به نام او که یادش آرامبخش جانهاست

کتاب : آموزش برنامه نویسی اندروید.برای برنامه نویسان تحت وب با استفاده از MoSync Framework

Programming for Android OS.

Mobile Web Application

Framework : MoSync

ترجمه و تألیف : میلاد فشی

زمستان ۱۳۹۰ (۹ بهمن)

نحوه مطالعه این کتاب :

برنامه نویسی برای گوشی های اندروید دو شیوه است :

شیوه اول برنامه نویسی Native اندروید است. با استفاده از زبان جاوا برای اندروید برنامه مینویسید.در فصول اول و دوم و سوم مبانی ابتدایی و اولیه برنامه نویسی به این روش را توضیح داده ام.اگر دوست دارید فقط برای اندروید برنامه بنویسید و به تمام ویژگی های سخت افزاری گوشی اندروید دسترسی داشته باشید از این روش استفاده کنید و فقط این فصول را برای آشنایی اولیه بخوانید.و بعد سراغ کتاب های متوسط و پیشرفته بروید.

شیوه دوم برنامه نویسی برای اندروید اصطلاحاً Mobile Web Application است که از فصل چهارم به بعد به این شیوه پرداختم.تسلط من روی این شیوه است.و مزیت این شیوه برنامه نویسی Cross Platform است.شما با یک بار کدنویسی میتوانید برنامه خود را بر روی اندروید و iOS و windows phone اجرا کنید.اما عیب این روش این است که برنامه شما به تمامی ویژگی های سخت افزاری گوشی دسترسی ندارد و این سبک برای نوشتن برنامه های سیستمی که خیلی با سخت افزار در ارتباط است اصلا مناسب نیست.ولی برای برنامه های کاربردی مناسب است.

فهرست مطالب

مقدمه : پیش درآمدی بر اندروید

اندروید چیست ؟

اندروید سیستم عامل متن بازی[1] است برای گوشی های هوشمند(Smartphone) و کامپیوتر های دستی[2] و تبلت ها[3] که توسط " Open Handset Alliance[4] " و "Google" ساخته شد. این سیستم عامل، دارای ۱۲ میلیون کد است که ۳ میلیون آن XML، ۲.۸ میلیون خط C و ۲.۱ میلیون آن جاوا می باشد. در سال های آینده اندروید در میلیون ها سیستم همراه و موبایل مورد استفاده قرار خواهد گرفت. شاید برخی به اشتباه فکر کنند که اندروید یک پلتفرم سخت افزاری است.ولی اندروید تنها یک سیستم عامل است که برای موبایل ها ساخته شده است. هسته[5] سیستم عامل اندروید لینوکس است و دارای یک رابط کاربری ، غنی[6] و کاربر پسند[7] است و دارای توابعی برای مدیریت تماس های تلفنی است و همچنین برنامه های کاربردی و سودمندی برای کاربران نهایی[8] دارد و کتابخانه هایی از کد برای راحتی کار برنامه نویسان و توسعه دهندگان نرم افزار دارد و از چند رسانه ای هم به خوبی پشتیبانی میکند. درصورتی که علاقه مند به برنامه نویسی

---

[1] Open source
[2] Handheld
[3] Tablet
[4] برای اطلاعات بیشتر به سایت www.OpenHandsetAlliance.com مراجعه شود.

[5] Kernel(Nucleus)
[6] Rich
[7] User Friendly
[8] End User

بوده و یا در این زمینه یک حرفه ای هستید ، زمان یادگیری گسترش نرم افزار های اندروید فرا رسیده است. امروزه پلت فرم های بسیار زیادی برای موبایل وجود دارد که رقیب اندرویدند. سیمبین، آی فون، ویندوز موبایل، بلک بری، جاوا موبایل، لینوکس موبایل، و غیره از این دسته هستند. دربین پلت فرم های موجود اندروید، دارای ویژگی های خاصی است. و این ویژگی ها شاید کسانی را که میگویند «با وجود پلتفرم های مختلف برای موبایل چه نیازی به پلتفرم جدید است؟» را کمی متقاعد کند.البته بعضی از این ویژگی ها در پلتفرم های قبلی پشتیبانی میشد.

ویژگی های خاص اندروید

- یک نرم افزار متن باز مبتنی بر لینوکس است و بدون پرداخت هیچ هزینه ای می توان آنرا دستکاری نمود - .یک نرم افزار مبتنی بر مولفه[1] است که کار برنامه نویسی را بسیار راحت تر می کند .
- پشتیبانی از SQLite برای ذخیره داده ها که امکانات بسیار مفیدی را برای برنامه نویس فراهم می کند .
- پشتیبانی از سخت افزارهایی چون دوربین، قطب نما و شتاب سنج، Bluetooth[۱], EDGE[۲], 3G[۳], GPS[۴], GSM[۵], Wi-Fi[۶].

[1] Component-Based

- برنامه ها را می توان در لایه های امنیتی مختلفی اجرا نمود که امکان ماندگاری مناسبی داشته و مزیت مهمی برای گوشی های هوشمند خواهد بود .

- گرافیک و صدای با کیفیت قابل قبول[۷] را پشتیبانی می کند. گرافیک ها و انیمیشن های دوبعدی مبتنی بر فلش را پشتیبانی میکند و از نظر نوع گرافیک هم برداری[۸] است و همانطور که میدانید نسبت به تصاویر بیتمپ[۹] کیفیت بهتری دارد. از انیمیشن های سه بعدی مبتنی بر OpenGL هم میتوان استفاده کرد در نتیجه اندروید ابزارهای لازم برای ساخت بازی های قابل قبول را دارد.

