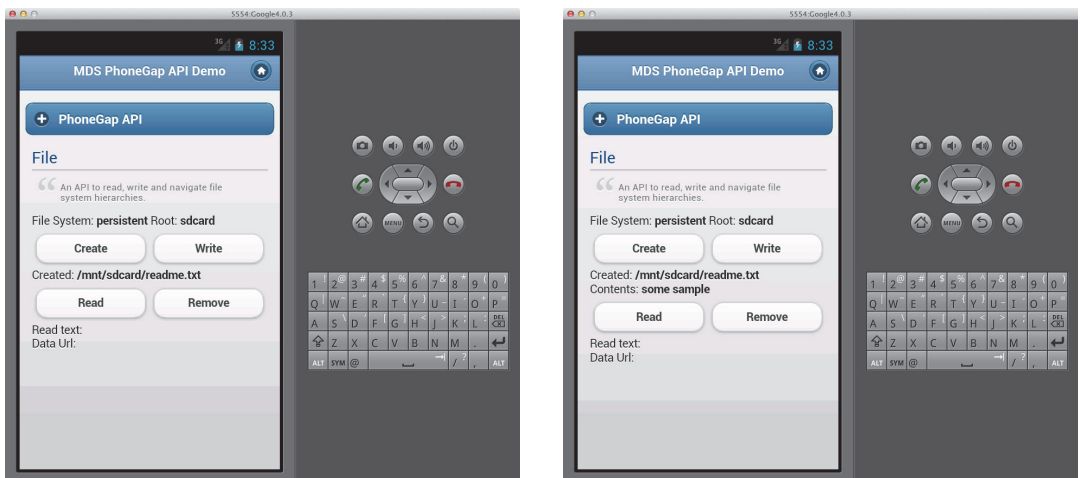
스텝 1

폰갭의 File API는 단말기에 파일을 생성하고 내용을 작성하고 수정/삭제하는 등의 기능을 지원하고, 폴더에 대해서도 읽기/생성/삭제 등의 기능을 지원할 뿐 아니라 서버에 파일을 업로드하는 등의 기능도 지원합니다. 이 데모 앱은 파일 시스템 기능 중 가장 기본적인 기능들만 실험할 수 있습니다.



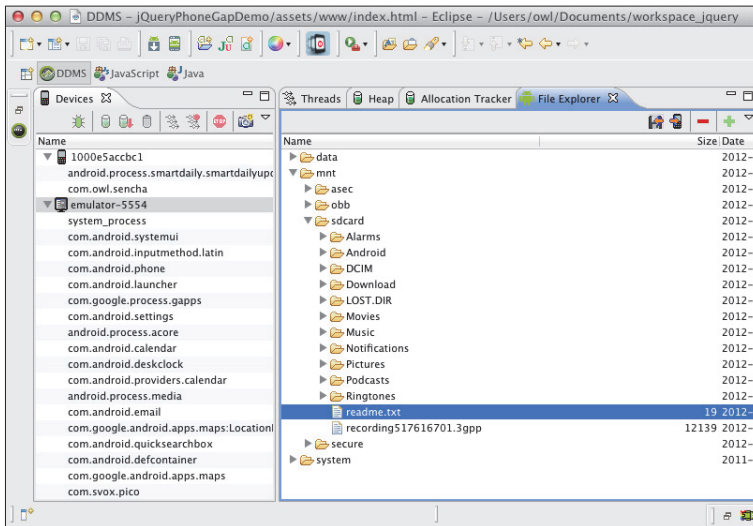
스텝 2

"Create" 버튼을 클릭하면 /mnt/scard/readme.txt 파일이 단말기에 생성되고, "Write" 버튼을 클릭하면 "some sample"이라는 안내문과 함께 프로그램에서 준비한 문자열이 생성된 파일의 내용으로 기록됩니다.



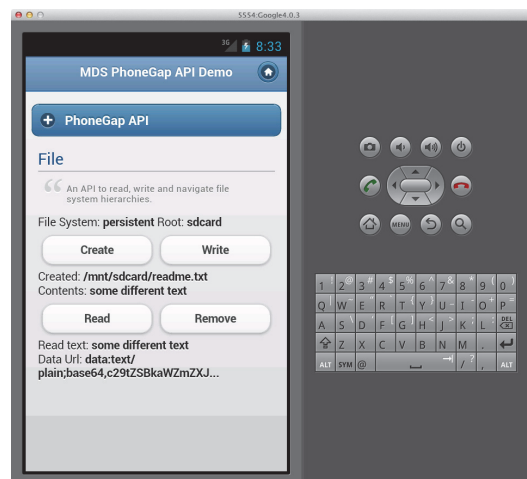
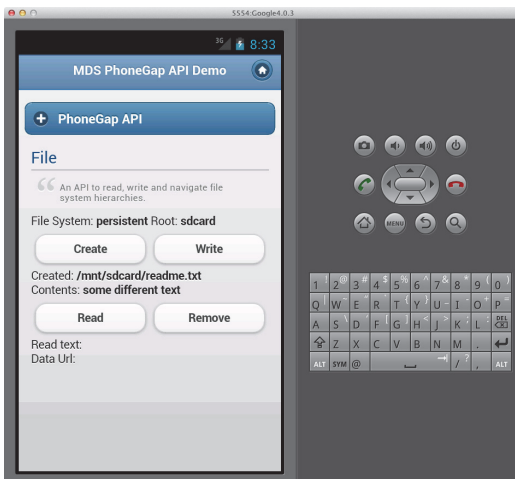
스텝 3

이클립스의 DDMS > Devices 창에서 실험중인 가상기기를 선택하고 File Explorer 창을 보면, 그림과 같이 /mnt/sdcard/readme.txt 파일이 생성되어 있으며 19 바이트의 크기로 내용이 담겨져 있는 것을 확인할 수 있습니다.



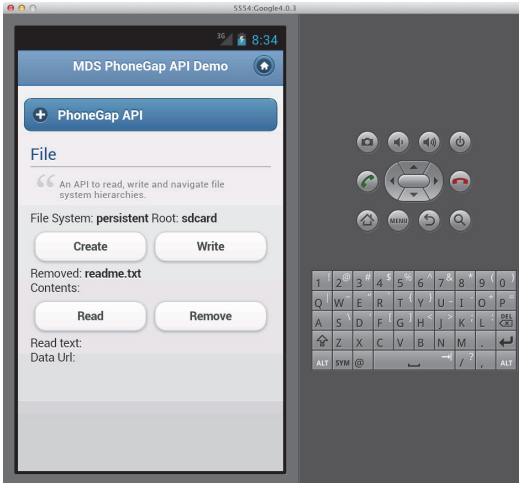
스텝 4

"Write" 버튼을 또 한 번 누르면 "some different text"라는 안내문과 함께 내용을 기록했다는 안내가 나타나고, "Read" 버튼을 클릭하면 파일에 있는 내용을 읽어와 그림과 같이 화면에 출력합니다.



스텝 5

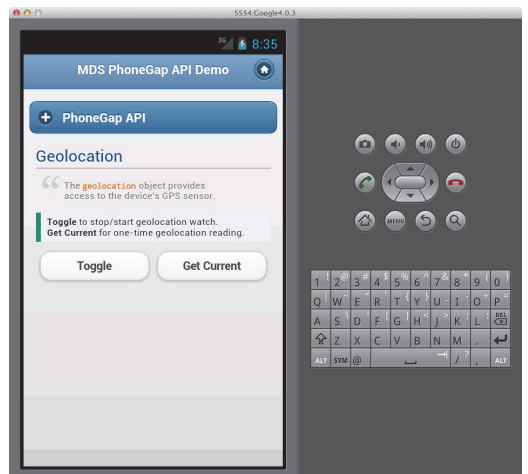
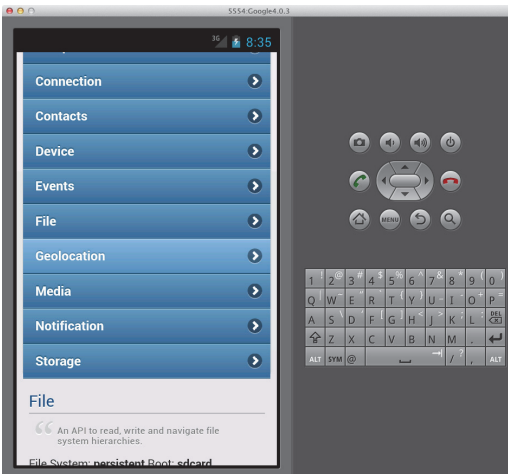
"Remove" 버튼을 클릭하면 그림과 같이 파일을 삭제했다는 안내문을 볼 수 있습니다. DDMS를 확인해보면 앞서 확인했던 파일이 사라졌을 것입니다.

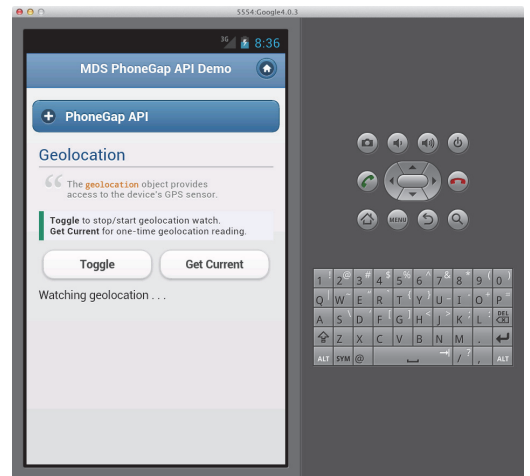
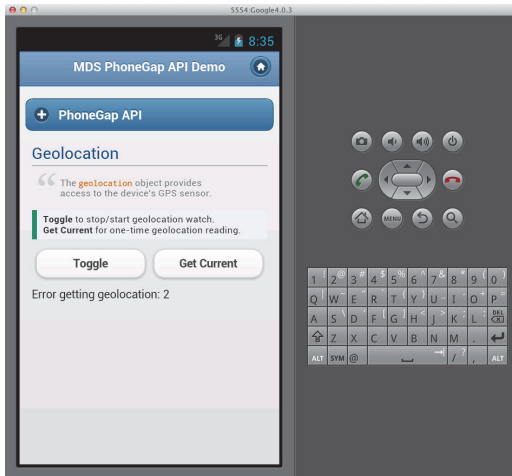


가상기기 폰갭 데모 : GPS 위치 감지 센서(Geolocation)

스텝 1

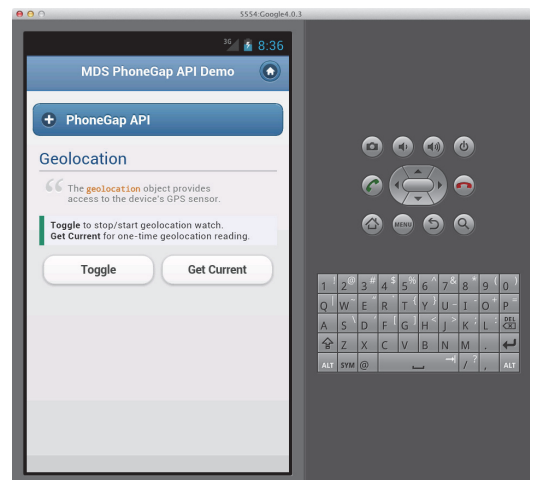
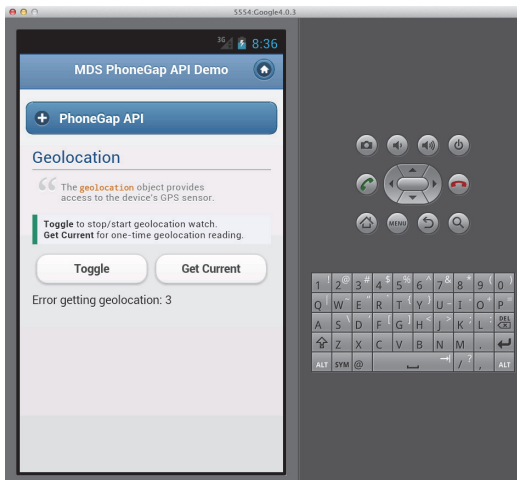
폰갭의 Geolocation API는 단말기의 GPS 좌표 정보를 입수하는 기능을 합니다. 이 데모 앱에서는 Geolocation으로 구한 GPS 좌표로 화면에 구글 지도를 출력하도록 구성하고 있습니다. "Get Current" 버튼을 클릭하면 현재 단말기의 위치 정보를 1회만 가져오고, "Toggle" 버튼을 클릭하면 주기적으로 위치 변동을 감지합니다.





스텝 2

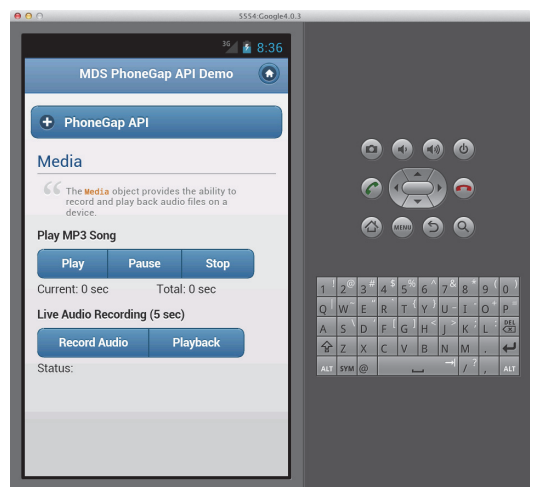
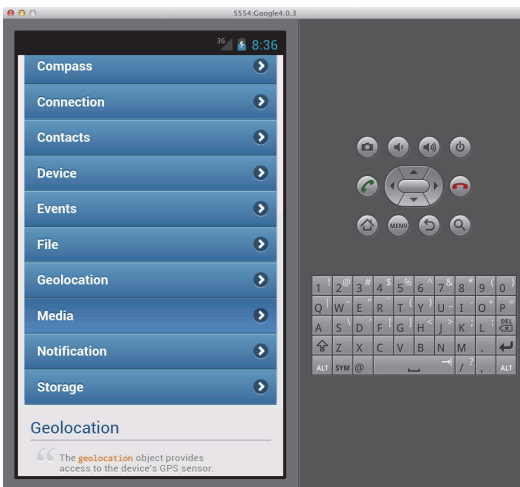
그림에서 보여주는 바와 같이 필자의 가상기기에서는 위치 정보를 가져오지 못했습니다. 이 경우 그림과 같은 안내문을 출력합니다. 가상기기에서도 이 실험을 어느 정도 계속 진행할 수 있지만, 실물 단말기가 있다면 불필요한 실험일 수밖에 없습니다. 나중에 실물 단말기에서 이 부분을 실험해보도록 하겠습니다.



가상기기 폰갭 데모 : 미디어(Media)

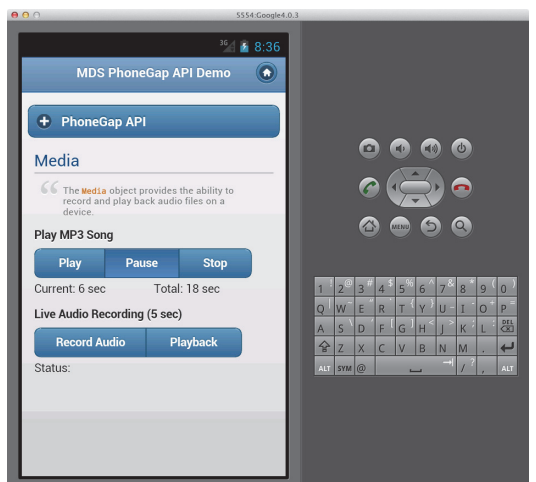
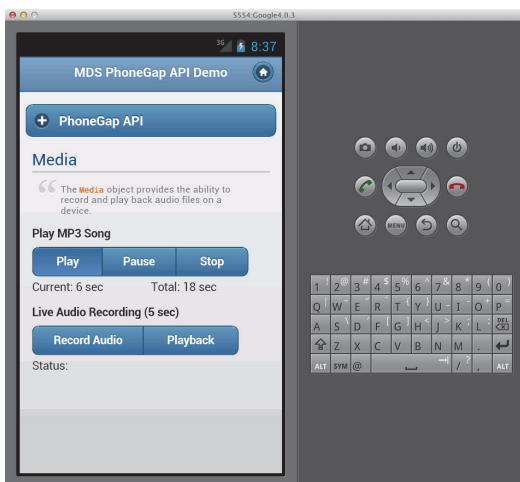
스텝 1

폰갭의 Media API는 오디오를 재생하는 기능을 지원합니다. 폰갭의 공식 매뉴얼에 따르면 초창기에는 폰갭의 원래 목표대로 네이티브 앱 프로그램과의 연동으로 미디어 플레이어를 구현했으나, 앞으로는 HTML5에서 지원하는 비디오/오디오 기능이 더 대중적이고 편리하기 때문에 HTML5를 이용하는 방식으로 업그레이드한다고 합니다.



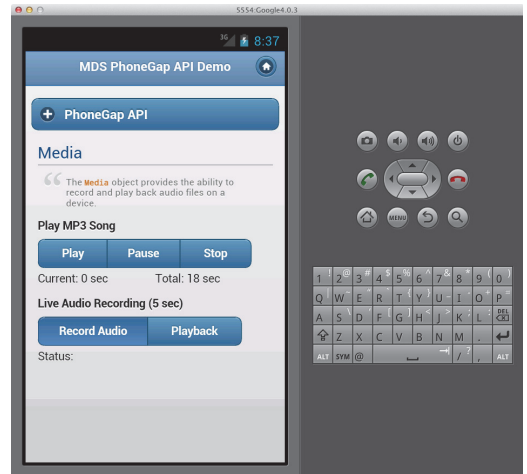
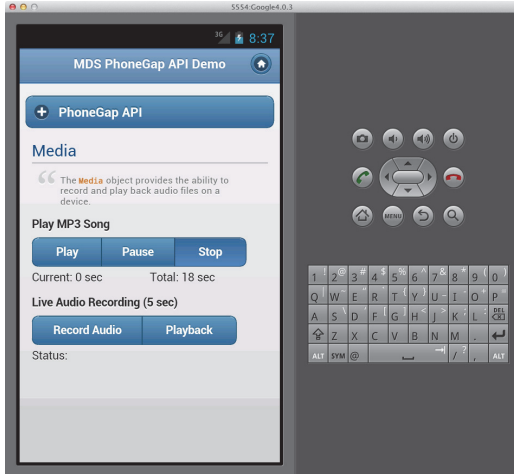
스텝 2

"Play" 버튼을 클릭하면 프로그램에서 지정한 음원이 재생됩니다. "Pause" 버튼을 클릭하면 재생되던 음원이 중지됩니다.



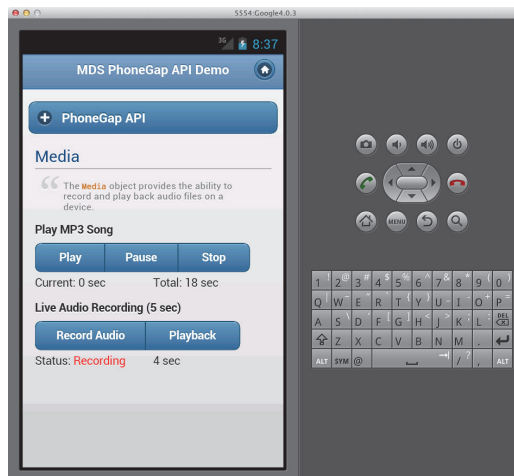
스텝 3

"Stop" 버튼을 클릭하면 재생 중이던 음원이 종료됩니다. 이 외에도 "Record Audio" 버튼을 클릭하면 녹음이 시작됩니다.



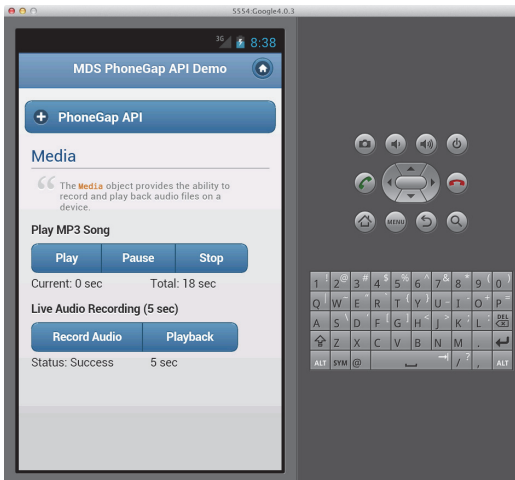
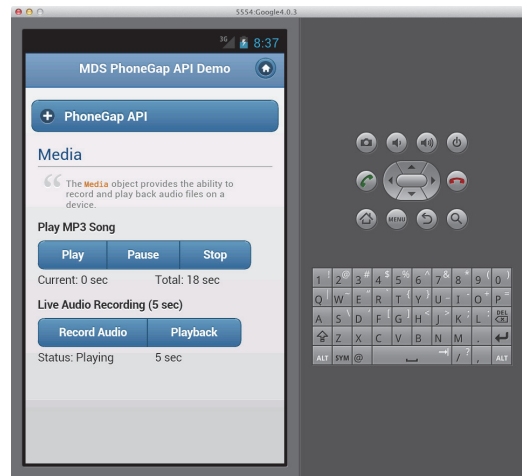
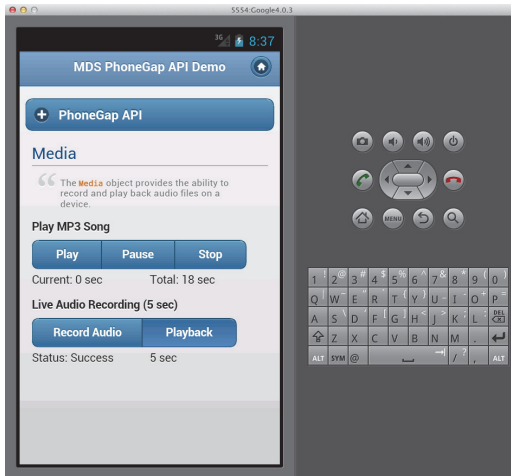
스텝 4

녹음이 진행 중일 때는 "Recording"이라는 상태가 표시되면서 녹음 진행 시간이 흐르고, 녹음이 종료되면 "Success"라는 안내문이 나옵니다.



스텝 5

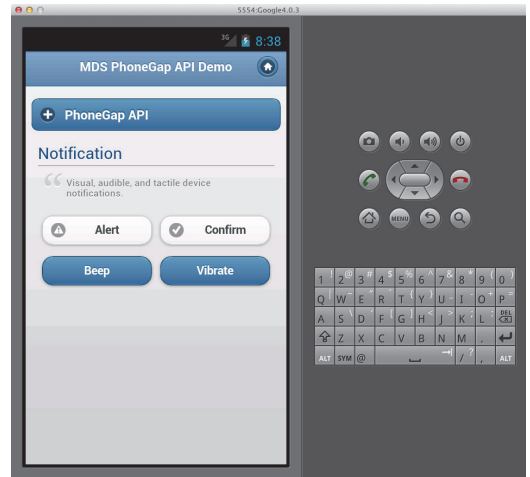
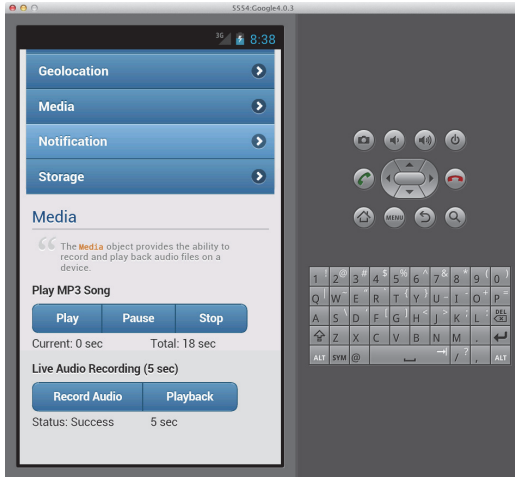
"Playback" 버튼을 클릭하면 녹음된 음원이 재생되면서 "Playing"이라는 안내문이 나타납니다.



가상기기 폰갭 데모 : 알림(Notification)

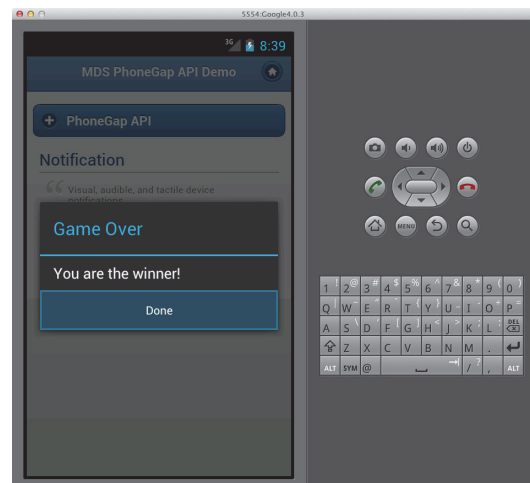
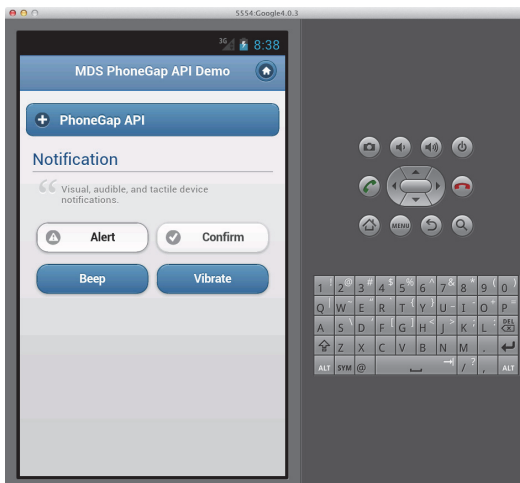
스텝 1

폰갭의 Notification API는 다양한 알림 기능에 대한 솔루션을 제공합니다. 경고 대화상자, 확인 대화상자, 알림 벨소리, 진동이 주요 기능입니다.



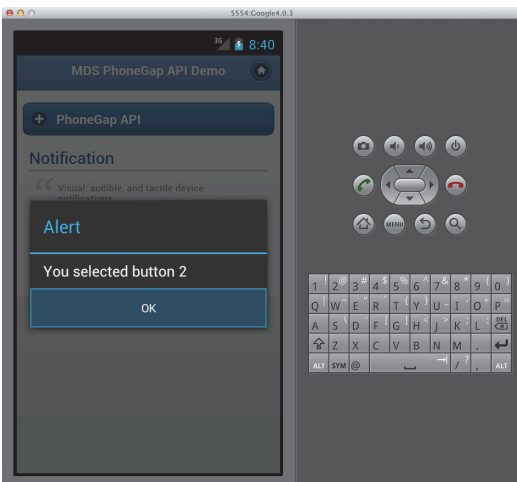
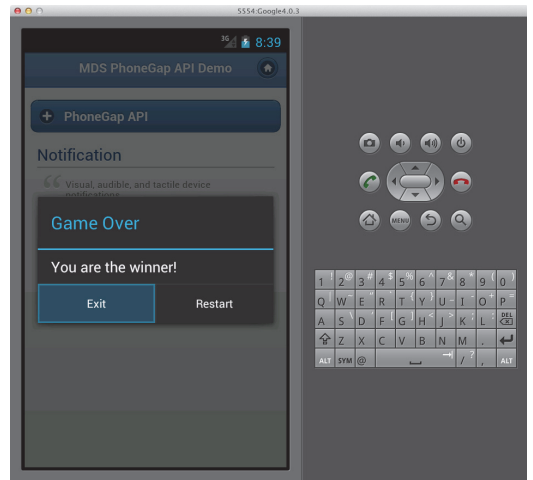
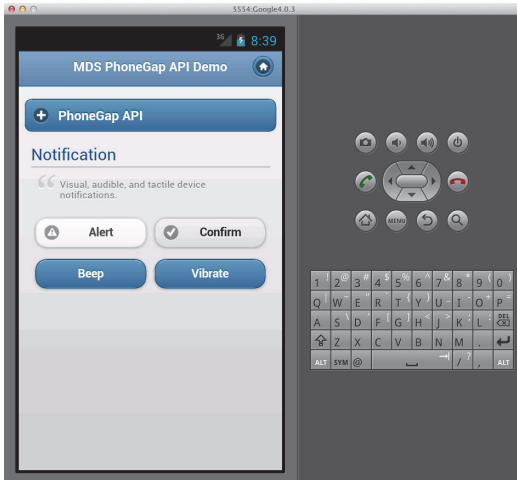
스텝 2

"Alert" 버튼을 클릭하면 그림과 같이 단순 알림 또는 경고 대화상자가 나타납니다. "Done" 버튼을 클릭하면 대화상자가 닫힙니다.



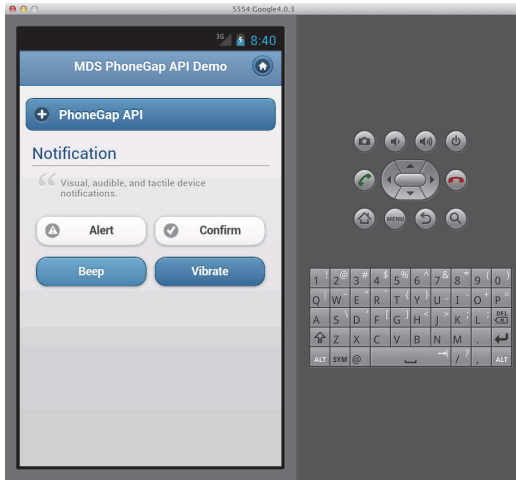
스텝 3

"Confirm" 버튼을 클릭하면 두개의 버튼이 있는 확인형 대화상자가 나타납니다. "Exit" 버튼을 클릭하면 두 번째 버튼을 선택했다는 대화상자가 나타납니다. "OK" 버튼을 클릭하여 대화상자를 닫았습니다.



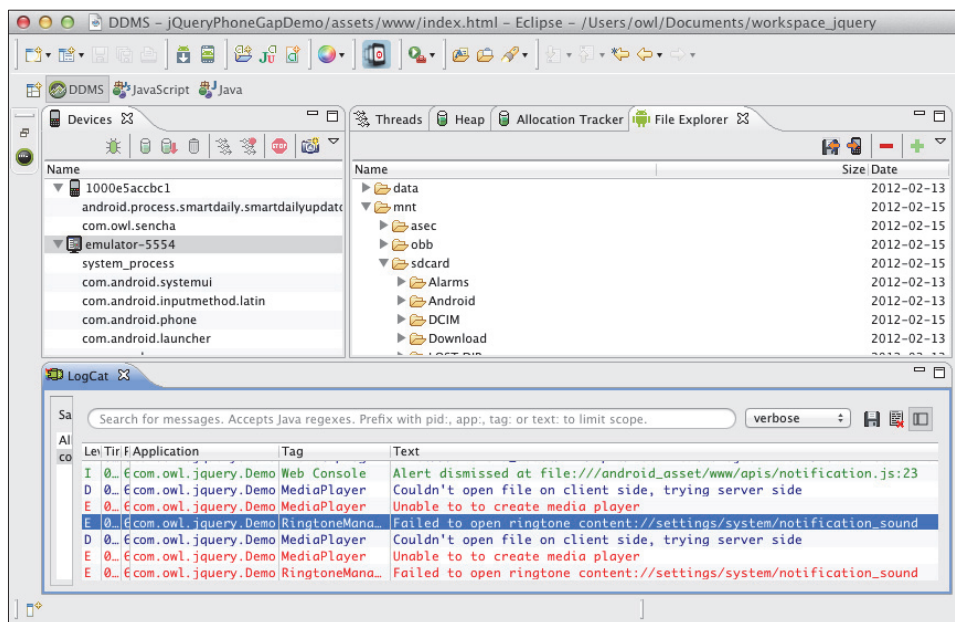
스텝 4

"Beep" 버튼을 클릭하면 알람 벨소리가 들리지 않았습니다. 이는 가상기기에 기본 벨소리 파일이 없기 때문입니다.



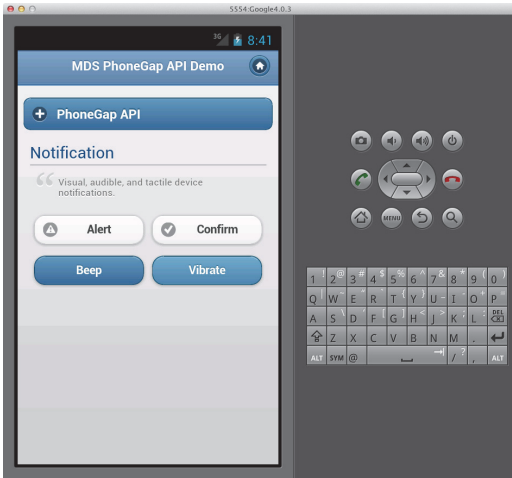
스텝 5

DDMS > LogCat에서 로그를 확인해보면 그림과 같이 벨소리(ringtone) 파일을 열 수 없다는 오류 안내문을 확인할 수 있습니다.



스텝 6

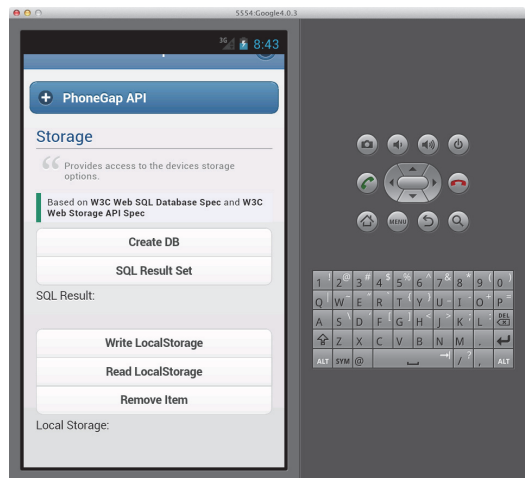
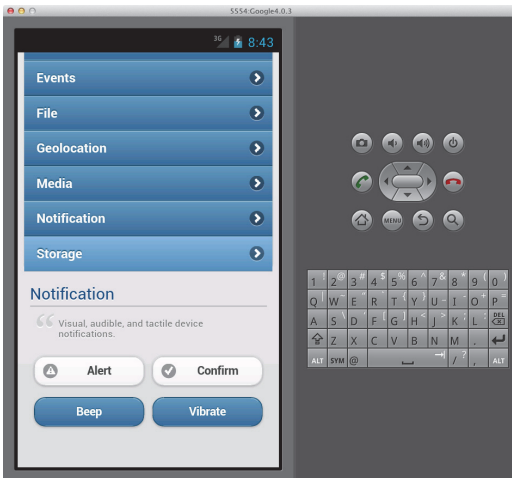
"Vibrate" 버튼 역시 가상기기에서 진동 기능을 지원하지 않기 때문에 실험할 수 없습니다. 이 기능은 나중에 실물 단말기에서 실험하도록 하겠습니다.



가상기기 폰갭 데모 : 로컬 데이터베이스(Storage)

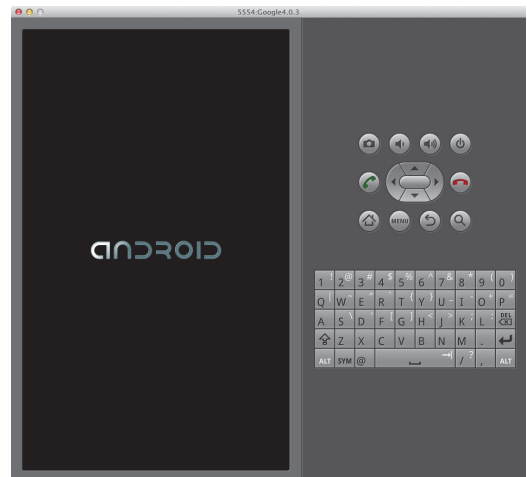
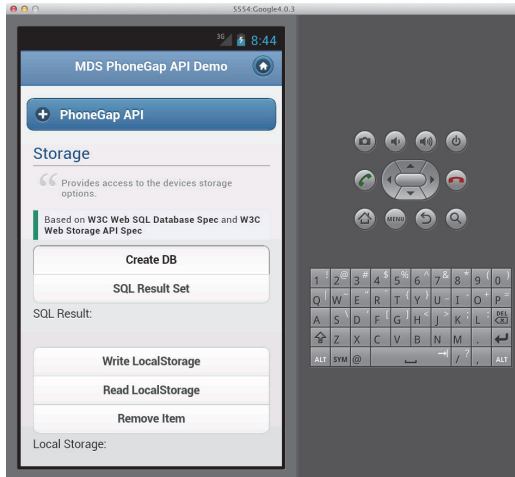
스텝 1

폰갭의 Storage API는 HTML5에서 지원하는 로컬 데이터베이스 기능을 쉽게 사용할 수 있도록 지원합니다. 폰갭의 Storage 기능은 W3C Web SQL Database를 이용하는 방식과 W3C Web Storage를 이용하는 방식이 있습니다. W3C Web SQL Database는 SQL 문을 이용하여 입출력하는 일반적인 데이터베이스를 말하고, W3C Web Storage는 Key-Value 형식의 단순한 환경 변수를 사용하는 방식을 의미합니다.



스텝 2

하지만 불행히도 필자의 가상기기는 "Create DB" 버튼을 클릭하자 다운되어 버렸습니다. 본서의 목적이 폰갭이 아니고 폰갭이 버전업되면서 이런 문제는 알아서 해결될 가능성이 높으며, 실물 단말기에서 작동하는 것이 최종 목표이기 때문에 이 문제는 잠시 후 다시 살펴봅니다.

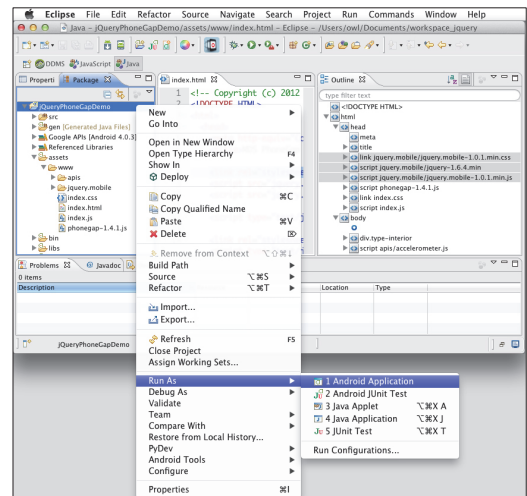


안드로이드 실물 단말기에서 실험하기

가상기기에서 못다 한 실험을 실물 단말기에서 실험해봅니다. 가상기기에서는 4.0.X 아이스크림 샌드위치에서 실험했지만, 현재 대개 2.2.X 또는 2.3.X 버전의 단말기를 사용하므로 갤럭시탭 버전 2.3.X에서 실험하기 위해 안드로이드 프로젝트의 참조 라이브러리를 이 버전에 맞게 변경합니다.