- اندروید در حالت پیشفرض فایل های MP3, AAC , OGG , AMR , MIDI MPEG4 WAV , BMP , GIF , PNG , JPG را پشتیبانی میکند. اندروید Adobe Flashرا نیز پخش میکند و میتواند فایل های GIF متحرک را با حرکت پخش کند .برای پخش فایل های جریان دار مانند صوت و ویدئو نیز میتوانید از تگ ویدئو html5 و همچنین تکنولوژی Adobe Flash

---

[1] Open specification for short-range wireless communication
[2] Enhanced Data Rates for Global Evolution-advanced standard for wireless communication (next step up from GSM) -
[3] Third Generation
[4] Global Positioning System
[5] Global System for Mobile Communications
[6] Wireless Fidelity
[7] High Quality
[8] Vector
[9] Bitmap

Streamingاستفاده کنید.در نسخه های جدید اندروید، موتور جاوااسکریپت مرورگر کروم که سرعت بسیار بالایی در اجرای کدهای جاوا اسکریپت دارد به مرورگر اندروید متصل شده است. (در ضمن مرورگر اندروید کدهای HTML5را پشتیبانی میکند)

- پشتیبانی از سخت افزار های روز از ویژگی های مناسب دیگر اندروید است. کد های اندروید مبتنی بر جاوا بوده و توسط مترجم Dalvik ترجمه می شوند و همچنین جاوا نیز یک زبان مستقل از سخت افزار است . پشتیبانی از دستگاه های ورودی مانند کیبورد، صفحات لمسی[۱] و ترک بال[۲](گوی مسیر) نیز از ویژگی های اندروید است.

- اندروید بر خلاف سیستم عامل iOS آیفون که فقط پردازنده های ARM را پشتیبانی میکند، بر روی انواع مختلفی از پردازنده ها ARM, MIPS, Power[۳] Architecture, x86 قابل نصب است. از سال ۲۰۰۸ تاکنون تلفن های همراه متعددی با استفاده از این سیستم عامل به بازار ارائه شده اند. همچنین چندین Tablet PCنیز با استفاده از این سیستم عامل به بازار ارائه شده اند.

[1] Touch Screen
[2] Track Ball
[3] Advanced Risk Machine

- دارای مرورگر داخلی است :(Integrated browser) که منطبق بر موتور WebKit منبع باز است.

- سیستم عامل اندروید به صورت خودکار چرخه حیات[1] (طول عمر) برنامه ها را مدیریت میکند و به کمک لایه های امنیتی برنامه ها را از دسترسی غیر مجاز حفظ میکند و برنامه ها را به اصطلاح ایزوله(Isolate) میکند.

- سیستم عامل اندروید برای سخت افزارهای با ظرفیت حافظه کم و ظرفیت باطری کم بهینه شده است که پلتفرم های قبلی به این صورت بهینه نبودند.

مقدمه ای از جاوا

جاوا زبانی شبیه به C++ است (از نظر نحوی[2] و از نظر شی گرایی) . ولی از C++ کوچکتر است. –چون که در جاوا از قابلیت های غیر ضروری C که در C++ هم وجود داشت صرفنظر شده است– حال به بررسی دلایلی که باعث محبوبیت جاوا شد میپردازم: جاوا زبانی بود که در دهه ۹۰ توسط Sun Micro system توسعه داده شد و هدف آن اجرا در محیط وب (اینترنت) بود ولی حالا جای زبان های خوبی مثل C++ را گرفته

[1] Life Cycle
[2] Syntax

است و در انواع سیستم ها با معماری های مختلف کاربرد دارد. جاوا به خاطر اینکه از C++ کوچکتر است و قابلیت حمل[۱] بالاتری دارد(چون که مترجم جاوا کد ماشین تولید نمیکند بلکه کد میانی[۲] به نام بایت کد تولید میکند) و همچنین استفاده از آن ساده تر است (زیرا جاوا قوی تر است و مدیریت حافظه را به صورت خودکار خودش انجام میدهد) و نیز بر روی ویژگی های مهمی مثل امنیت[۳] و مستقل از پلتفرم[۴] بودن در آن کار شده است ، به همین خاطر ویژگی های یک زبان خوب را دارد. و این مسائل باعث فراگیر شدن این زبان شده است.

جاوا دارای مفسری است که کد میانی[۵] که اصلاحاً به آن بایت کد (Byte Code) گفته میشود برای ماشین مجازی جاوا[۶] تولید میکند.برای درک مفاهیم ماشین مجازی و بایت کد به ضمیمه کتاب مراجعه شود.

پیش نیاز های این کتاب

خوشبختانه شروع کار برای برنامه نویسی اندروید بسیار راحت است. حتی نیاز نیست که یک موبایل اندروید داشته باشید. فقط نیاز به یک کامپیوتر دارید که SDK[۱] را بر روی آن نصب کنید و یک شبیه ساز[۲] موبایل.

---

[1] Portable
[2] Intermediate code
[3] Security
[4] Platform Independent
[5] Intermediate code
[6] Java Virtual Machine

کیت توسعه نرم افزاری (Software Development Kit) اندروید قابل نصب بر روی سیستم عامل های ویندوز، لینوکس و مک OS X میباشد. بدیهی است برنامه ای که ساخته میشود قابل استفاده بر روی تمامی محصولات مبتنی بر اندروید خواهد بود. قبل از اینکه شروع به برنامه نویسی کنید احتیاج است تا جاوا، محیط توسعه (IDE) و کیت توسعه نرم افزار(SDK) را بر روی کامپیوتر خود نصب کنید.

ابزارهای لازم برای نصب و راه اندازی برنامه نویسی برای اندروید در لوح فشرده ای که به همراه این کتاب آموزشی است قرار داده شده است. برای توضیحات بیشتر به لوح فشرده مراجعه شود.این ابزار ها شامل جاوا(JDK) ، محیط توسعه Eclipse ، Android SDK و غیره است.ADT[3] هم یک افزونه[4] است که برای اکلیپس ساخته شده است و این افزونه به شما کمک میکند تا در محیط اکلیپس براحتی و به سرعت برای سیستم عامل اندروید کد بنویسید. فقط دقت نمایید که با توجه به ۳۲ بیتی یا ۶۴ بیتی بودن سیستم عامل خود ابزارهای فوق را نصب کنید. شما میتوانید با مراجعه به اینترنت هم جدیدترین نسخه های این ابزارها را به صورت رایگان دانلود نمایید. برای دانلود محیط مجتمع برنامه نویسی اکلیپس(Eclipse IDE) میتوانید به سایت زیر

http://www.eclipse.org/downloads

---

[1] Software Development Kit
[2] Emulator
[3] Android development Tools
[4] Plug-in

مراجعه کرده و آخرین نسخه آن را با توجه به سیستم عامل خود دانلود نمایید و سپس بر روی سیستم خود نصب و یا استخراج نمایید.