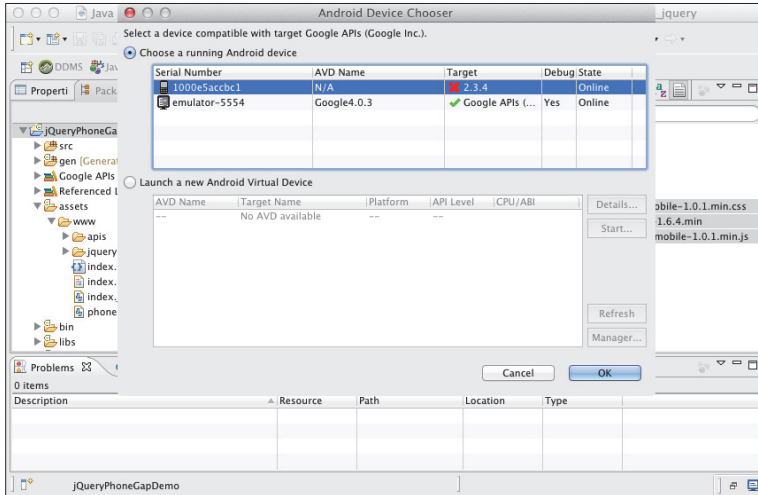
스텝 1

실물 단말기를 USB로 개발 컴퓨터에 연결하고, 그림과 같이 프로젝트를 선택한 후, “콘텍스트 메뉴 > Run As > Android Application” 메뉴를 실행해봅니다.



스텝 2

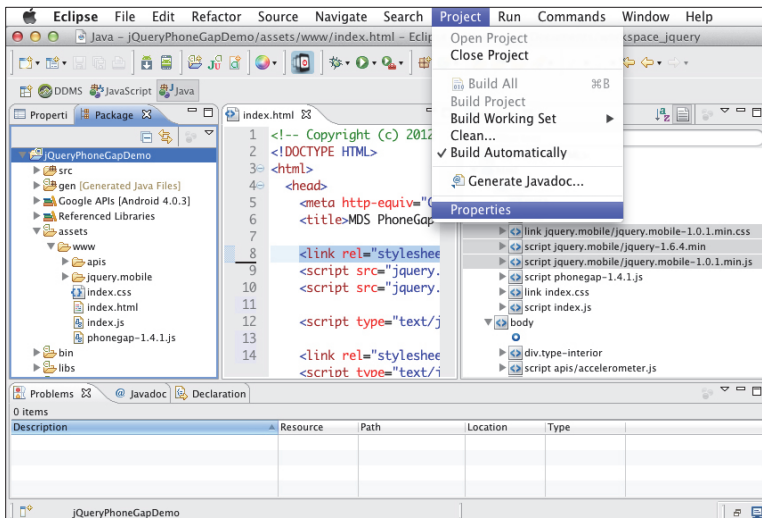
그림과 같이 개발 컴퓨터에 연결된 실물 단말기를 확인할 수 있으나 이 단말기의 버전이 2.3.4 이므로 4.0.3 버전으로 만든 이 프로젝트를 실행할 수 없습니다.



안드로이드 SDK 버전 변경 : 구 버전으로 다운그레이드

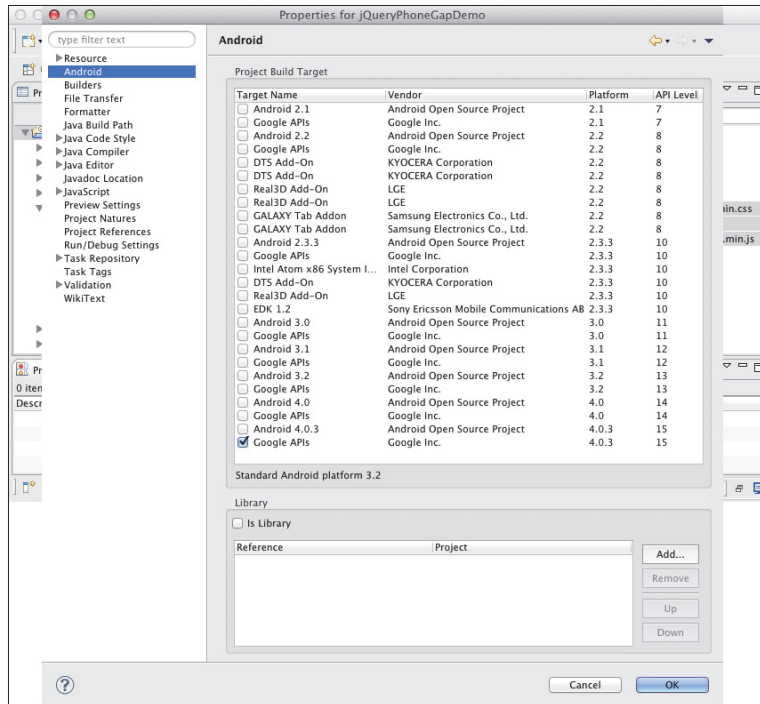
스텝 1

프로젝트를 선택하고 “Project > Properties” 메뉴를 실행하여 프로젝트 속성 창을 호출합니다.



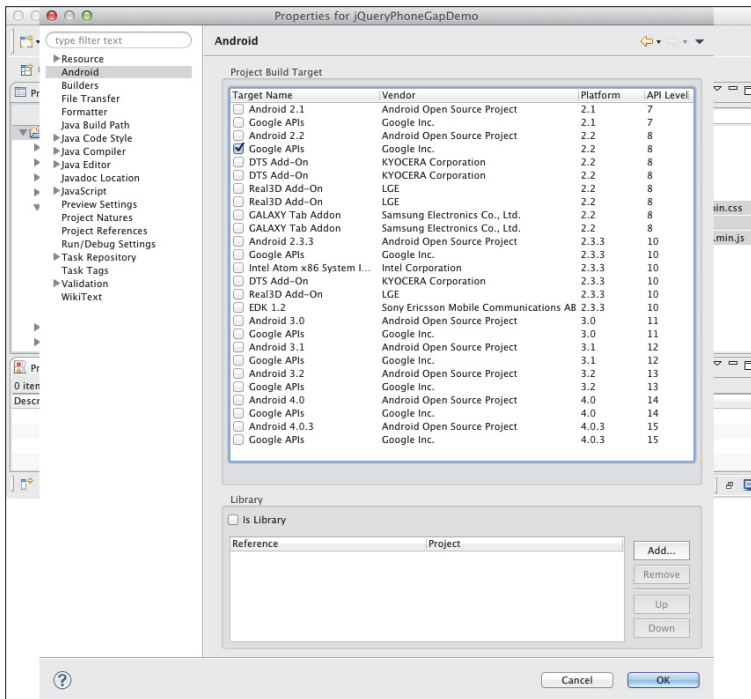
스텝 2

“Android > Project Build Target”을 보면 그림과 같이 Google APIs 4.0.3, API Level 15로 설정되어 있습니다.



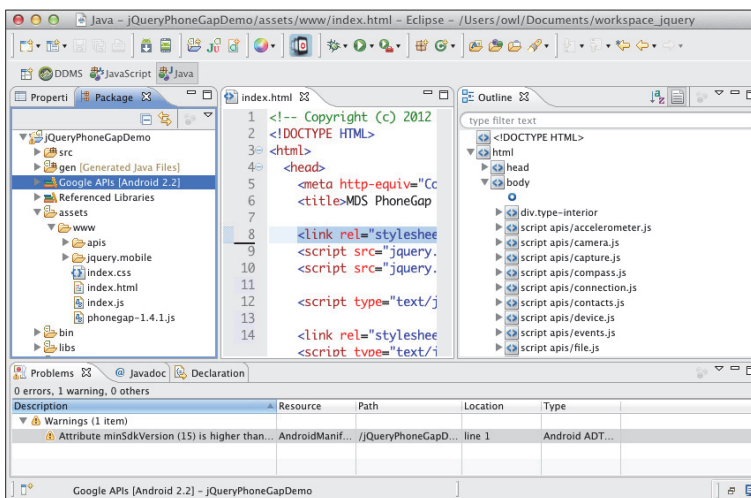
스텝 3

위의 설정을 그림과 같이 Google APIs 2.2, API Level 8로 변경했습니다. 갤럭시의 경우 대부분 23X 버전까지 업그레이드되지만, LG U+와 같은 구형 단말기의 경우 22X 버전까지만 업그레이드됩니다. 따라서 개발 당시 대중들이 사용하는 최하위 버전을 감안하여 2.2 버전을 기준으로 실험해 보기로 했습니다. "OK" 버튼을 클릭하여 이 프로젝트의 안드로이드 API 버전을 변경 적용합니다.



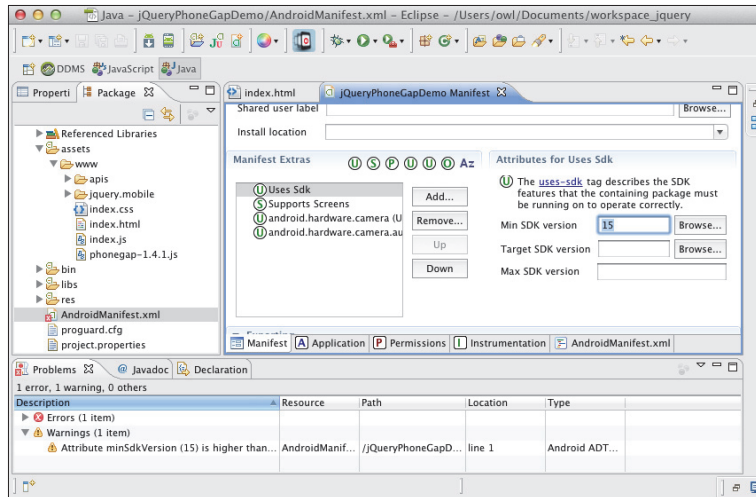
스텝 4

그림과 같이 Google APIs 2.2를 프로젝트에서 참조하는 것을 확인할 수 있습니다. 하지만 Problems 창을 보면 minSdkVersion 등에 대한 경고문이 나타납니다. 이와 같이 안드로이드 버전을 변경할 때는 간단하게 수동으로 수정해야 할 사항이 있습니다.



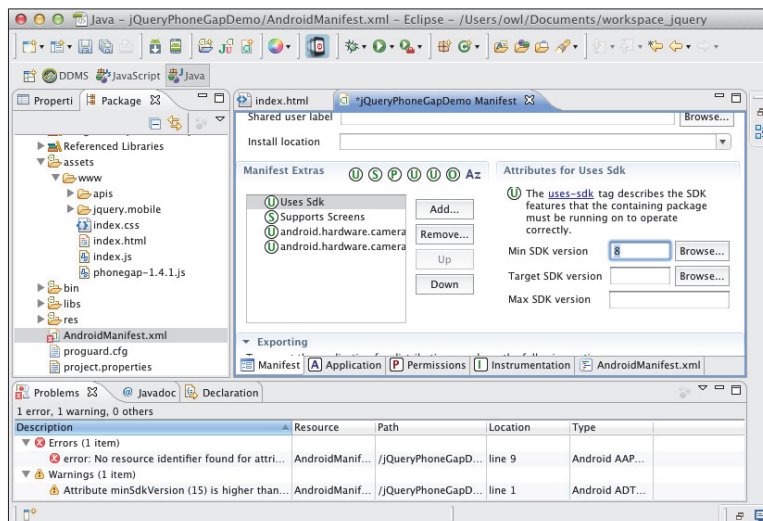
스텝 5

AndroidManifest.xml 파일을 열고, Manifest 탭에서 “Uses Sdk > Min SDK version” 설정을 확인해보면 이전에 사용했던 API Level 15로 설정되어 있는 것을 알 수 있습니다.



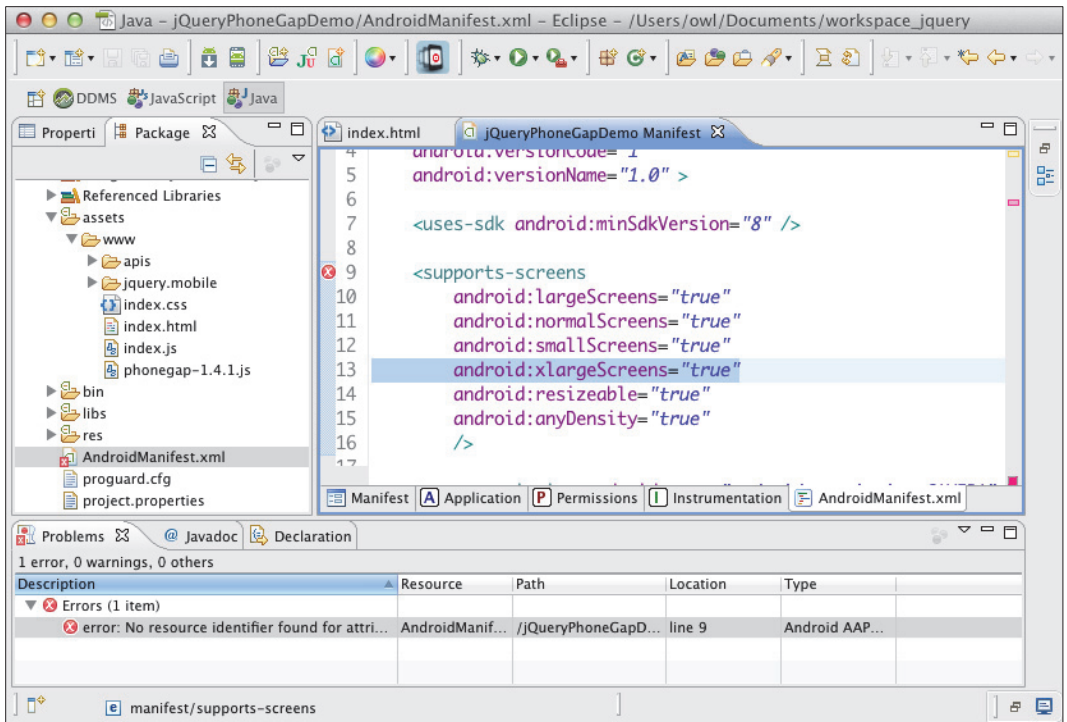
스텝 6

그림과 같이 Min SDK version을 Android 2.2에 해당하는 API Level 8로 수정합니다.



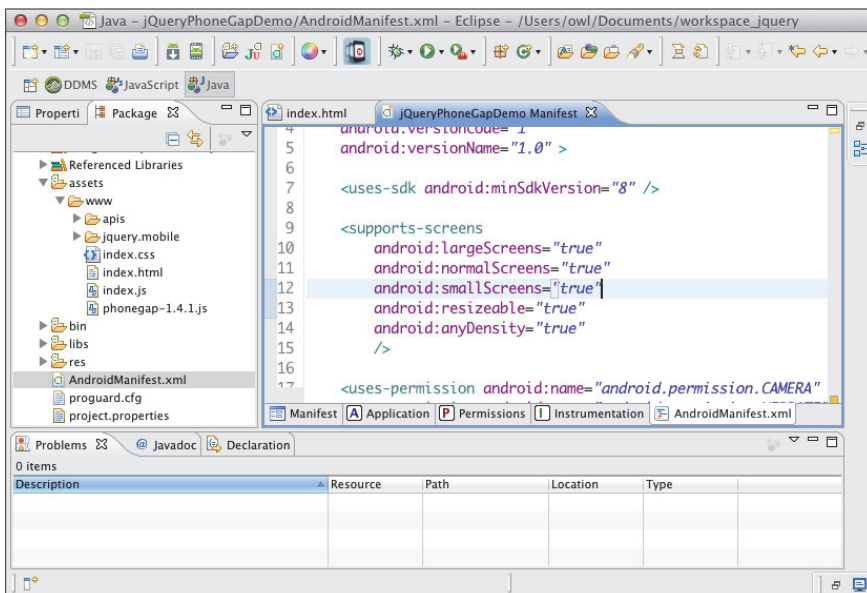
스텝 7

AndroidManifest.xml의 소스를 열고 그림과 같이 android:xlargeScreen 속성을 삭제합니다. 구형 단말기의 경우 큰 화면을 지원하지 않기 때문에 이와 같은 조치가 필요합니다.



스텝 8

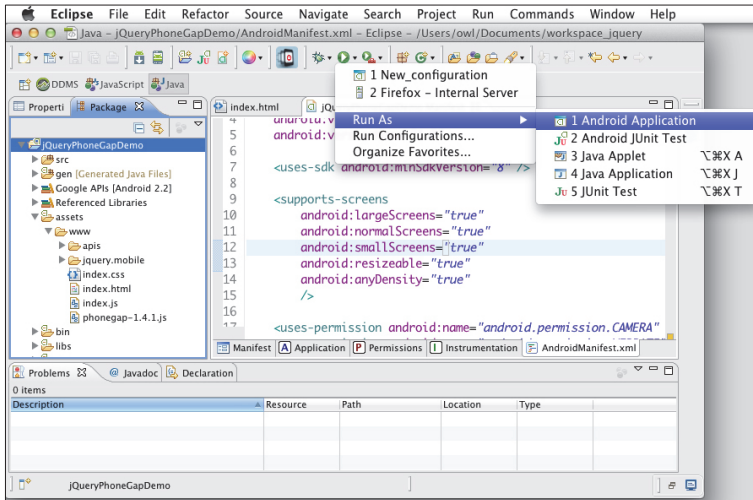
AndroidManifest.xml 파일을 저장하면 이클립스가 자동으로 재컴파일을 실행하고 더 이상 오류가 나타나지 않습니다.



실물 단말기에서 jQuery Mobile 폰갭 데모 앱 실행하기

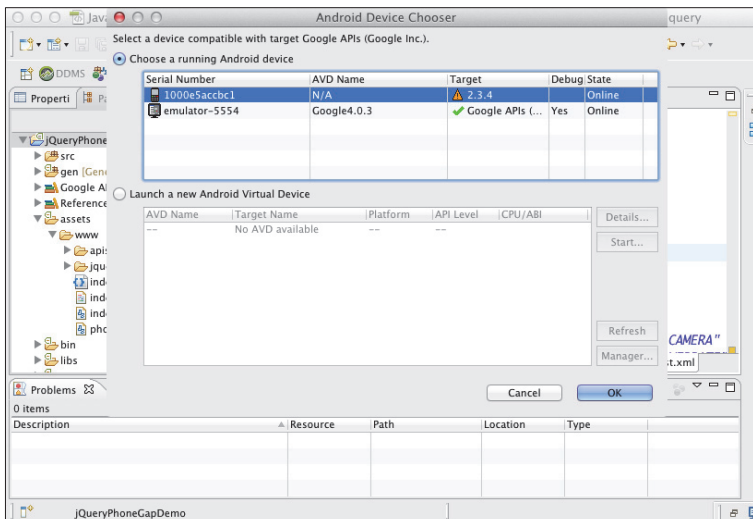
스텝 1

프로젝트를 선택하고 “Run As > Android Application” 메뉴를 실행합니다.



스텝 2

Android Device Chooser 창에서 실물 단말기를 선택하고 “OK” 버튼을 클릭하면 이클립스가 실물 단말기에 이 데모 앱을 설치합니다.

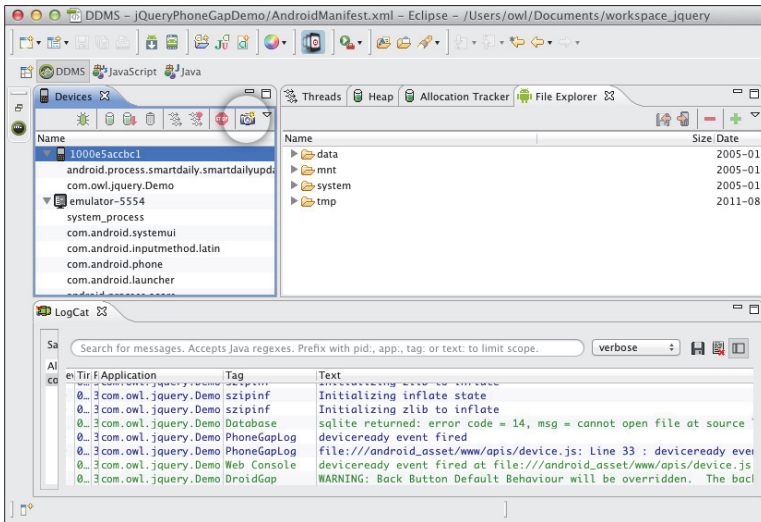


DDMS의 실물 단말기 화면 캡처

실물 단말기에서 실험할 경우 다음과 같이 DDMS를 이용하여 화면을 캡처하는 방법이 있습니다.