برای دانلود SDK نیز میتوانید به سایت زیر مراجعه فرمایید(متاسفانه کاربران ایرانی به بخش های مربوط به اندروید از سایت گوگل مجوز دسترسی ندارند!)

http://code.google.com/android/download.html

البته یکی از سایت های فارسی زبان خوب در زمینه برنامه نویسی اندروید سایت WWW.kamalan.com است که مدیر این سایت برنامه نویس فعال و محترم آقای حسام الدین کمالان است.میتوانید همانند بنده از مطالب این سایت هم کمک بگیرید.خصوصاً ایشان مطالبی را که در سایت گوگل هستند و ایرانی ها مجوز دسترسی به آن را ندارند را دانلود کرده اند و برای شما بر روی سایت خود قرار داده اند.

در صورتی که اصلاً با زبان C++ و یا زبان های شبیه به آن که شی گرا هستند(مثل C++,C#و جاوا) آشنا نیستید پیشنهاد بنده این است که یک کتاب آموزش گام به گام تهیه نمایید و بصورت موازی همراه با این کتاب مطالعه فرمایید. اگر میخواهید برای پلتفرم اندروید برنامه بنویسید شما نیاز به تجربه برنامه نویسی برای پلتفرم های قبلی موبایل ندارید. وحتی اگر تجربه برنامه نویسی برای پلتفرم های قبلی موبایل را دارید بهتر

است که آن ها را فراموش کنید و بدون تعصب[1] و مقایسه به یادگیری این پلتفرم متفاوت بپردازید.

مطالب زیر دقیقاً از سایت www.prozhe.com آورده شده ، این مطالب از طرف مصطفی شبانی برای سایت پروژه دات کام ارسال شده است . در این مطالب به نحوه تولید و توسعه سیستم عامل اندروید پرداخته و سپس به انواع نسخه های اندروید و آینده اندروید اشاره کرده است ، و میتوانید به عنوان مطالعه بیشتر بخش « **تاریخچه ای کامل از اندروید**» را بخوانید . و اگر هم عجله دارید که کار با اندروید را شروع کنید میتوانید از خواندن این مطالب صرف نظر نمایید!

تاریخچه ای کامل از اندروید

کمتر از سه سال پیش زمانی که سیستم عامل اندروید برای نخستین بار توسط کنسرسیومی به رهبری گوگل معرفی شد، کمتر کسی پیش‌بینی می‌کرد که در این مدت کوتاه این سیستم عامل موفق به پیشی گرفتن از سیستم عامل‌های پرطرفدار و جا افتاده تلفن همراه چون ویندوزموبایل، لینوکس و پالم شده و خود را به عنوان تهدیدی جدی برای رقبایی چون سیمبین، RIM و آیفون نشان دهد. اندروید پا را از این هم فراتر

[1] Open mind

گذاشته و علاوه بر حضور قدرتمند در بازار تلفن‌های همراه هوشمند، وارد عرصه‌های دیگری مانند تبلت‌ها و حتی تلویزیون نیز شده است.

به نوشته ی هومن کبیری، رشد اعجاب‌آور اندروید به گونه‌ای بوده است که بسیاری از کارشناسان پیش‌بینی می‌کنند ، این سیستم عامل تا سال ۲۰۱۲ دومین سیستم عامل پرطرفدار تلفن‌های همراه جهان خواهد بود. تخمینی که نه تنها دور از دسترس نمی‌نماید بلکه بسیار محافظه‌کارانه به شمار می‌رود. چرا که با روند رشد این سیستم عامل و اقبال شرکت‌های مختلف به آن، کسب رتبه اول نیز برای اندروید چندان دور از ذهن نیست. مروری خواهیم داشت بر تاریخچه و روند شکل‌گیری این سسیتم عامل، موفقیت‌ها و چشم‌انداز آتی آن.

پیش از ورود به اطلاعات مربوط به اندروید، نخست به نام آن می‌پردازیم. بنابر ترجمه دیکشنری کمبریج، اندروید این گونه تعریف شده است: «یک ربات (ماشینی که به وسیله کامپیوتر کنترل می‌شود) که به گونه‌ای ساخته شده تا شکل ظاهری شبیه به انسان داشته باشد.» شاید بتوان نزدیک‌ترین معنی در زبان فارسی به اندروید را آدم آهنی یا آدم ماشینی دانست.

از مدیریت شرکت کوچک اندروید تا مدیریت پروژه در خلاق‌ترین شرکت جهان

در ماه ژوئیه سال ۲۰۰۵ گوگل شرکت اندروید در پالو آلتوی کالیفرنیا را خرید. شرکت

کوچک اندروید که توسط اندی روبین، ریچ ماینرز، نیک سیرز و کریس وایت پایه‌گذاری شده بود، در زمینه تولید نرم‌افزار و برنامه‌های کاربردی برای تلفن‌های همراه فعالیت می‌کرد. اندی روبین مدیر ارشد اجرایی این شرکت پس از پیوستن اندروید به گوگل به سمت قائم‌مقام مدیریت مهندسی این شرکت و مسئول پروژه اندروید در گوگل منصوب شد.

در واقع می‌توان روبین را پایه‌گذار اندروید دانست. چرا که وی علاوه بر اینکه ایده تولید اندروید را در شرکت کوچک خود پرورش داد، در سمت مدیر این پروژه در شرکت گوگل توانست ایده خود را پیاده‌سازی کند و سیستم عامل اندروید را با نام شرکت کوچک پیشین خود روانه بازار نماید.

تیم اندروید به رهبری روبین فعالیت خود را برای تولید پلتفرم موبایل مبتنی بر کرنل لینوکس آغاز کردند. درز اخباری از فعالیت‌های این تیم به خارج از گوگل، سبب بروز شایعاتی مبنی بر تمایل گوگل به تولید تلفن همراه در اواخر سال ۲۰۰۶ گردید. این شایعات زمانی بیشتر قوت گرفت که در سپتامبر ۲۰۰۷ نشریه اینفورمیشن ویک در گزارشی خبر از ثبت چندین حق امتیاز و اختراع در حوزه تلفن همراه توسط گوگل داد.

تولد یک آدم آهنی!

با اعلام زمان کنفرانس خبری شرکت گوگل در نوامبر سال ۲۰۰۷ دیگر تمامی رسانه‌ها و افکار عمومی جهان چشم انتظار مشاهده نخستین تلفن همراه ساخت گوگل بودند. ولی غافلگیری بزرگ رخ داد. هیچ خبری از «یک» گوشی تلفن همراه نبود بلکه خبر داغ آن روز در مورد ورود صدها تلفن همراه در سال‌های پیش رو بود که توسط شرکت‌های مختلف تولید می‌شد. «اتحادیه گوشی باز» یا Open Handset Alliance در روز ۵ نوامبر ۲۰۰۷ اعلام موجودیت کرد.

۳۴ شرکت فعال در زمینه تولید نرم‌افزار، تولید کننده های تلفن همراه، اپراتور تلفن همراه و تولید کننده نیمه رساناها و پردازنده‌های تلفن همراه اعضای مؤسس این اتحادیه بودند. در میان نام‌های مشهور در بین اعضای مؤسس، شرکت‌هایی چون سامسونگ، LG، موتورولا، HTC، T-Mobile، NTT DoCoMo[1]، اینتل، Nvidia، تگزاس اینسترومنتس، کوآلکام، برادکام، تلفونیکا، اسپرینت، eBay و البته گوگل به چشم می‌خوردند. اریک اشمیت مدیر ارشد اجرایی گوگل در این مراسم گفت: «اعلام امروز بسیار جاه‌طلبانه‌تر از معرفی تنها «یک» تلفن گوگلی است که در چند هفته اخیر توسط رسانه‌ها پیش‌بینی شده بود.از دیدگاه ما پلتفرمی که ما ارائه کرده‌ایم، هزاران تلفن گوناگون را به بازار روانه خواهد کرد.»

[1] **Nippon Telegraph and Telephone**(telephone company dominating the telecommunication market in Japan)

نخستین گوشی مبتنی بر اندروید توسط شرکت HTC با همکاری T-Mobile تولید شد. این گوشی که به فاصله کمتر از یک سال از تشکیل اتحادیه Open Handset Alliance یعنی در ۲۲ اکتبر ۲۰۰۸تولید شد، در بازارهای مختلف به نام‌های HTC Dream، T-Mobile G1 و Era G1 به بازار عرضه گردید.

آدم آهنی تقویت می‌شود

نهم دسامبر ۲۰۰۸ روز تاریخی دیگری برای اندروید بود. در این روز ۱۴ عضو جدید از نام‌های معروف صنعت تلفن همراه جهان به اتحادیه Open Handset Alliance پیوستند. در بین این نام‌ها باید به سونی اریکسون، اریکسون، توشیبا، آسوس، گارمین، هوآوی و آرم اشاره کرد. روند پیوستن شرکت‌های بزرگ به اتحادیه تا به امروز نیز ادامه داشته است و شرکت‌هایی چون ایسر، آلکاتل، لنوو، شارپ، فاکسکان، NEC، کیوسرا، NXP، ST-Ericsson، مارول، ZTE و دل نیز از جمله شرکت‌هایی بوده‌اند که به جمع پشتیبانی کنندگان اندروید پیوسته‌اند.

کپی‌رایت و حق امتیاز

حق امتیاز اندروید به صورت اپن سورس بر اساس حق امتیاز آپاچی یا Apache License ارائه می‌گردد. بر این اساس شرکت‌های عضو اتحادیه می‌توانند با دسترسی به کدهای اصلی اندروید آن را مطابق دلخواه خود تغییر دهند و کد تغییر یافته را بدون عودت دادن برای خود حفظ کنند.

ویرایش‌های اندروید با طعم شیرینی‌جات و دسرها!

گوگل ویرایش‌های گوناگون اندروید را علاوه بر شماره ویرایش با نام یک شیرینی یا دسر معرفی می‌کند. این نام البته از ترتیب حروف الفبا برای حرف اول آن نام نیز پیروی می‌کند به گونه‌ای که ویرایش‌های منتشر شده اندروید تا به امروز به این نام‌ها بوده‌اند:

Cupcake که نوعی کیک کوچک شبیه به کیک یزدی ایرانی است ولی با اندازه‌ای کمی بزرگ‌تر ، که نامی است برای ویرایش ۱.۵ اندروید،

Donut که در ایران هم به همان نام شهرت دارد و نوعی پیراشکی محسوب می‌شود، که نامی است برای ویرایش ۱.۶ اندروید،

Éclair که نوعی شیرینی خامه‌ای است شبیه به لطیفه ولی با اندازه بزرگ‌تر، که نامی است ، برای ویرایش ۲ اندروید و برای نسخه ۲.۱ به آن Éclair MR1[1] میگویند.

[1] Éclair Maintenance Release1

FroYo (مخفف Frozen yogurt) نوعی دسر است که با ماست یخ زده تهیه می‌شود و نامی است برای ویرایش ۲.۲ .

نام ویرایش بعدی اندروید هم Gingerbread یا نان زنجبیلی گذاشته شده است. همان گونه که مشاهده می‌شود ترتیب نام‌های شرینی‌ها و دسرها بر اساس حروف الفبا است. حالا که طعم این ویرایش‌ها را چشیدیم شاید بهتر باشد سری هم به ویژگی‌های فنی آنها بزنیم.

آغاز سال نو میلادی ۲۰۱۱ همراه با معرفی نسخه جدیدی از سیستم عامل اندروید از طرف گوگل بود که بنام اندروید ۳ یا Honey-Comb (شانه عسل) نامگذاری شد. این محصول در نمایشگاه CES لاس وگاس به معرض نمایش گذاشته و به مردم معرفی شد. این محصول بطور خاص برای تبلتها یا محصولات با صفحه نمایش بزرگ طراحی شده است.