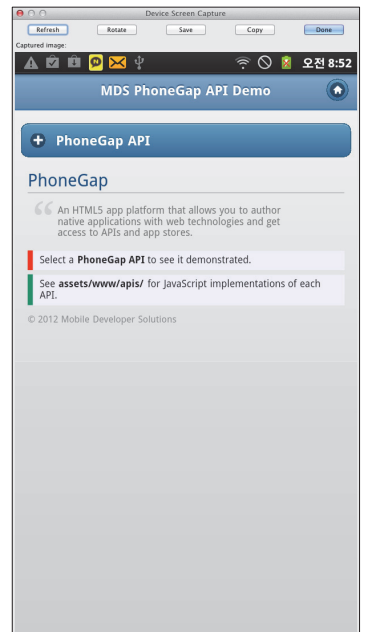
스텝 1

그림과 같이 DDMS 화면에서 실물 단말기를 선택하고 "Screen Capture" 아이콘 버튼을 클릭합니다.



스텝 2

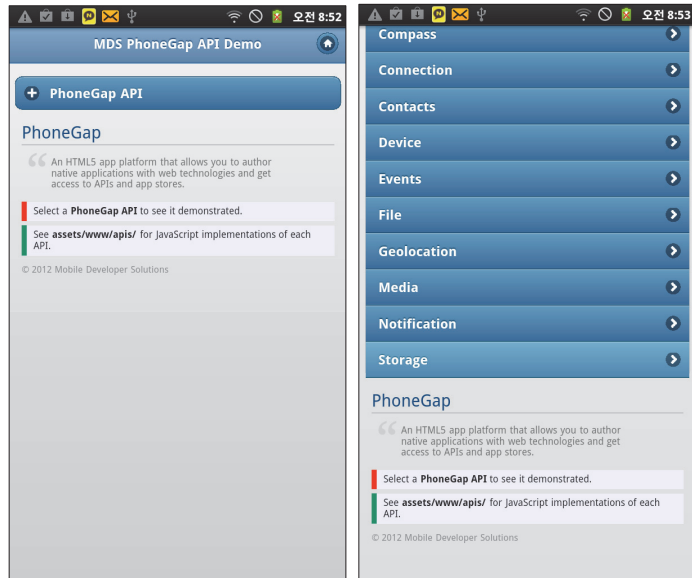
그림과 같이 Device Screen Capture 창이 나타나고 실물 단말기의 화면을 캡처합니다. "Refresh" 버튼으로 현재 실물 단말기에 있는 화면을 캡처 받을 수 있습니다. 이 기능은 기획자, 디자이너, 프로그래머 등이 한자리에 모여 스마트앱에 대한 논의를 한다거나 스토리 보드를 제작하는 등의 일련의 개발 과정에서 매우 유용하게 활용됩니다.



실물 단말기 폰갭 데모 : 로컬 데이터베이스(Storage)

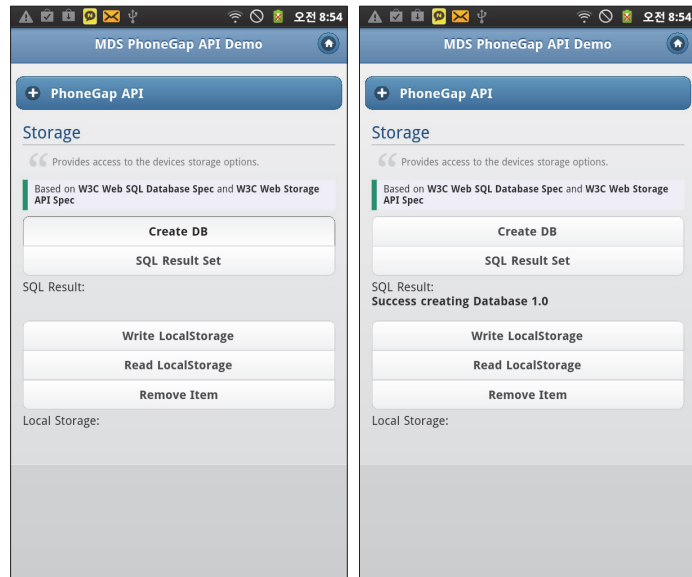
스텝 1

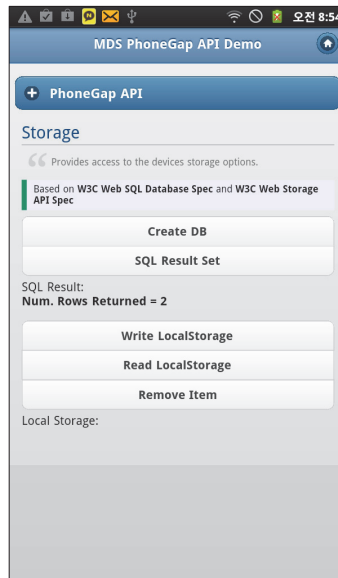
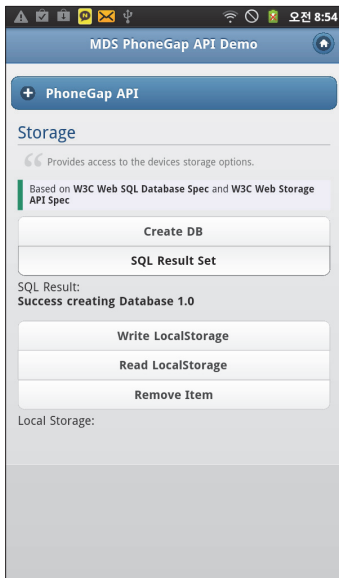
실물 단말기에서도 jQuery Mobile로 포장되고 프로그래밍된 폰갭 데모 앱이 그림과 같이 잘 나타납니다.



스텝 2

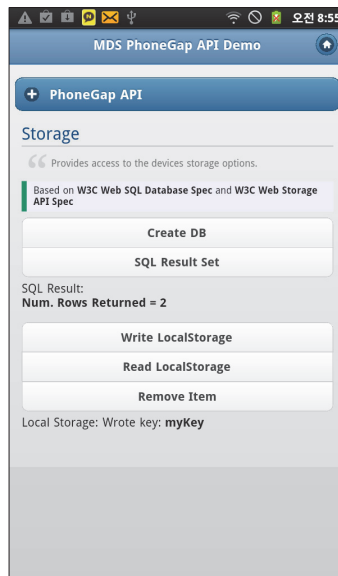
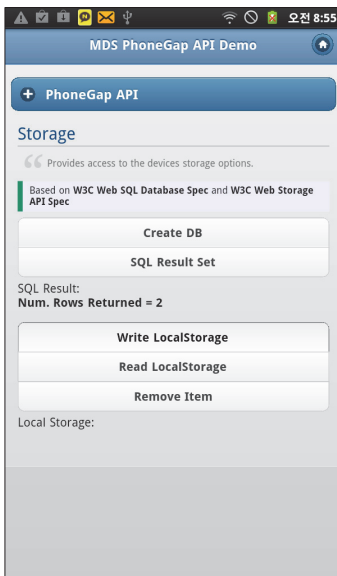
Storage 화면을 열고 가상기기에서 중단된 실험을 합니다. "Create DB" 버튼을 클릭하면 프로그래밍된 데이터베이스를 생성하고, 그림과 같이 "Success creating Database 1.0"이라는 안내문이 나타납니다. "SQL Result Set" 버튼을 클릭하면 생성된 데이터베이스에 프로그래밍된 테이블을 생성하고 2개의 샘플 레코드가 기록됩니다. 그리고 화면에 2개의 레코드가 추가됐다는 안내문이 출력됩니다.

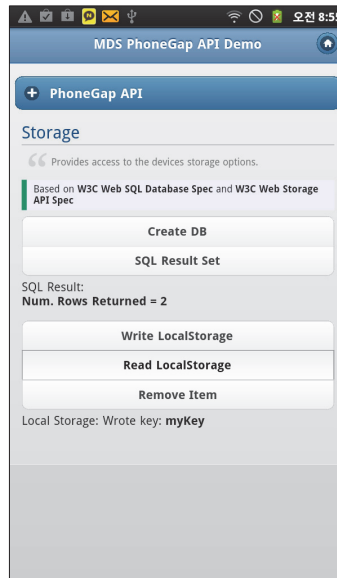
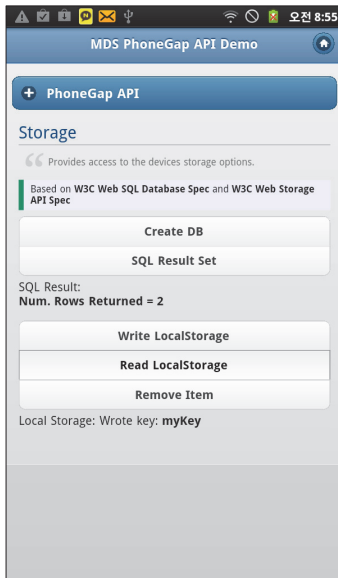




스텝 3

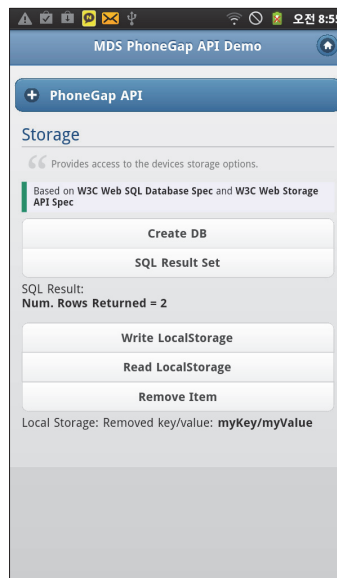
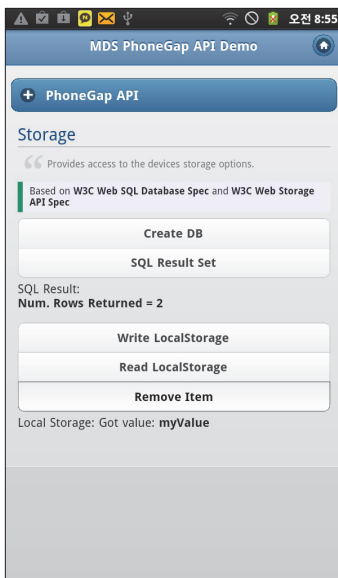
"Write LocalStorage" 버튼을 클릭하면 프로그래밍된 바에 따라 로컬 환경에 변수명이 myKey인 환경변수를 등록합니다. "Read LocalStorage" 버튼을 클릭하면 방금 등록한 myKey에 대한 값을 화면에 출력합니다.





스텝 4

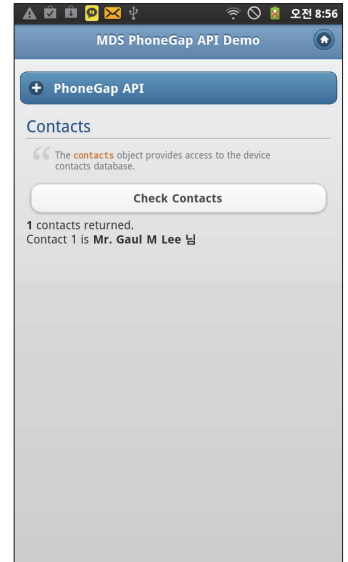
"Remove Item" 버튼을 클릭하면 앞서 등록한 환경변수를 삭제합니다.



실물 단말기 폰갭 데모 : 연락처(Contacts)

스텝 1

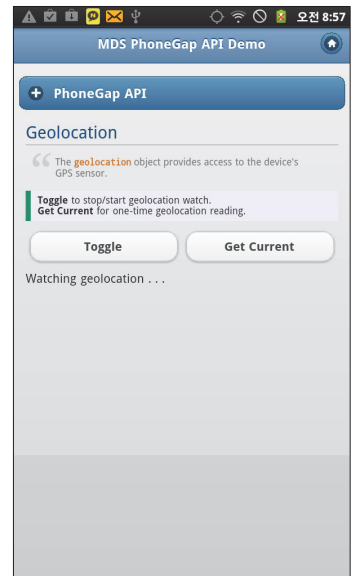
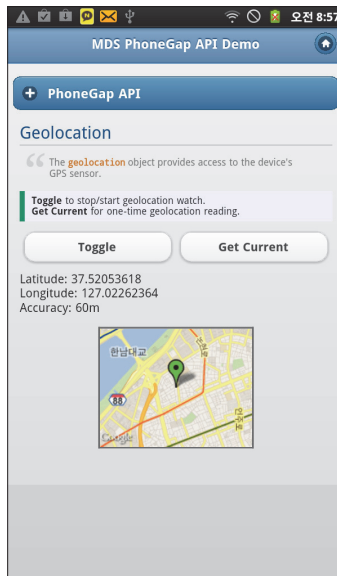
Contacts 화면에서 "Check Contacts" 버튼을 클릭하면 실험 단말기에 있는 연락처 정보가 화면에 나타납니다.

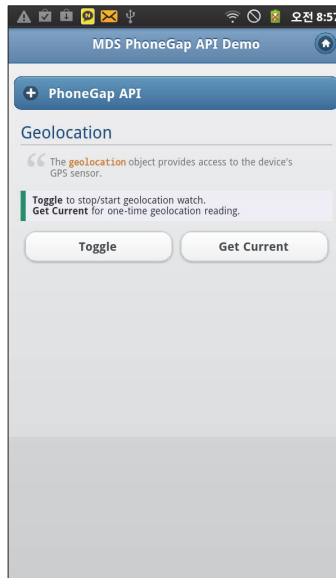
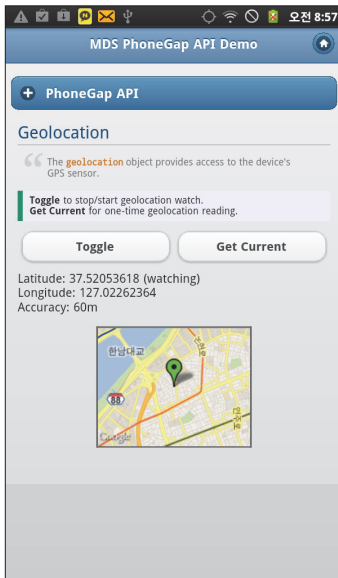


실물 단말기 폰갭 데모 : GPS 위치 감지 센서(Geolocation)

스텝 1

Geolocation 화면에서 "Get Current" 버튼을 클릭해봅니다. 그림과 같이 단말기의 위성 좌표로 구글 지도가 화면에 나타납니다. "Toggle" 버튼을 클릭하면 "Watching ..."이라는 안내문이 나타나면서 주기적으로 단말기의 위성 위치 정보를 구하고 화면에 구글 지도를 출력합니다. 다시 "Toggle" 버튼을 클릭하면 위성 좌표 감지를 종료합니다.

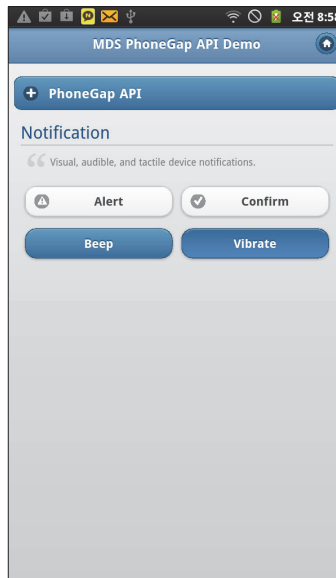
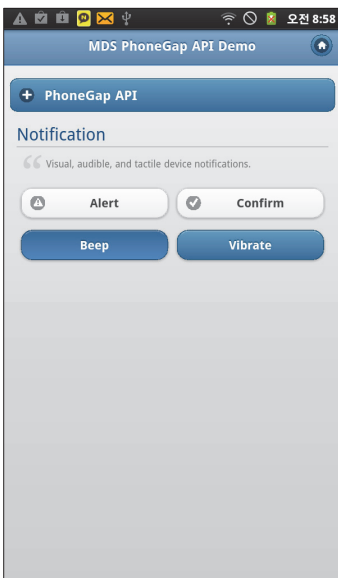




실물 단말기 폰갭 데모 : 대화상자(Notification)

스텝 1

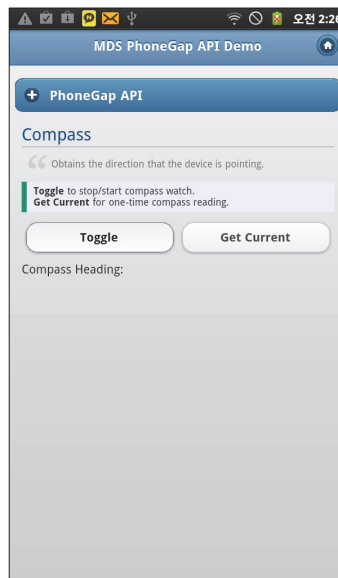
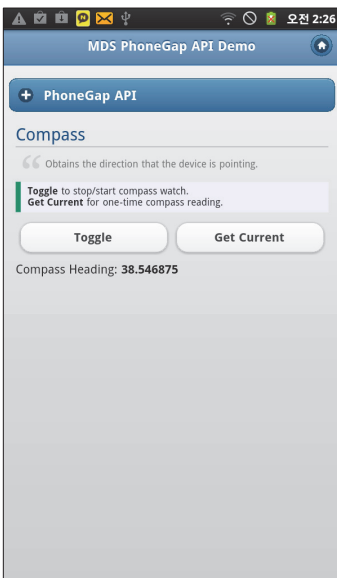
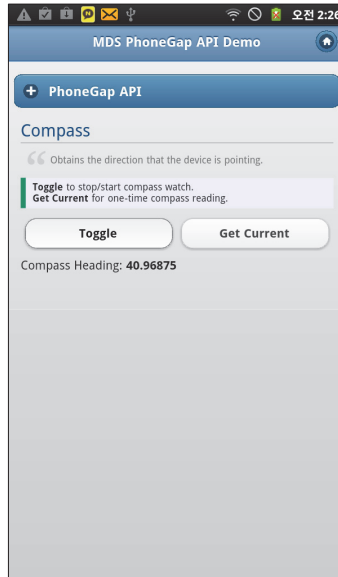
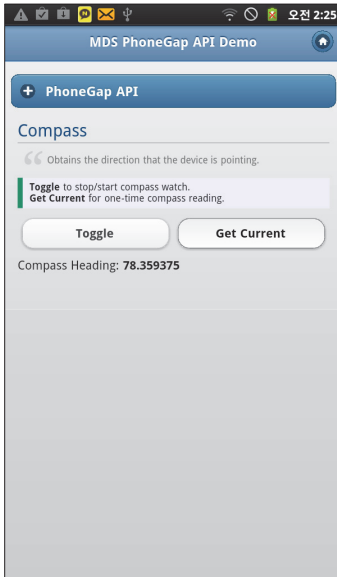
지면으로 보여 줄 수는 없지만 Notification 화면에서 Beep 버튼을 클릭하면 알람벨 소리가 들리고, Vibrate 버튼을 클릭하면 진동이 울립니다.



실물 단말기 폰갭 데모 : 방위 센서(Compass)

스텝 1

Compass 화면에서 "Get Current" 버튼을 클릭하면 단말기의 방위각을 감지하여 화면에 출력합니다. "Toggle" 버튼을 클릭하면 주기적으로 단말기의 방위를 감지하여 화면에 출력하고, 다시 "Toggle" 버튼을 클릭하면 방위 감지를 종료합니다.