اندروید نسخه ۱.۵ یا Cupcake

نسخه ۱.۵ اندروید نخستین نسخه‌ای بود که به طور رسمی منتشر شد.(قبل از Cupcake نسخه Base ساخته شد که آزمایشی بود و در واقع نسخه تحقیقاتی بود) این نسخه اندروید مبتنی بر کرنل لینوکس بود. از جمله قابلیت‌هایی که در این ویرایش گنجانده شده بود، باید به موارد زیر اشاره کرد :

- امکان ضبط فیلم از طریق دوربین فیلمبرداری آن
- فرستادن فیلم به سایت YouTube و عکس به سایت Picasa به صورت مستقیم از روی گوشی
- صفحه کلید مجازی با قابلیت پیش‌بینی کلمات وارد شده
- پشتیبانی از پخش استریوی موسیقی از طریق بلوتوث (A2DP) و کنترل پخش موسیقی یا ویدیو از طریق بلوتوث (AVRCP).
- قابلیت اتصال اتوماتیک به دستگاه های بلوتوث
- امکان شخصی‌سازی صفحه اصلی با استفاده از ویجت‌ها و یا پرونده‌های شخصی
- جابجایی انیمیشنی تصاویر به هنگام عوض شدن صفحات

اندروید نسخه ۱.۶ یا Donut

در ۱۵ سپتامبر ۲۰۰۹ اندروید نسخه ۱.۶ یا دونات منتشر شد. این نسخه اندروید مبتنی بر کرنل لینوکس نسخه ۲،۶،۲۹ بود و قابلیت‌های زیر را به اندروید افزود:

- بهبود در سرویس اندروید مارکت(بازار اندروید)
- رابط کاربری یکپارچه برای دوربین عکسبرداری، دوربین فیلمبرداری و گالری تصاویر
- امکان انتخاب چند عکس برای پاک کردن در منوی گالری

- به‌روزرسانی ویژگی جست‌وجوی صوتی
- به‌روزرسانی ویژگی جست‌وجو با قابلیت جست‌وجو در موارد نشانه‌گذاری شده (Bookmarks)، تاریخچه (History)، اسامی (Contacts) و وب از صفحه اصلی[1]
- پشتیبانی از تکنولوژی‌های به‌روز شده CDMA/EVDO، x۸۰۲،۱، VPN و موتور تبدیل متن به صدا[2]
- پشتیبانی از رزولوشن WVGA برای صفحه نمایش
- افزوده شدن قابلیت‌های حرکتی در سیستم عامل و ابزار برنامه‌نویسی برای برنامه‌نویسان

نسخ ۲ و ۲.۱ یا Éclair

هر دو نسخه ۲ و ۲.۱ اندروید مانند نسخه ۱.۶ مبتنی بر کرنل لینوکس طراحی شده‌اند. اندروید ویرایش ۲ در ۲۶ اکتبر ۲۰۰۹ معرفی شد. در سوم دسامبر ۲۰۰۹ SDK نسخه ۲،۰،۱ معرفی شد(در نسخه های مثل ۲.۰.۱ که عدد دوم صفر ولی عدد سوم یک است منظور این است که بیس و پایه سیستم عامل عوض نشده و امکانات جدیدی هم به آن اضافه نشده است و فقط باگ[3] های آن بر طرف شده است). SDK ویرایش ۲.۱ در ۱۲ ژانویه ۲۰۱۰ منتشر گردید. اهم امکانات اضافه شده در این نسخ به شرح زیر هستند:

---

[1] Home Screen
[2] Text to Speech
[3] Bug

- سرعت سخت‌افزاریِ بهبود یافته
- ویژگی چند لمسی Multi Touch
- پشتیبانی از رزولوشن‌های بیشتر برای صفحه نمایش
- رابط کاربری به‌روزرسانی شده
- مرورگر اینترنتی با قابلیت پشتیبانی از HTML5
- دفترچه تلفن به‌روزرسانی شده
- گوگل مپ نسخه ۳،۱،۲
- پشتیبانی از Microsoft Exchange
- افزوده شدن امکان فلاش داخلی برای دوربین
- افزوده شدن زوم دیجیتال دوربین
- به‌روزرسانی صفحه کلید مجازی
- پشتیبانی از بلوتوث نسخه ۲.۱
- اضافه شدن قابلیت کاغذ دیواری‌های متحرک
- اضافه شدن امکان ارسال فایل با استفاده از بلوتوث

نسخه ۲.۲ یا FroYo

اندروید نسخه ۲.۲ در ۲۰ مه ۲۰۱۰ معرفی شد. این ویرایش اندروید مبتنی بر کرنل لینوکس است و قابلیت‌های زیر به آن اضافه شده است :

• افزایش سرعت سیستم عامل، حافظه و عملکرد سیستم بین ۲ تا ۵ برابر نسخه ۲

• افزایش سرعت اجرای برنامه‌های کاربردی با استفاده از تکنیک‌های JIT[1](ترجمه در زمان اجرا متدی بود که برای رفع مشکل کندی جاوا استفاده شد.چون که جاوا برای رسیدن به هدف مستقل از ماشین بودن[2] باید به صورت مفسری کار میکرد و مفسر هم همواره کندتر از کامپایلر بوده است. به همین خاطر تکنیک ترجمه در زمان اجرا برای رفع این مشکل توسط شرکت سان ساخته شد).

• اضافه شدن موتور جاوا اسکریپت V8 کروم به مرورگر اینترنتی

• افزایش پشتیبانی از Microsoft Exchange با قابلیت‌هایی چون سیاست حریم شخصی به‌روز شده، همسان‌سازی تقویم و ...

• اندروید مارکت به‌روز شده با قابلیت به‌روزرسانی خودکار برنامه‌های کاربردی

• شماره‌گیری صوتی و انتقال دفترچه تلفن از طریق بلوتوث

• امکان نصب برنامه‌های کاربردی بر روی حافظه‌های جانبی

• پشتیبانی از فلش نسخه و بهبود عملکرد دوربین در حالت‌های عکس و فیلمبرداری

---

[1] Just In Time
[2] Machine Independent

نان زنجبیلی – نسخه ۲.۳

نام ویرایش۲.۳ اندروید «Gingerbread یا نان زنجبیلی» است . در این ویرایش هم امکانات جدیدی به سیستم عامل اضافه شد وبسیاری از باگ های سیستم عامل رفع شد .