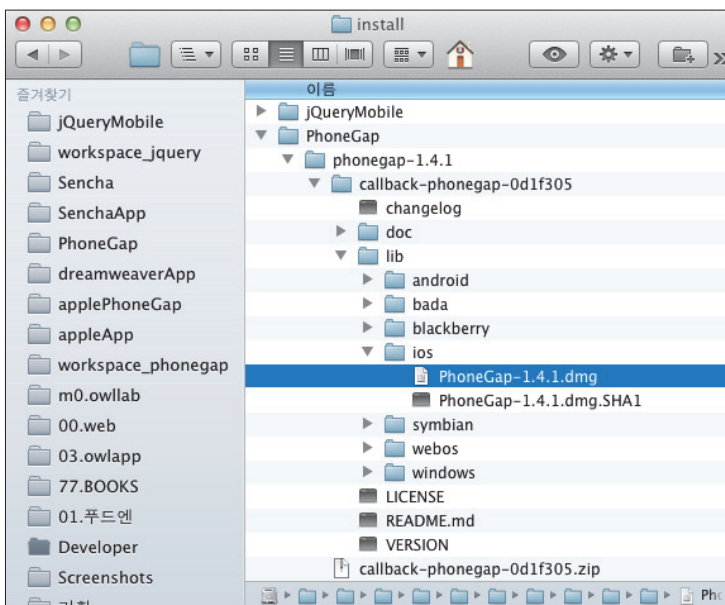
3.2 아이폰용 웹앱 프로젝트

Xcode 폰갭 플러그인 설치

아이폰용 폰갭은 다음과 같이 플러그인 설치형으로 제공됩니다.

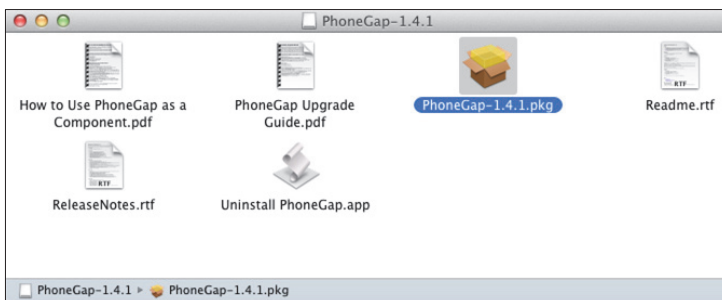
스텝 1

폰갭 사이트에서 다운받은 폰갭 패키지에서 ios 폴더에 있는 PhoneGap-x.x.x.dmg 파일이 설치본이 들어 있는 일종의 가상 디스크 이미지 파일입니다. 이 파일을 더블클릭합니다.



스텝 2

가상 디스크 이미지가 마운트되면 설치 안내 파일과 설치본을 찾을 수 있습니다. 설치 파일인 PhoneGap-x.x.x.pkg 파일을 더블클릭하면 설치 마법사가 나타납니다.



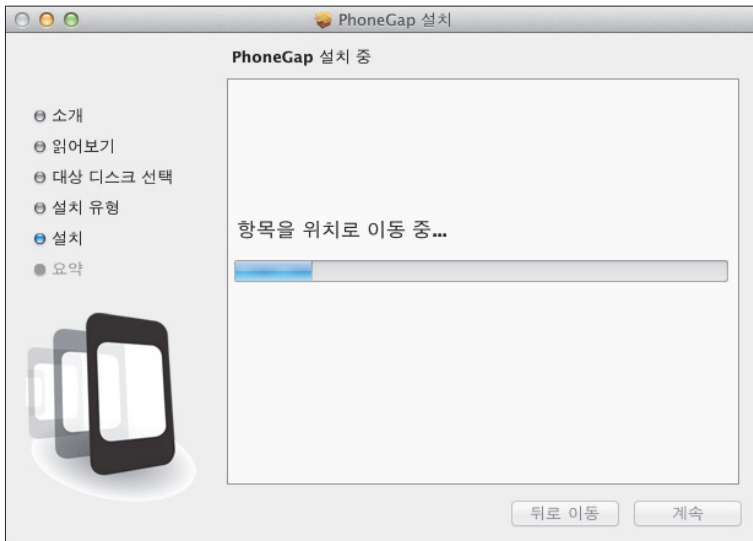
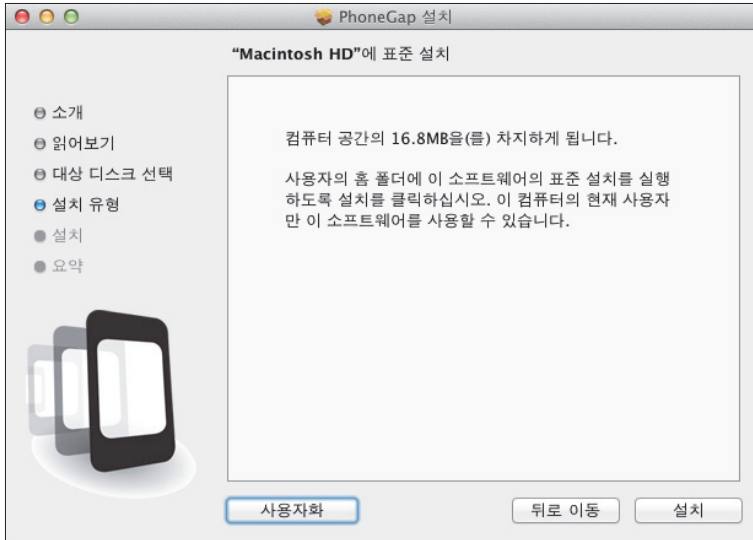
스텝 3

설치 안내문을 확인하면서 "계속" 버튼을 클릭합니다.



스텝 4

필요에 따라 "사용자화" 버튼을 이용하여 설치 위치나 설치 패키지를 선별할 수 있습니다.
"설치" 버튼을 클릭하면 파일 복사 과정이 진행됩니다.



스텝 5

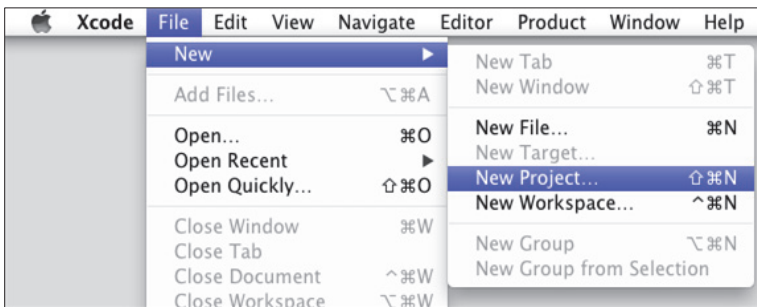
설치가 완료되면 그림과 같이 나타납니다. "닫기" 버튼을 클릭하여 설치를 완료합니다.



폰갭 프로젝트 생성

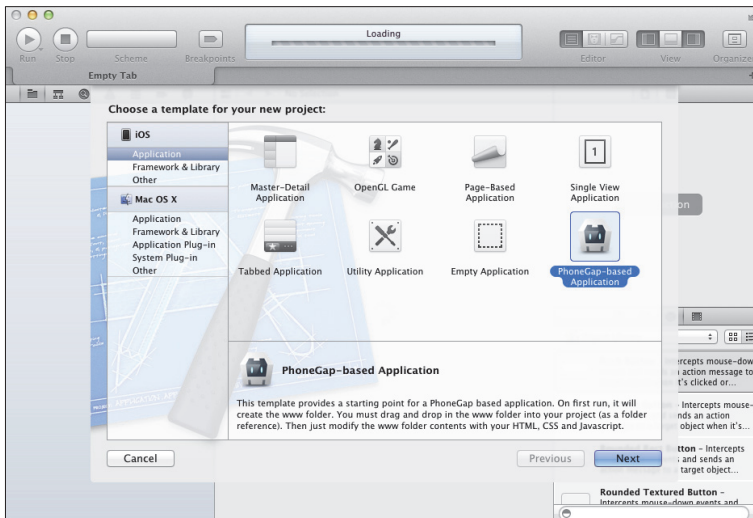
스텝 1

아이폰 앱 개발 도구인 Xcode 프로그램을 실행하고, "File > New > New Project..." 메뉴를 실행합니다.



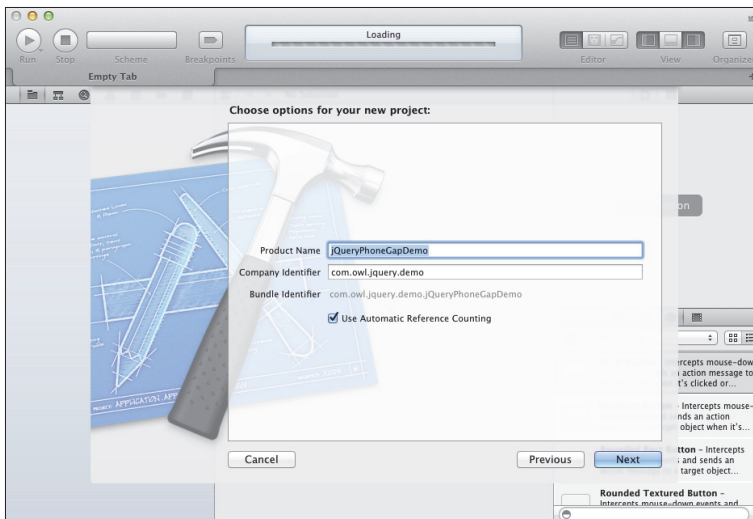
스텝 2

프로젝트 템플릿 중 PhoneGap-based Application을 선택하고 "Next" 버튼을 클릭합니다.



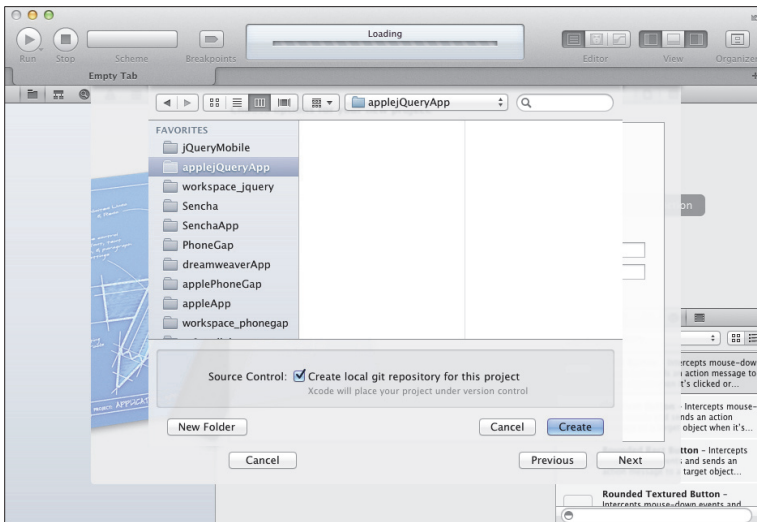
스텝 3

그림과 같이 프로젝트명과 패키지명을 입력하고 "Next" 버튼을 클릭합니다.



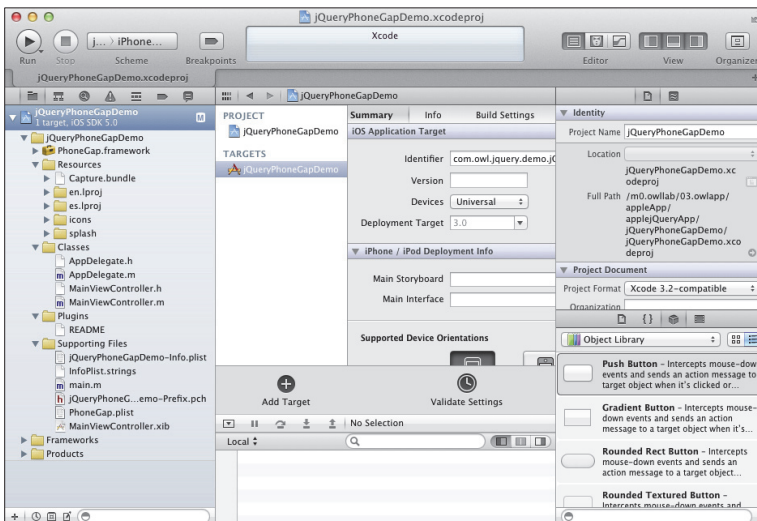
스텝 4

프로젝트 소스를 저장할 폴더를 선택하고 "Create" 버튼을 클릭합니다.



스텝 5

그림과 같이 폰갭 데모 프로젝트가 생성되었습니다.

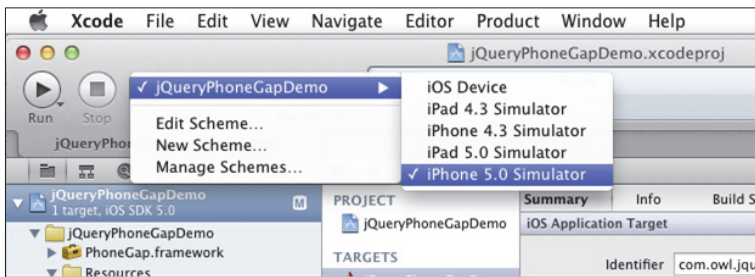


프로젝트에 www 폴더 추가

생성된 프로젝트를 다음과 같이 가상기기에 실행하면 샘플 소스가 들어 있는 www 폴더가 생성됩니다. 하지만 이 www 폴더는 생성됐을 뿐 프로젝트에 등록되지 않은 상태이므로 수동으로 www 폴더를 드래그앤드롭하여 프로젝트에 등록해주어야 합니다.

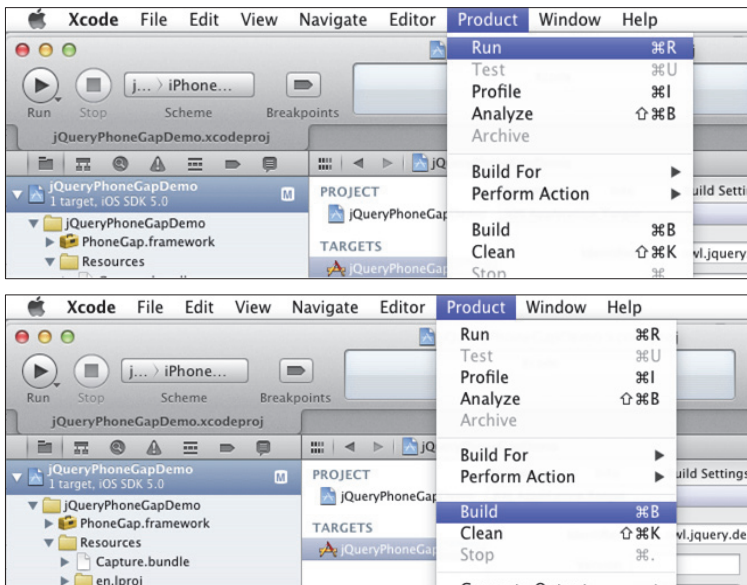
스텝 1

“Xcode > Scheme > 프로젝트 > iPhone x.x Simulator”를 선택하여 가상기기에 이 프로젝트를 실행하도록 설정합니다.



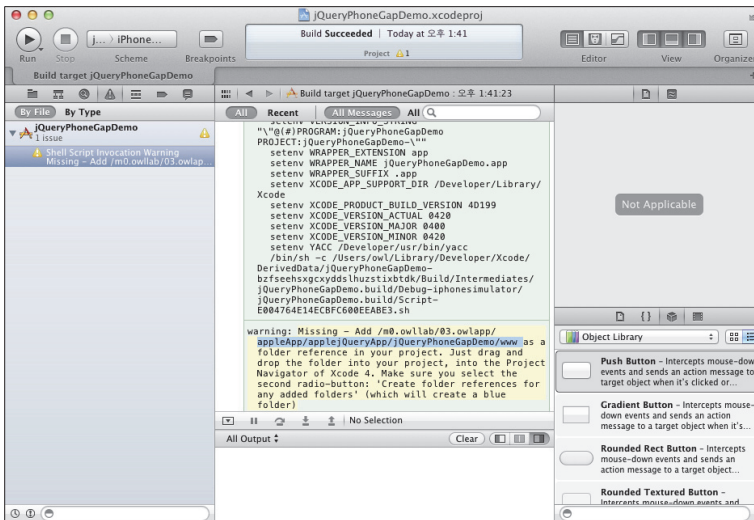
스텝 2

“Xcode > Project > Run” 메뉴를 실행하거나 “Xcode > Project > Build” 메뉴를 실행하면 Xcode 는 프로젝트를 컴파일하면서 백그라운드에서 www 폴더를 생성합니다.



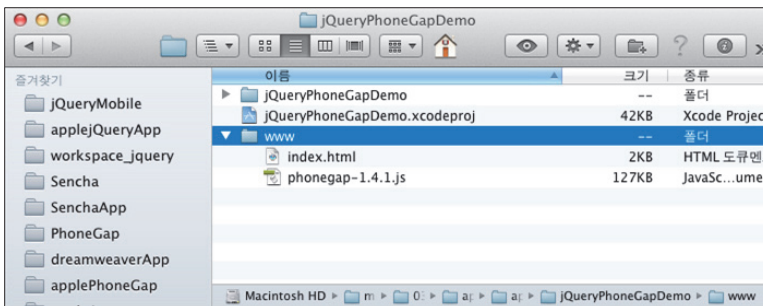
스텝 3

그림과 같이 컴파일 과정에서 www 폴더가 프로젝트에서 누락됐다는 경고문을 볼 수 있습니다.



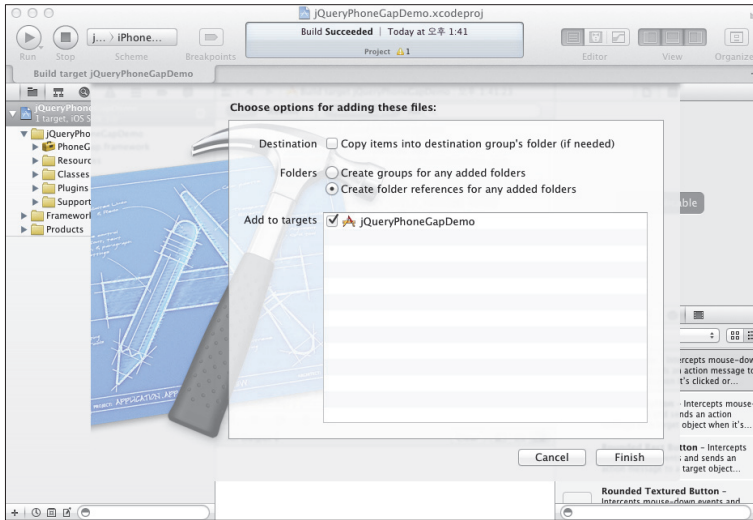
스텝 4

Finder에서 프로젝트 폴더를 확인해보면 그림과 같이 www 폴더가 생성되어 있습니다. 여기서 중요한 것은 폰갭 라이브러리 파일인 phonegap-x.x.x.js 파일을 잘 보관하는 것입니다. 폰갭은 안드로이드, 아이폰, 윈도우폰 등 각 플랫폼에 적합한 폰갭 라이브러리 파일이 서로 다릅니다.