شانه عسل یا نسخه ۳.۰

واسط کاربری جدید این سیستم عامل بصورت کامل و از اول پیاده سازی شده ست. طراحی صفحه برنامه ها (اپلیکیشن ها) مانند iOS از شفافیت خاصی برخوردار است. به طور کلی در این نسخه تغییرات بنیادی انجام شده است مثل تغییر ظاهر کیبرد ورودی(شکل گرافیکی کلیدها تغییر کرده و بمنظور راحتتر نویسی، جابجایی هایی هم در کلیدها انجام شده است. اینکار باعث شده تا حتی در سرعت های بالا کلیدها بهتر دیده شوند و تایپ حروف راحتتر انجام شود). و تغییر مرورگر سیستم عامل و شبیه شدن آن به مرورگر کروم. و نیز در این نسخه از سنسورهای بیشتری پشتیبانی شده است تا به برنامه نویسان قدرت بیشتری در ساخت برنامه ها داشته باشند. و خیلی تغییرات دیگر که در اینجا به آن اشاره نشده است.(به قول معروف شنیدن کی بود مانند دیدن!)

میزان محبوبیت نسخه‌های مختلف اندروید

آخرین آمار منتشره از سوی گوگل در خصوص میزان محبوبیت نسخ مختلف اندروید نشان می دهد که طعم شیرینی خامه‌ای برای کاربران دلچسبتر بوده است. عمده‌ترین دلیل این امر هم ارائه نسخه بروزرسانی به ویرایش ۲.۱ از سوی موتورولا برای پرطرفدارترین گوشی اندروید یعنی دروید بوده است.

بر اساس آمار منتشر شده، در هفته منتهی به شانزدهم ژوئن ۲۰۱۰، نیمی از گوشی‌های اندروید موجود در بازار به سیستم عامل نسخه ۲.۱ یا همان Éclair مجهز بوده‌اند و پس از آن نسخه ۱.۶ با ۲۵ درصد محبوب‌ترین نسخه بوده است که با فاصله کمی به نسبت نسخه ۱.۵ در جایگاه دوم قرار گرفته است. سایر نسخ اندروید هم در مقایسه با این سه نسخه سهمی بسیار ناچیز دارند بگونه‌ای که مجموع سهم بازار سایر نسخ اندروید تنها ۵/۰ درصد سهم بازار را تشکیل می‌دهد.

سرعت انتشار ویرایش‌های اندروید

اندروید با سرعت اعجاب‌آوری در حال پیشرفت است. در کمتر از ۱ سال و از سپتامبر ۲۰۰۹ چهار ویرایش اصلی این سیستم عامل یعنی ویرایش‌های ۱.۶، ۲، ۲.۱ و ۲.۲ منتشر شده است. این امر باعث شده تا تنها برخی از شرکت‌ها که به طور متمرکز و با تمام توان بر روی این سیستم عامل کار می‌کنند، مانند موتورولا و HTC، بتوانند همگام با ارائه ویرایش‌های جدید اندروید گوشی‌های خود را به‌روز کنند ولی سایر شرکت‌ها رفته رفته

در حال عقب افتادن از این قافله هستند. به عنوان مثال باید به شرکت سونی اریکسون اشاره کرد.

این شرکت نخستین گوشی اندرویدی خود را با نام XPERIA X10 که مبتنی بر اندروید نسخه ۱.۶ است، به بازار معرفی کرد. سونی اریکسون رابط کاربری ویژه خود و امکانات ابتکاری فراوانی به X10 افزوده است. اوایل بهار سال جاری سونی اریکسون با خوشحالی اعلام کرد که قصد دارد تا پایان سال جاری میلادی گوشی‌های X10 خود را با نسخه ۲.۱ اندروید به‌روزرسانی کند. کمتر از دو هفته بعد نسخه ۲.۲ اندروید منتشر شد و شرکت‌های موتورولا و گوگل در همان زمان اعلام کردند که گوشی‌های دروید و نگزوس وان خود را تا یک ماه بعد به اندروید ۲.۲ مجهز خواهند ساخت.

اتفاقی که شاید ۶ ماه بعد برای X10 سونی اریکسون بیفتد! چنین وضعیتی برای شرکت‌های بزرگی چون سامسونگ و ال‌جی نیز وجود دارد. این شرکت‌ها هم هنوز نتوانسته‌اند سرعت واکنش خود را با سرعت سرسام‌آور پیشرفت اندروید هماهنگ سازند. در صورتی که این شرکت‌ها نتوانند به چنین هماهنگی دست یابند، شکاف بین تولیدکنندگان پیشروی اندروید یعنی موتورولا و HTC با بقیه بسیار بیشتر خواهد شد.

در هر حال به نظر می‌رسد گوگل باید فکری به حال فاصله ایجاد شده بین رقبا بکند وگرنه بروز این ناهماهنگی بازار را نیز دچار آشفتگی خواهد کرد. همان گونه که در سطور پیشین نیز ذکر شد در حالی که گوگل خود را برای ارائه نان زنجبیلی یعنی نسخه

بعد از ۲.۲ اندروید آماده می‌کند، هنوز نیمی از دستگاه‌های اندروید فعال، از نسخه‌های پایین‌تر از ۲.۱ استفاده می‌کنند و این بدان معنا است که گوگل بسیار سریع‌تر از یارانش در اتحادیه Open Handset Alliance حرکت کرده است و آنها نتوانسته‌اند خود را با آن همراه سازند.

سهم بازار اندروید در مقایسه با سایر سیستم‌های عامل تلفن‌های هوشمند، رشد اعجاب‌آور اندروید را نشان می‌دهد. اندروید برای نخستین بار در سه ماهه اول سال ۲۰۱۰ توانست گوشی‌های بیشتری از مهم‌ترین رقیب خود یعنی اپل به فروش برساند. برخی کارشناسان بر این باور هستند که اگر گوگل موفق شود اپل را از پیش روی خود بردارد RIM سیمبین را نیز پشت سر خواهد گذاشت. گروهی از کارشناسان، استراتژی اندروید در مقابله با اپل را با استراتژی مایکروسافت در اوایل دهه ۱۹۷۰ مقایسه می‌کنند.