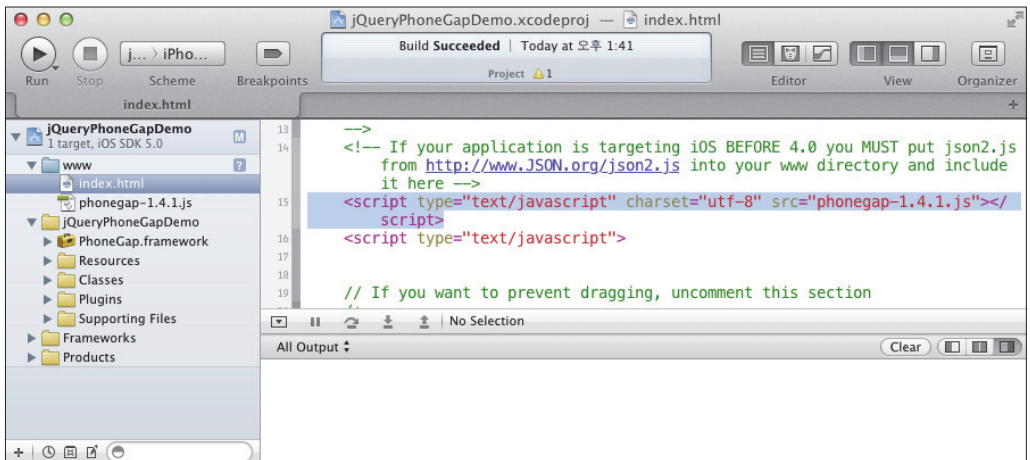
스텝 5

위의 www 폴더를 마우스로 드래그하여 Xcode의 프로젝트 최상위 위치에 드롭하면 그림과 같이 "파일 등록 옵션 설정" 화면이 나타납니다. "Create folder references ..." 옵션을 선택하면 프로젝트에 등록만 하고 파일을 복사하지 않습니다. 본 사례는 이미 프로젝트 폴더에 www 폴더가 있기 때문에 별도로 복사해올 필요가 없어 이 옵션을 사용하고 있습니다. "Finish" 버튼을 클릭하여 www 폴더를 프로젝트에 등록합니다.



스텝 6

그림과 같이 프로젝트에 `www` 폴더가 등록된 것을 확인할 수 있고, `index.html` 파일의 소스를 간단히 살펴보면 `phonegap-x.x.x.js` 파일을 참조하는 것을 볼 수 있습니다. "Run" 버튼을 클릭하여 재컴파일하고 가상기기에서 실행하면 더 이상 오류나 경고가 나타나지 않고 가상기기에서 프로젝트가 실행됩니다.



스텝 7

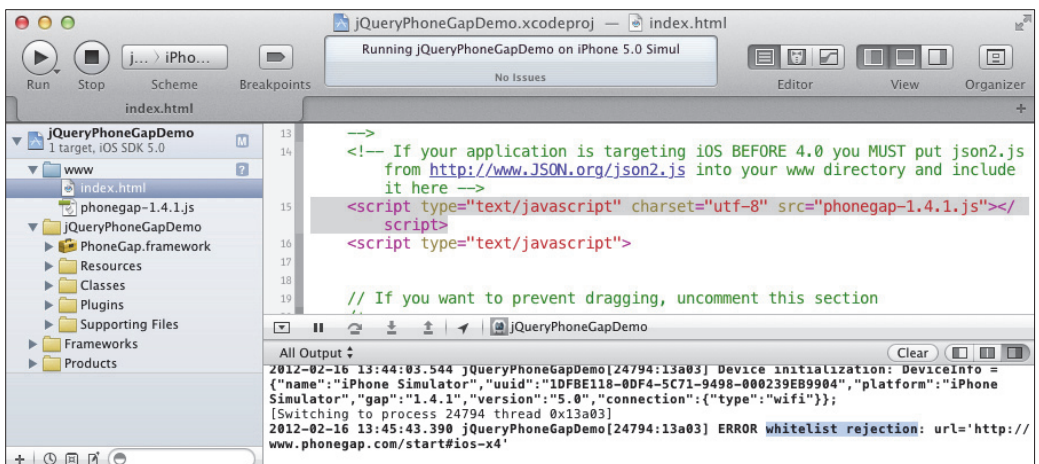
그림과 같이 가상기기에서 프로젝트가 실행되고 폰갭 라이브러리를 올바르게 호출했다는 대화상자가 나타나는 것을 볼 수 있습니다. "OK" 버튼을 클릭하여 대화상자를 닫고 "PhoneGap Start" 링크 버튼을 클릭하면 폰갭 웹사이트로 이동하지 못하는 문제를 발견할 수 있습니다.



웹 통신을 위한 화이트리스트 등록 : PhoneGap.plist

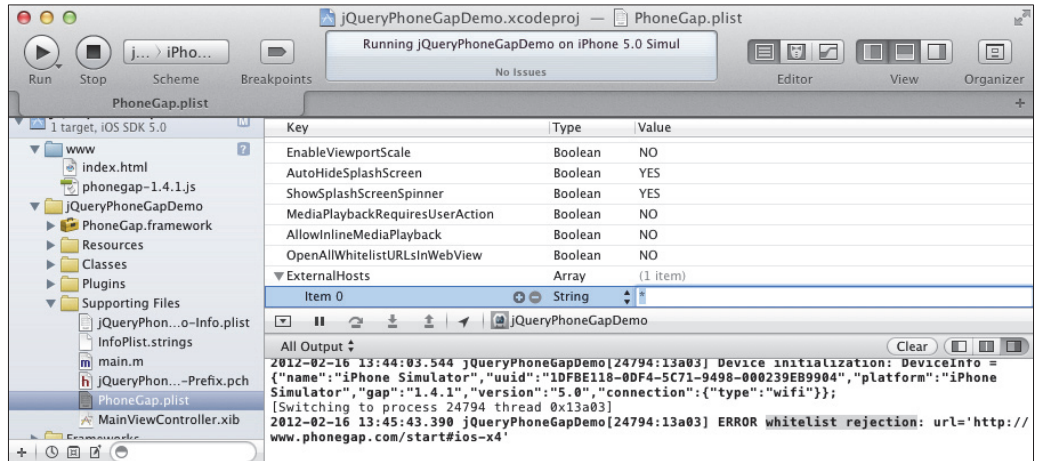
스텝 1

Xcode에서 로그를 확인해보면 그림과 같이 "whitelist rejection ..." 에 대한 오류가 발생한 것을 확인할 수 있습니다. 아이폰의 경우 이와 같이 웹 통신에 대한 도메인을 앱 속성에 등록해주어야 합니다.



스텝 2

그림과 같이 PhoneGap.plist 파일에서 "+" 버튼을 이용하여 ExternalHosts 속성에 item을 하나 추가하고, 웹 통신을 허용할 도메인을 입력합니다. 본 사례의 경우 모든 웹 통신을 허용하도록 "*"를 입력하고 있습니다. PhoneGap.plist 파일을 저장하고 Run 버튼을 클릭하면 Xcode는 이 프로젝트를 재컴파일한 후 가상기기에 재설치하여 실행합니다.



스텝 3

이제 "PhoneGap Start" 링크 버튼을 클릭하면 폰갭의 웹사이트로 연결할 수 있습니다.



웹앱 소스 등록 : jQuery Mobile 폰갭 데모 소스

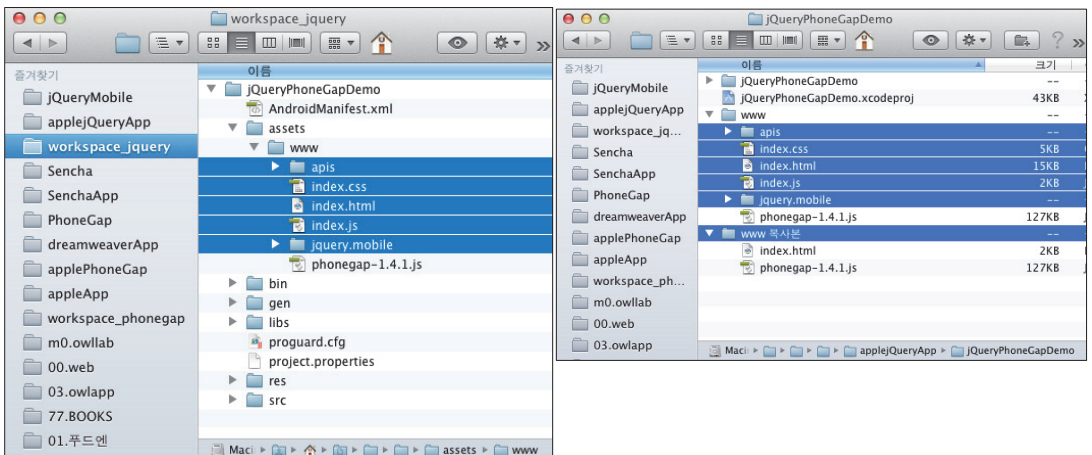
아이폰용은 안드로이드와 같이 샘플 소스를 선택적으로 추가하는 옵션을 지원하지 않습니다. 따라서 다음과 같이 안드로이드에서 만들었던 웹앱 소스를 Xcode 프로젝트의 www 폴더에 수동으로 추가해야 합니다.

스텝 1

안드로이드에서 실험으로 검증했던 웹앱 소스를 그림과 같이 준비합니다. 이 때 주의할 사항은 phonegap-x.x.x.js 파일은 복사해가서는 안 된다는 점입니다. jQuery Mobile 관련 라이브러리와 소스들은 HTML5 표준에 따른 솔루션이기 때문에 HTML5를 지원하는 플랫폼에는 모두 호환됩니다. 따라서 jQuery Mobile은 플랫폼에 따른 특성이 없어 완벽한 "하이브리드" 개념을 실현하고 있습니다.

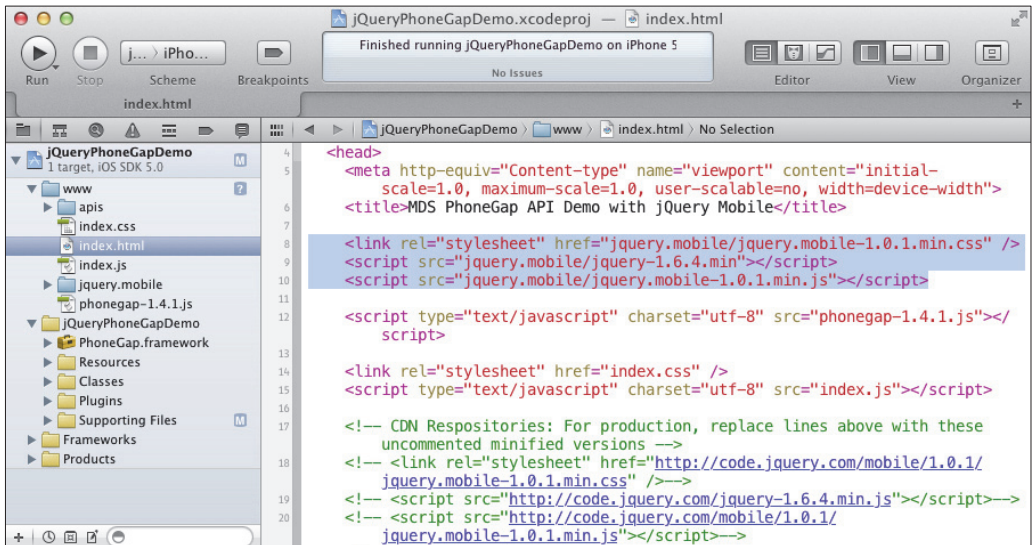
하지만 폰갭의 경우는 다릅니다. 폰갭은 각 플랫폼 고유의 기능과 연동해야 하기 때문에 플랫폼에 따른 소스가 각기 다를 수밖에 없습니다. 그래서 폰갭 라이브러리는 각 플랫폼마다 서로 다른 라이브러리를 사용합니다. 물론 폰갭도 자바스크립트를 사용하기 때문에 각 플랫폼의 특성을 고려한 통합 라이브러리를 만들 수 있으나 성능의 관점에서는 좋은 방법이라 할 수 없습니다. 나중에 폰갭이 각 플랫폼의 네이티브 라이브러리를 통합한다면 하나의 폰갭 자바스크립트 라이브러리 파일을 사용할 수도 있겠지만, 집필 당시까지는 이 문제에 유의해야 할 것입니다.

그림과 같이 phonegap-x.x.x.js 파일을 제외하고 www 폴더에 있는 모든 파일을 선택하고 복사하여 Xcode의 www 폴더 안에 붙여 넣습니다. 참고로 필자는 이전의 www 폴더를 복사본으로 백업해 두었습니다. 개발을 하다 보면 실수로 phonegap-x.x.x.js 파일까지 복제해오는 착오에 대비하기 위해서 입니다.



스텝 2

Xcode는 www 폴더의 변경된 파일을 자동으로 인식합니다. index.html 파일을 열어 참조하고 있는 jQuery Mobile 라이브러리 설정과 폰갭 라이브러리 설정을 확인해봅니다. Run 버튼을 클릭하면 프로젝트를 재컴파일하여 가상기기에 재설치하고 실행합니다.



가상기기에서 실험하기

스텝 1

jQuery Mobile로 만든 폰갭 데모가 아이폰 가상기에서도 같은 디자인으로 나타납니다.



스텝 2

폰갭의 각각의 기능들은 이미 안드로이드에서 보여주었으므로 생략하겠습니다. 참고로 단말기에서 홈 버튼(□)을 눌러 데모 앱을 빠져나오면 그림과 같이 jQueryPhoneGapDemo 앱 아이콘을 확인할 수 있습니다. 이 아이콘을 변경하는 것은 Xcode에서 작업해야 합니다.



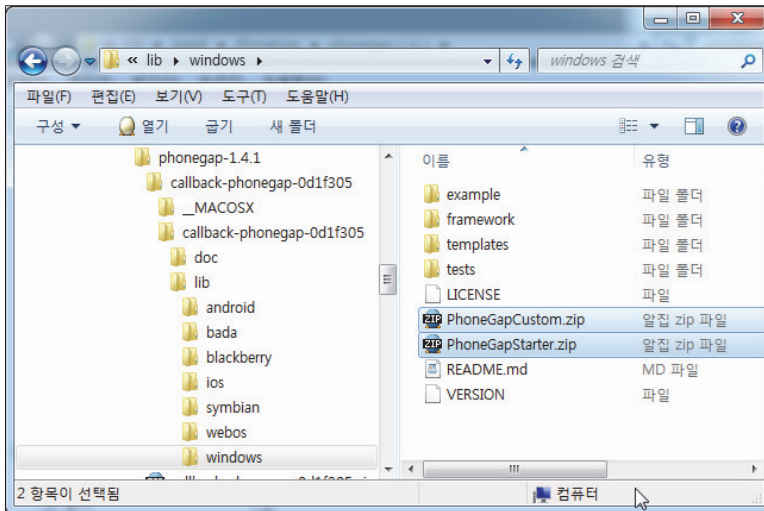
3.3 윈도우폰용 웹앱 프로젝트

윈도우폰의 개발 도구는 Visual Studio입니다. 윈도우폰용 폰갭은 프로젝트 템플릿을 등록해서 폰갭 프로젝트를 생성하는 방식을 사용합니다.

Visual Studio 폰갭 템플릿 설치

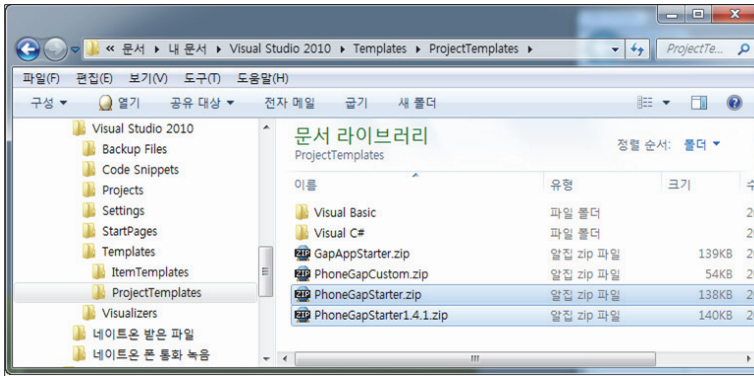
스텝 1

폰갭 사이트에서 다운받은 폰갭 패키지의 windows 폴더에 zip으로 압축된 윈도우폰용 폰갭 프로젝트 템플릿이 있습니다. 폰갭 1.4.1은 두 개의 폰갭 프로젝트 템플릿을 제공합니다. 이 두 파일을 복사해서 Visual Studio의 프로젝트 템플릿 폴더에 붙여 넣는 것으로 설치를 완료합니다.



스텝 2

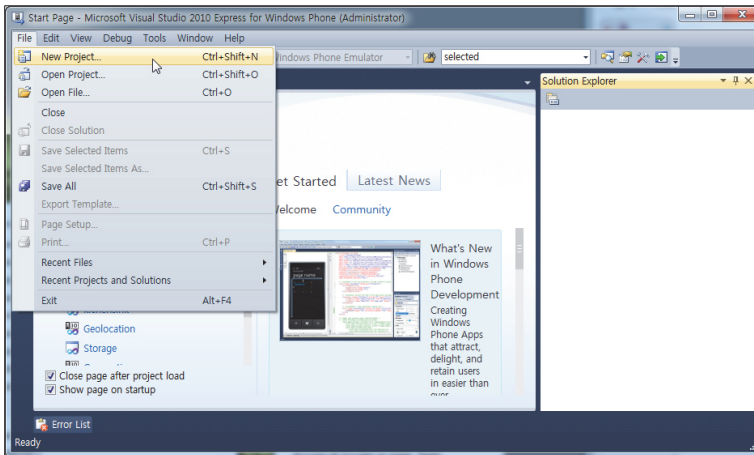
비주얼 스튜디오의 프로젝트 템플릿 폴더는 그림과 같이 "내 문서\Visual Studio 2010\Templates\ProjectTemplates"입니다. 여기에 폰갭 프로젝트 템플릿 파일을 넣어두면 비주얼 스튜디오가 부팅할 때 추가한 폰갭 프로젝트 템플릿을 읽어 옵니다.



폰갭 프로젝트 생성

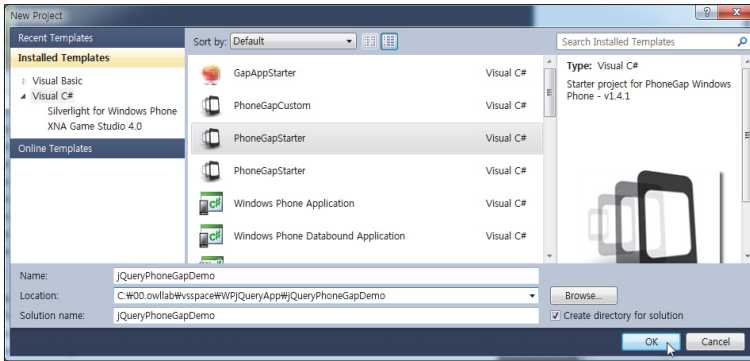
스텝 1

비주얼 스튜디오를 실행하고 “File > New Project...” 메뉴를 실행합니다.



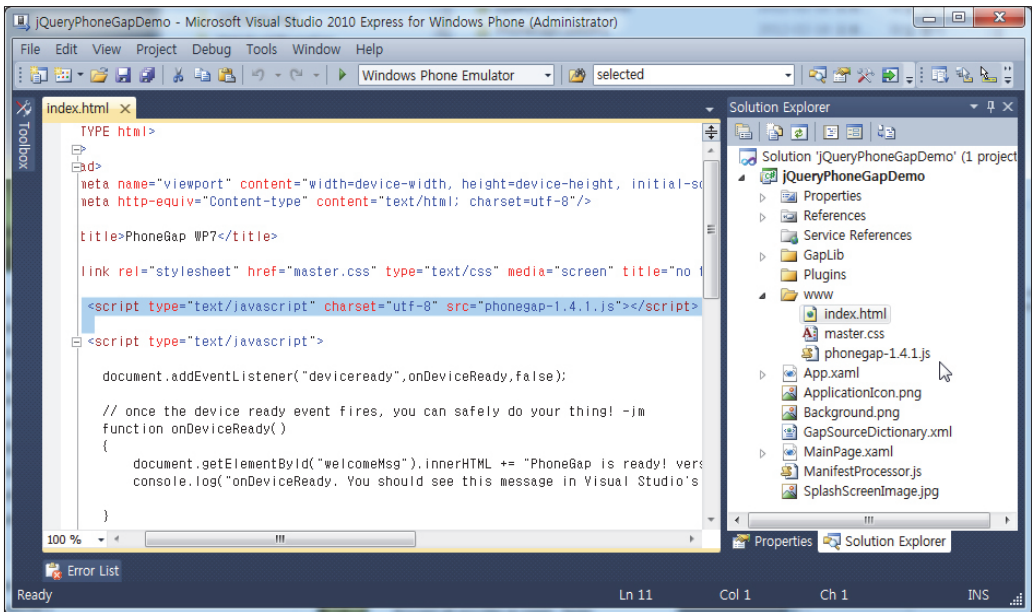
스텝 2

“New Project” 창에서 앞서 추가한 PhoneGapStarter 프로젝트 템플릿을 선택하고 프로젝트명과 솔루션명을 입력한 후 “OK” 버튼을 클릭합니다.



스텝 3

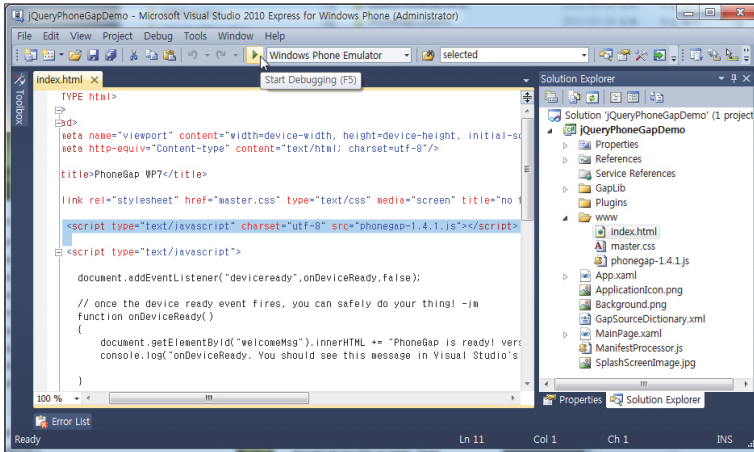
그림과 같이 간단히 윈도우폰용 폰갭 프로젝트가 생성됐습니다. index.html 파일을 열어보면 폰갭 라이브러리를 호출하는 구문을 볼 수 있습니다. 이 샘플 소스에서 눈여겨 볼 또 하나의 구문은 <meta> 태그로 작성한 viewport 설정입니다. 이 뷰포트 설정은 뷰화면의 가로/세로를 단말기의 크기와 동일하게 설정하고 있습니다. 이 설정은 나중에 참조할 필요가 있을 것이므로 기억해두기 바랍니다.



가상기기에서 실험하기

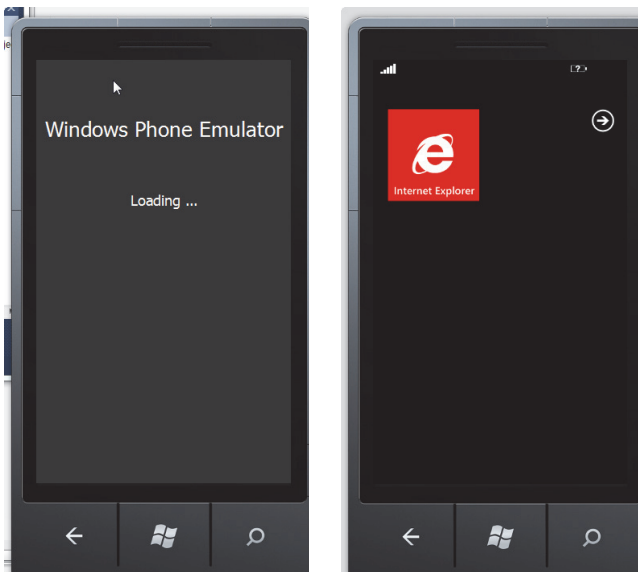
스텝 1

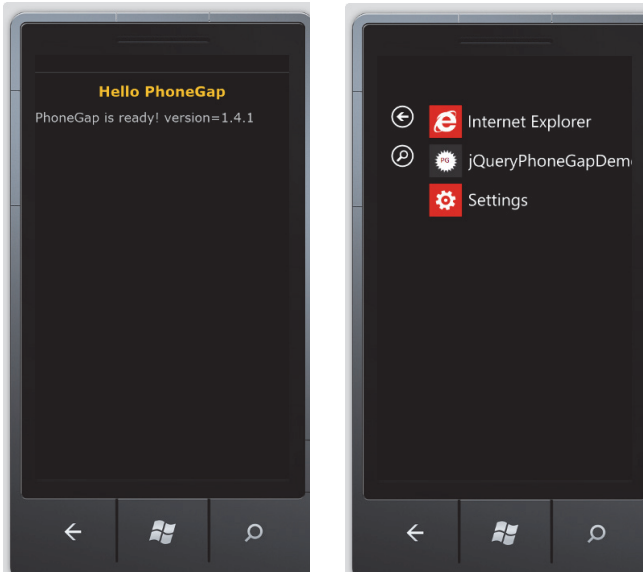
그림과 같이 "Start Debugging" 아이콘 버튼을 클릭하면 이 프로젝트를 컴파일하고 가상기기가 실행됩니다.



스텝 2

가상기기에 나타나고, 샘플 앱을 설치하며, index.html로 작성된 화면이 나타납니다. 단말기의 홈 버튼을 클릭해보면 jQueryPhoneGapDemo 앱이 앱 목록에 나타납니다.



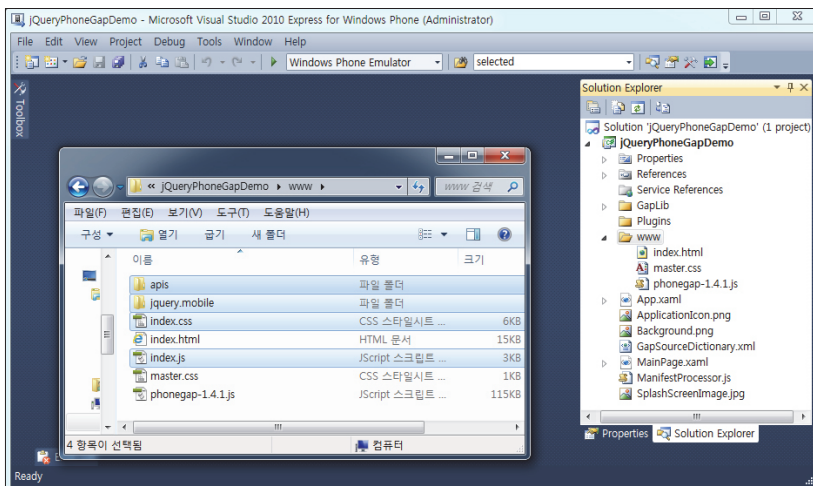


웹앱 소스 등록 : jQuery Mobile 폰갭 데모 소스

윈도우폰도 아이폰에서와 같이 안드로이드에서 만든 웹앱 소스를 가져옵니다. phonegap-x.x.x.js는 가져오지 않고 비주얼 스튜디오용 폰갭 프로젝트 템플릿이 제공하는 폰갭 라이브러리를 사용합니다.

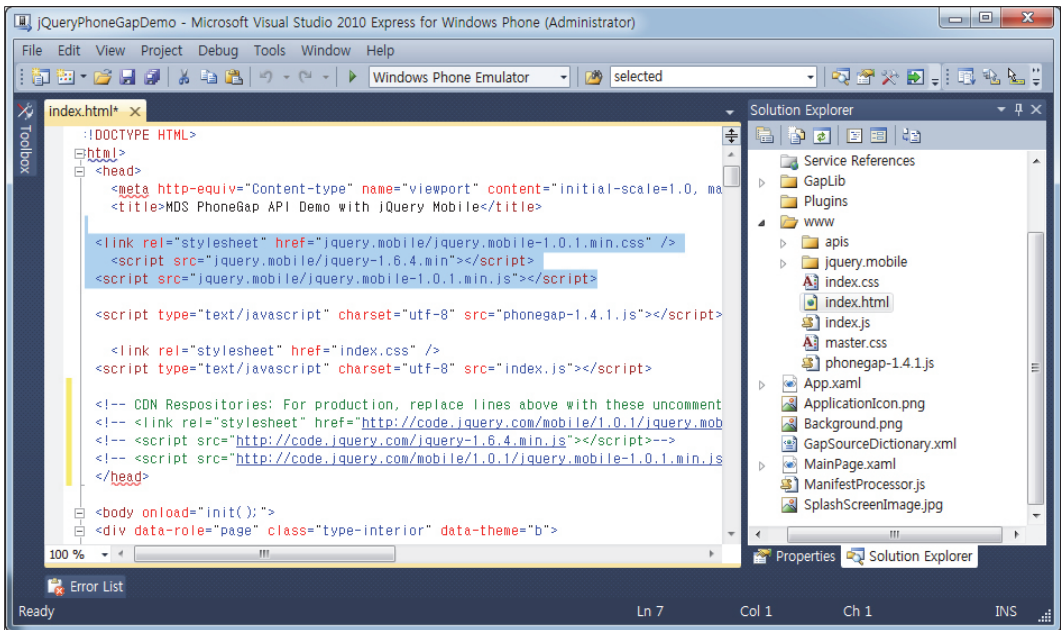
스텝 1

그림처럼 파일 탐색기로 웹 소스를 www 폴더에 복사해옵니다. 비주얼 스튜디오는 아이폰의 Xcode와 달리 www 폴더에 새로 추가된 파일들을 비주얼 스튜디오 폴더에 등록해야 합니다. 파일 탐색기에서 새로 추가된 파일과 폴더를 선택하고 드래그앤드롭으로 비주얼 스튜디오의 www 폴더에 등록합니다.



스텝 2

index.html 파일을 열면 그림과 같이 새로 가져온 소스로 변경되어 있는 것을 확인할 수 있습니다.

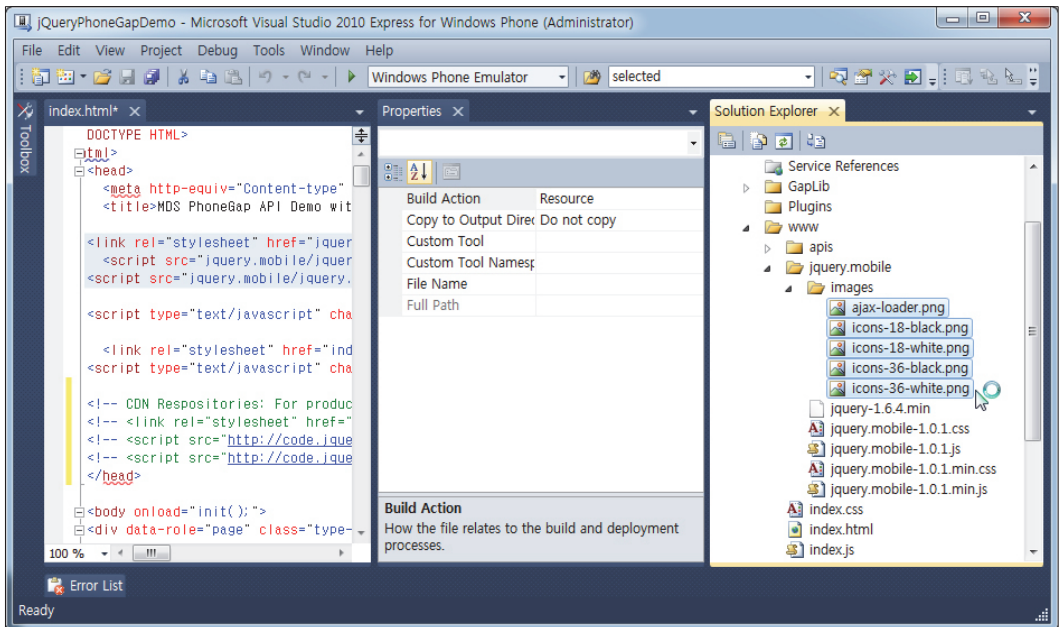


신규 소스의 Content 설정

비주얼 스튜디오에서는 추가한 소스 중 .png나 .min과 같은 확장자의 파일을 Content 형으로 자동 설정하지 않아 화면에 이미지가 나타나지 않거나 jQuery Mobile 라이브러리를 참조할 수 없는 현상이 있습니다. 다음과 같이 추가한 웹앱 소스 중 Build Action이 Content로 설정되지 않은 소스를 찾아 교정하는 작업이 필요합니다.

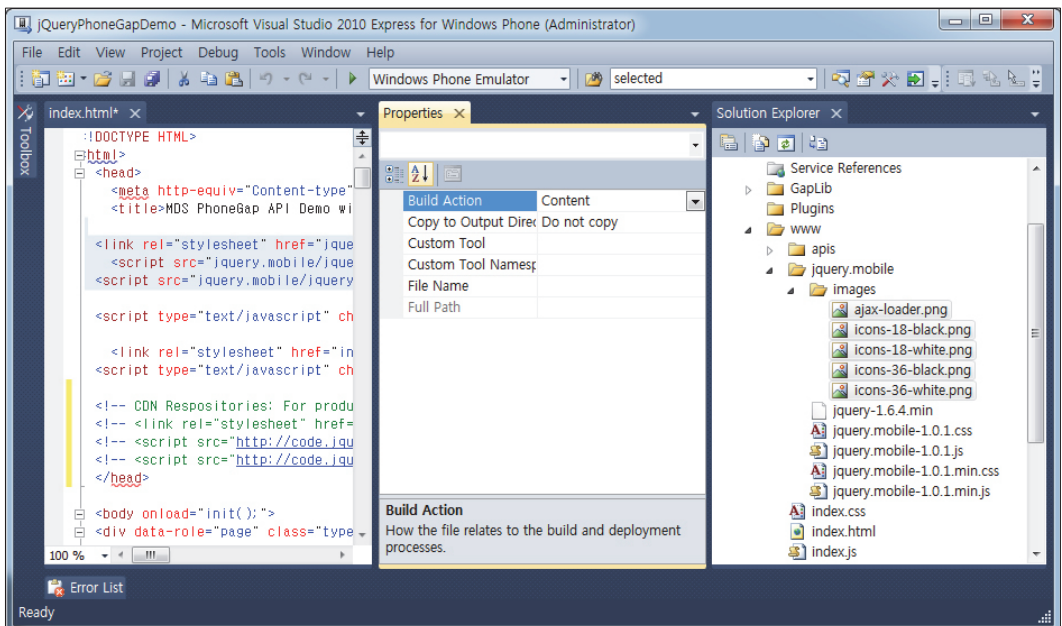
스텝 1

jQuery Mobile에서 사용하는 이미지들을 찾아 그림과 같이 Properties 창에서 Build Action 속성을 확인해보면 Resource로 되어 있습니다.



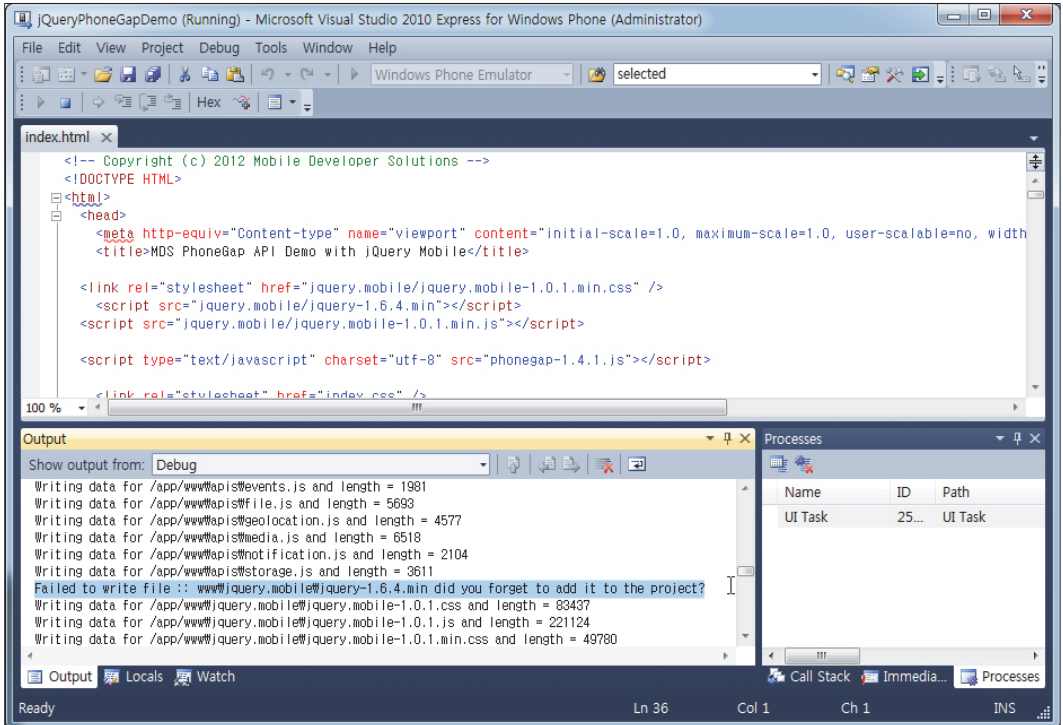
스텝 2

이 이미지들의 Build Action을 "Content"로 변경합니다.



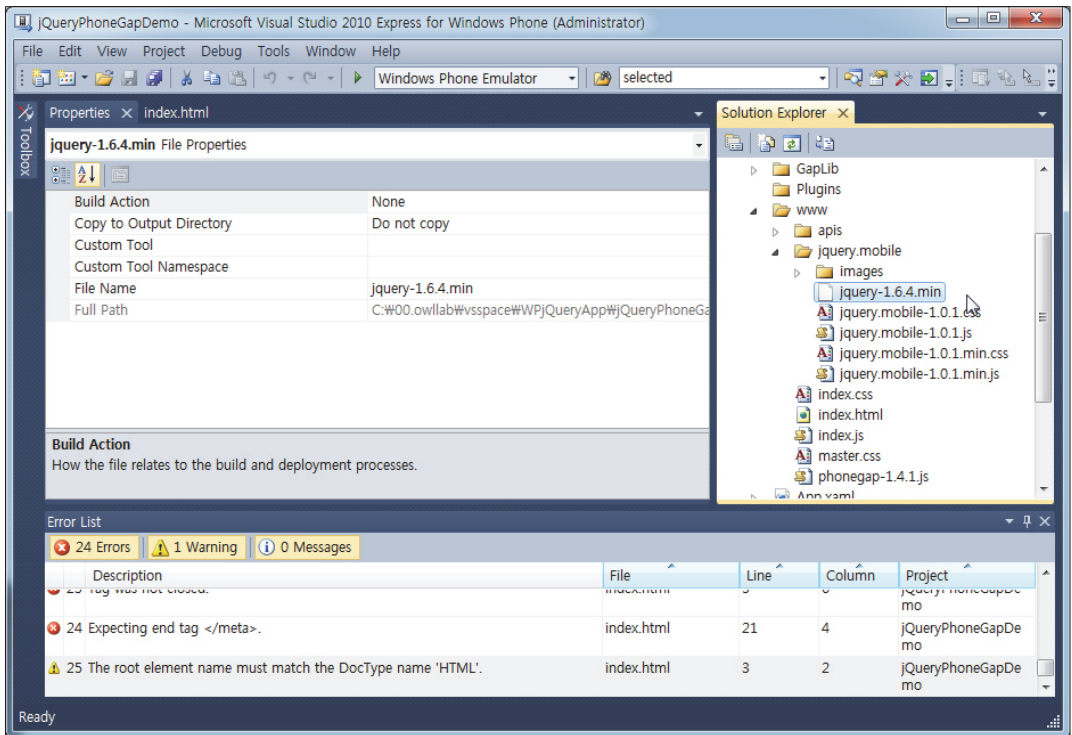
스텝 3

이 상태에서 실행을 시도하면 그림과 같이 Output 창에서 jquery-x.x.x.min 파일이 누락했다는 안내문을 볼 수 있습니다.