جایی که مایکروسافت توانست با فروش حق امتیاز استفاده از سیستم عامل خود به سایر شرکت‌ها به سلطه مکینتاش خاتمه دهد و حالا گوگل به همین استراتژی و به کمک بزرگ‌ترین تولید کنندگان تلفن همراه مانند سامسونگ، ال‌جی، سونی اریکسون، موتورولا و اچ تی سی، قصد دارد روند رشد آیفون اپل را متوقف سازد و به نظر می‌رسد تا حد زیادی هم موفق بوده است.

ذکر این نکته ضروری است که نخستین گوشی آیفون در ۲۹ ژوئن ۲۰۰۷ به بازار عرضه شد در حالی که نخستین گوشی مبتنی بر اندروید بیش از یک سال بعد و در اکتبر ۲۰۰۸

روانه بازار شد. اما به غیر از اپل بقیه رقبا نیز از دست اندروید جان به در نبرده‌اند. اندروید در سه ماهه نخست سال ۲۰۱۰ توانست سهم بازار خود را از ۶/۱ درصد در مدت زمان مشابه در سال گذشته به ۶/۹ درصد برساند و با پشت سر گذاشتن ویندوز موبایل و لینوکس در رده چهارم پرطرفدارترین سیستم عامل تلفن‌های همراه هوشمند قرار گیرد.

با اقبال بیشتر سایر تولید کنندگان به گوشی‌های اندروید به نظر می‌رسد روند رشد این سیستم عامل نه تنها کند نگردد بلکه شتاب بیشتری نیز پیدا کند. تاکنون بالغ بر ۶۱ مدل دستگاه مبتنی بر اندروید با ۲۱ برند مختلف تولید شده است. بنابر آخرین گزارش‌ها در حال حاضر هر روز یکصدهزار گوشی مبتنی بر اندروید به فروش می‌رسد. با نرخ کنونی گوگل ۳۶ میلیون گوشی در سال به فروش خواهد رساند. این رقم زمانی معنا پیدا می‌کند که بدانیم شرکت اچ تی سی، چهارمین تولید کننده تلفن‌های همراه هوشمند در جهان سالانه ۱۷ میلیون گوشی تلفن همراه به فروش می‌رساند. اچ تی سی پیش از این ۷۰ درصد گوشی‌های مبتنی بر اندروید را تولید می‌کرد.

رقمی که اکنون به زحمت به ۵۰ درصد می‌رسد. با نرخ کنونی و در صورت ثابت ماندن نرخ فروش آیفون، اندروید خواهد توانست اپل را نیز پشت سر گذاشته و خود را به عنوان تهدیدی جدی برای RIM مطرح کند. شرکت‌های بزرگ تولید تلفن همراه اعلام کرده‌اند قصد دارند تولید گوشی‌های مبتنی بر اندروید خود را شتاب بخشند. فعال‌ترین تولیدکننده گوشی‌های مبتنی بر اندروید یعنی موتورولا اعلام کرده تا پایان سال جاری

میلادی ۲۰ مدل گوشی مبتنی بر اندروید به بازار عرضه خواهد کرد. شرکت ال‌جی هم اعلام کرده است قصد دارد همین تعداد گوشی را تا پایان سال جاری با سیستم عامل اندروید به بازار عرضه نماید.

سامسونگ دومین تولیدکننده تلفن‌های همراه در جهان هم اعلام کرد نیمی از گوشی‌های تلفن همراه هوشمند این شرکت در سال ۲۰۱۰ مبتنی بر اندروید خواهند بود. بقیه تولیدکنندگان هم هر روز علاقه بیشتری به تولید گوشی‌های تلفن همراه مبتنی بر اندروید از خود نشان می‌دهند. با این اوصاف انتظار می‌رود اندروید بتواند جهشی شگرف در سهم بازار را رقم زند.

دستگاه‌های شاخص مبتنی بر اندروید

در بین شرکت‌های تولید کننده تلفن‌های همراه هوشمند مبتنی بر اندروید دو شرکت موتورولا و HTC تحرک بسیار گسترده‌تری به نسبت سایر رقبا دارند. موتورولا که زمانی دومین تولیدکننده تلفن همراه جهان بود، پس از زیان‌دهی در دوره‌های مالی متوالی و کاهش چشمگیر سهم بازار در موقعیتی بحرانی قرار گرفته بود، با تغییر ناگهانی استراتژی خود و کنار نهادن سایر سیستم‌های عامل تلفن همراه از قبیل ویندوز موبایل، سیمبین و لینوکس موبایل، تمامی تلاش خود را بر طراحی گوشی‌های مبتنی بر اندروید معطوف ساخت.

این شرکت توانست با استراتژی جدید خود چندین مدل گوشی هوشمند مبتنی بر اندروید به فروش برساند و با این کار پس از مدت‌ها سودآوری را تجربه کند. موتورولا رابط کاربری ویژه اندروید مختص به خود را با نام MOTOBLUR طراحی کرده و بر روی گوشی‌های خود قرار داده است. به نظر می‌رسد با توجه به توانمندی‌های این شرکت که خالق تلفن همراه به شمار می‌رود، بتوان از هم اکنون موتورولا را طلایه‌دار اندروید دانست. در خصوص HTC هم باید گفت هر چند این شرکت به موازات تولید گوشی‌های مبتنی بر اندروید به تولید گوشی‌های با سیستم عامل ویندوز موبایل هم می‌پردازد ولی رفته رفته، توان خود را بیشتر بر تولید گوشی‌های مبتنی بر اندروید هدایت می‌کند.

HTC علاوه بر تولید گوشی با برند خود به تولید گوشی‌های با برند سایر شرکت‌ها هم می‌پردازد و گوشی نگزوس وان شرکت گوگل یکی از همین نمونه‌ها است. گوگل تولید نخستین گوشی با نام تجاری خود را پس از اینکه شرکت سونی اریکسون از تولید آن با برند گوگل سر باز زد به HTC سپرد.

اندروید مارکت

اندروید مارکت سرویس فروش نرم‌افزارهای کاربردی برای گوشی‌های اندروید است. یک برنامه کاربردی ویژه اندروید مارکت به صورت از پیش بارگذاری شده بر روی

گوشی‌های اندروید نصب گردیده و به کاربران امکان می‌دهد نرم‌افزارهای مورد نیاز خود را خریداری و دانلود کنند. البته تمامی نرم‌افزارهای موجود در اندروید مارکت فروشی نیستند بلکه بیش از نیمی از نرم‌افزارهای موجود در اندروید مارکت به صورت رایگان عرضه می‌شوند و از این نظر اندروید بیشترین درصد نرم‌افزارهای رایگان را در بین تمامی سیستم‌های عامل تلفن‌های همراه هوشمند در اختیار کاربران قرار می‌دهد.