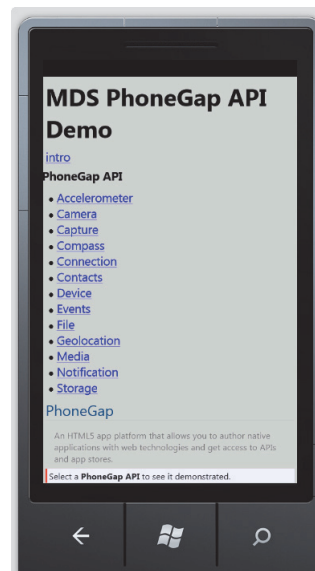
스텝 4

그림과 같이 jquery-x.x.x.min 파일을 찾아 Build Action을 확인해보면 None으로 설정되어 패키징하는 과정에 이 파일을 누락한 것입니다.



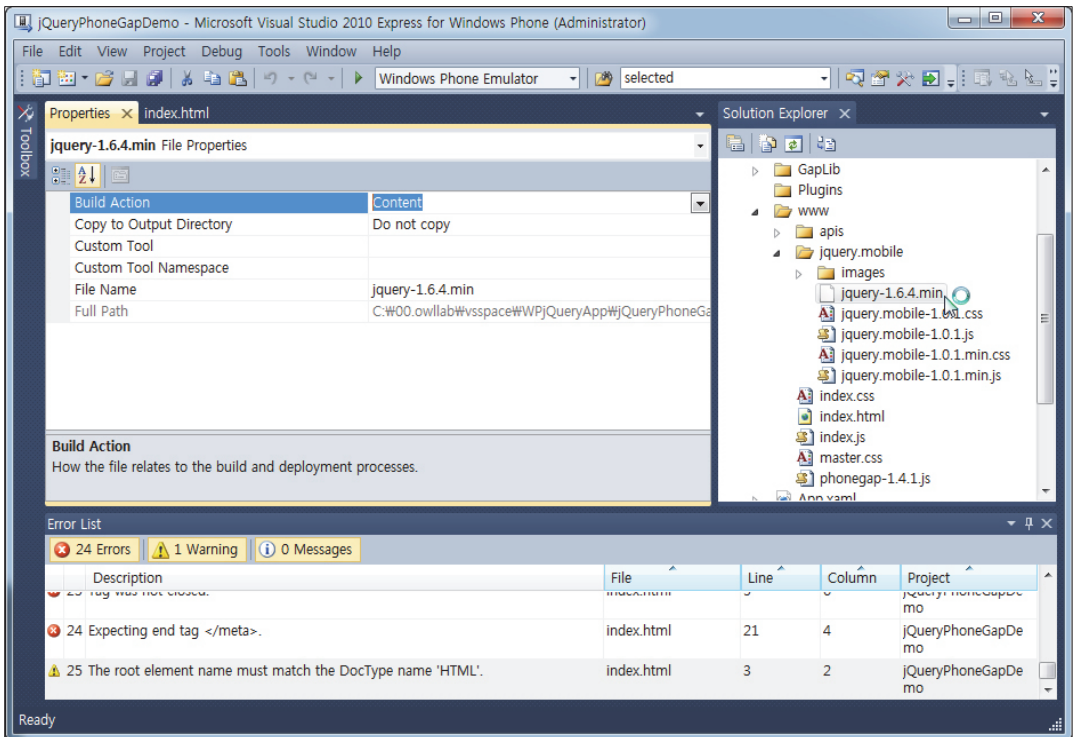
스텝 5

이 상태로 가상기기에 실행하면 그림과 같이 jQuery Mobile이 적용되지 않은 상태의 화면이 나타날 것입니다.



스텝 6

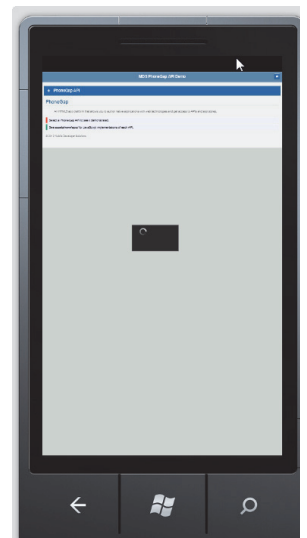
jquery-x.x.x.min 파일의 Build Action 속성을 Content로 변경합니다.



Viewport 교정

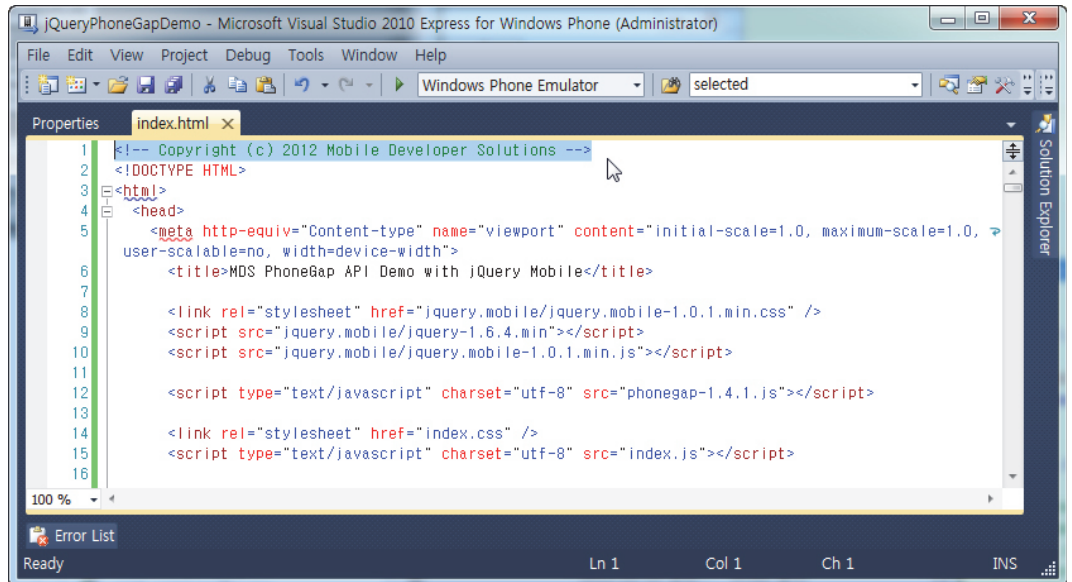
스텝 1

이 데모 앱을 실행해보면 안드로이드와 아이폰과는 달리 그림과 같이 축소되어 나타납니다. 이는 윈도우폰에서의 viewport 설정이 안드로이드나 아이폰과 다르다는 것을 의미합니다. 또한 로딩 화면이 사라지지 않는 문제가 발생할 수 있습니다. 이 두 문제는 모두 윈도우폰에서만 겪을 수 있는 문제입니다.



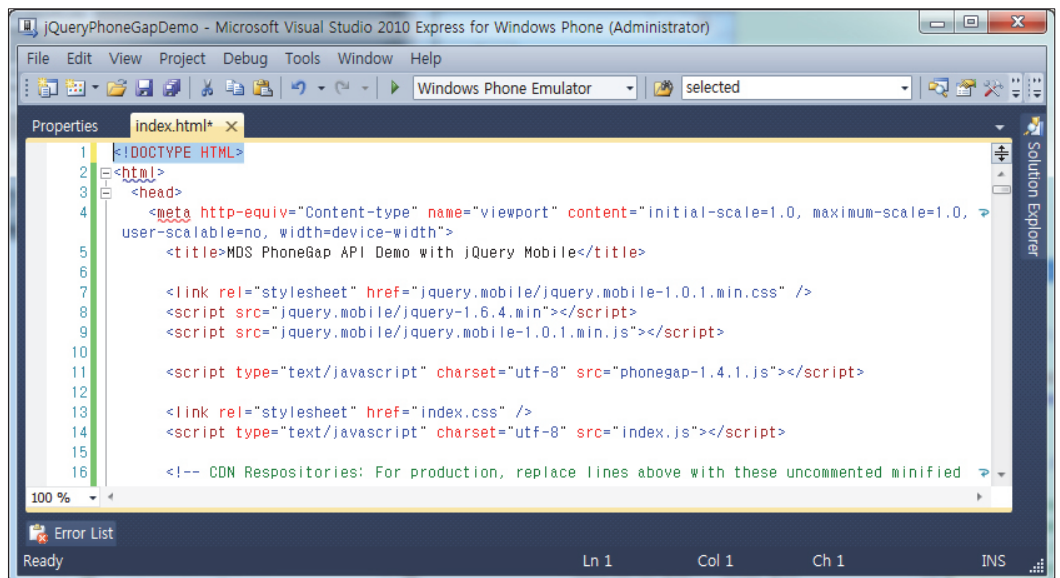
스텝 2

윈도우폰의 경우 최상단에 있는 주석 때문에 HTML5 형식을 제대로 인식하지 못하는 문제가 있습니다.



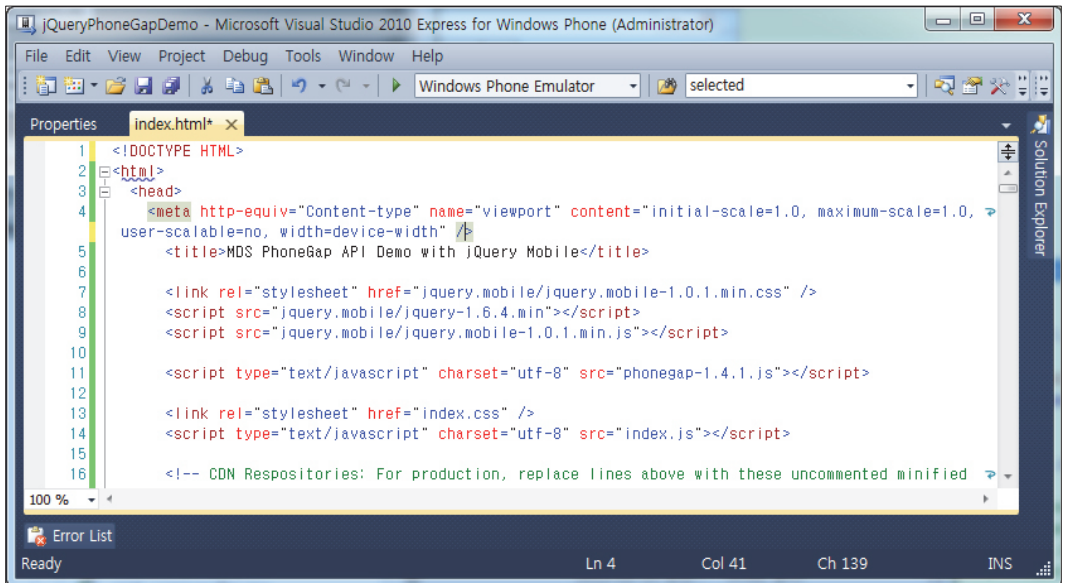
스텝 3

그림과 같이 최상단의 주석을 삭제했습니다.



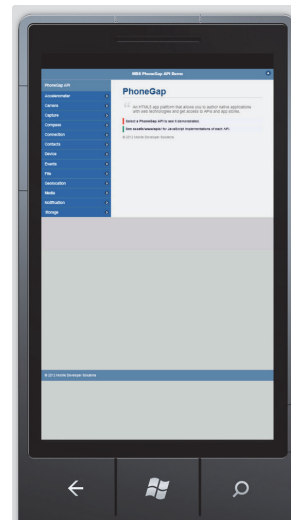
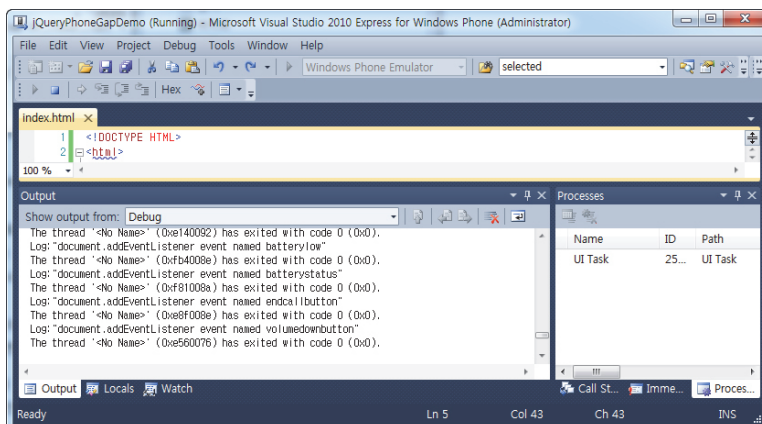
스텝 4

참고로 정확한 HTML5 작성법은 소스라인 4와 같이 태그의 끝을 `</>` 형식으로 닫아줘야 합니다. 이 부분은 큰 문제는 일으키지 않지만 비주얼 스튜디오에서 Syntax Error로 인식합니다. 그러나 실행하는 데는 문제가 없으므로 여기서는 그냥 넘어가겠습니다.



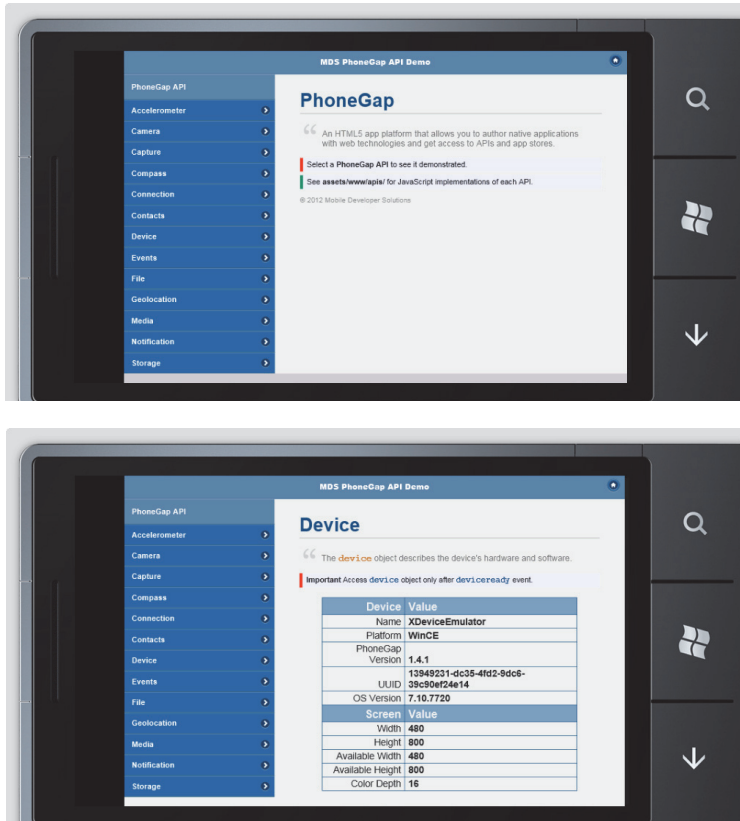
스텝 5

이 상태에서 가상기기에서 실행해보면 로딩 문제는 해결되고 화면의 크기도 어느 정도 해결돼 보이지만 개발자가 원하는 의도는 아닙니다.



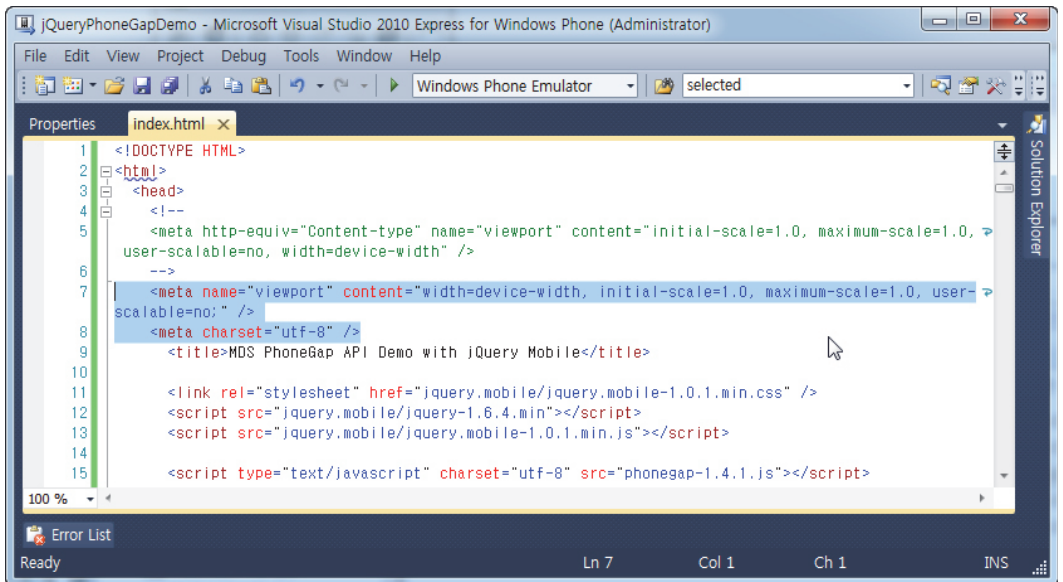
스텝 6

간단한 폰갭 기능을 시험해보면 그림과 같이 잘 작동합니다. 이와 같은 화면은 사실상 태블릿형 단말기에 적합합니다.



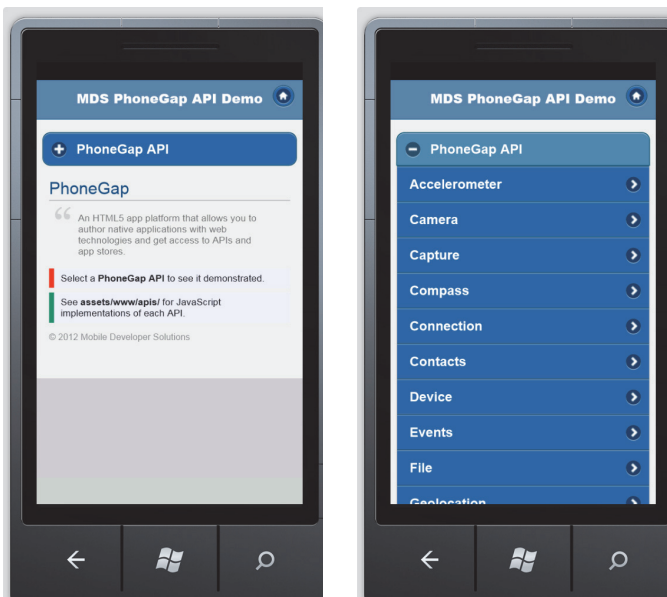
스텝 7

그림과 같이 index.html 파일을 열고 앞서 폰갭 템플릿에서 제공하는 viewport 구문으로 교체합니다.



스텝 8

위의 소스를 저장하고 재실행하면 이제는 개발자의 의도대로 그림과 같이 가상기기에 나타납니다.



스텝 9

그림과 같이 Device, Media, File, Storage 등을 가상기계에서 실험해볼 수 있습니다. 구체적인 실험은 앞서 소개한 안드로이드 사례를 참조하여 직접 시도해보기 바라며 지면상의 제약으로 본서에서는 생략하겠습니다.