هر برنامه‌نویس با ثبت نام، امکان فروش برنامه‌های خود در اندروید مارکت را دارد. ۷۰ درصد از مبلغ فروش برنامه‌های کاربردی به برنامه‌نویس تعلق می‌گیرد و ۳۰ درصد مابقی بین اپراتورها توزیع می‌شود. بر اساس سیاست‌های گوگل در حال حاضر تمامی برنامه نویسان عضو پروژه اندروید از سراسر جهان می‌توانند برنامه‌های کاربردی رایگان خود را از طریق اندروید مارکت در ۴۶ کشور عرضه کنند. برای اینکار کافی است برنامه نویسان فرمی مختصر را تکمیل کرده و البته ۲۵ دلار حق عضویت هم به گوگل بپردازند.

ولی تنها برنامه‌نویسان ساکن در نه کشور اتریش، فرانسه، آلمان، ایتالیا، ژاپن، هلند، اسپانیا، انگلستان و ایالات متحده آمریکا می‌توانند برنامه‌های خود را برای فروش در ۱۳ کشور استرالیا، اتریش، کانادا، فرانسه، آلمان، ایتالیا، ژاپن، هلند، نیوزلند، اسپانیا، سوییس، انگلستان و ایالات متحده آمریکا در معرض بازدید خریداران قرار دهند و سایر کشورها امکان مشاهده و خرید برنامه‌های غیر رایگان را ندارند.

نکته جالب توجه در این زمینه عدم حضور حتی یک کشور از منطقه خاورمیانه و شمال آفریقا (به غیر از اسرائیل) در لیست چهل و شش کشوری است که امکان دسترسی به اندروید مارکت را دارند! این در حالی است که گوشی‌های مجهز به اندروید از سوی اغلب شرکت‌های بزرگ از جمله موتورولا، HTC، سونی اریکسون و ال جی در این منطقه مدت‌ها است که به بازار عرضه شده‌اند!

تعداد برنامه‌های کاربردی موجود در اندروید مارکت تاکنون نزدیک به ۵۰۰۰۰ بوده است این در حالی است که برنامه‌های کاربردی موجود در iTunes برای گوشی آیفون به ۱۷۵۰۰۰ برنامه بالغ می‌شود و از این منظر به نظر می‌رسد که اندروید راهی دراز برای سبقت گرفتن از آیفون در پیش دارد. ولی در هر حال باید این نکته را هم در نظر داشت که هرچند تعداد برنامه‌های کاربردی نشانگر اقبال برنامه‌نویسان به پلت فرم موردنظر است، ولی تعداد بسیار زیاد برنامه‌ها برای کاربران همیشه هم خوب نیست.

چرا که آنان را مجبور می‌سازد تا جستجویی سخت و طاقت فرسا را برای دستیابی به برنامه مورد علاقه خود تجربه کنند. گوگل کوشیده است دسترسی به برنامه‌های کاربردی اندروید را به شدت محدود ساخته و تمامی دسترسی کاربران را از طریق نرم‌افزار اندروید مارکت نصب شده بر روی گوشی کاربران کانالیزه نماید. گوگل تا بدانجا پیش رفته است که حتی دسترسی به اطلاعات و امکان جست‌وجوی برنامه‌های

کاربردی اندروید از وب سایت رسمی اندروید مارکت از طریق کامپیوترهای شخصی، امکانپذیر نیست.

از طرف دیگر عدم امکان نصب نرم‌افزارهای دانلود شده بر روی کارت حافظه محدودیت دیگری است که کاربران برای دانلود برنامه‌ها از هر جای دیگری به غیر از اندروید مارکت با آن مواجه هستند. اما هنوز هم راه‌هایی برای دور زدن گوگل برای دستیابی به برنامه‌های کاربردی وجود دارد. بسیاری از برنامه‌نویسان از جمله شرکت‌های تولید برنامه‌های کاربردی نسخه قابل نصب برنامه‌های خود را علاوه بر اندروید مارکت از طریق وب سایت‌های خود نیز در اختیار کاربران قرار می‌دهند.

علاوه بر این برخی وب‌سایت‌ها اقدام به جمع‌آوری و در اختیار قراردادن برنامه‌های کاربردی پرطرفدار اندروید نموده‌اند. نمونه‌ای از این سایت‌ها Androlib و Androidzoom هستند. همچنین با نصب یک نرم‌افزار بر روی گوشی خود امکان خواهید یافت برنامه‌های دانلود شده بر روی کارت حافظه خود را بر روی گوشی نصب نمایید.

اگرچه با استفاده از راه‌هایی که گفته شد امکان نصب برخی نرم‌افزارها بر روی گوشی خود را خواهید داشت، ولی باید اذعان کرد که اغلب برنامه‌های اندروید به ویژه برنامه‌های اصلی آن که توسط خود گوگل طراحی شده‌اند، مانند نقشه‌های گوگل یا برنامه گاگلز (Goggles) تنها از طریق اندروید مارکت قابل دسترسی هستند. پس اگر

می خواهید به برنامه های کاربردی اصلی اندروید دسترسی داشته باشید، باید از اندروید مارکت نصب شده بر روی گوشی خود استفاده کنید.

آینده اندروید

خوب اگر فکر کرده‌اید که کار شما با اندروید تمام شده است سخت در اشتباه هستید. اگر کار شما هم با اندروید تمام شده باشد، کار اندروید با شما تمام نشده است. بله این آدم آهنی سبز رنگ پس از رسوخ در تلفن‌های همراه شما قصد دارد وارد تلویزیون‌های شما هم بشود. به چشمان خود شک نکنید. درست خوانده‌اید اندروید به زودی در تلویزیون‌های شما نیز خواهد بود. در همایش Google I/O در ماه مه ۲۰۱۰ شرکت‌های گوگل، سونی، اینتل، لاجیتک، بست بای، ادوبی و دیش نتورک از عرضه تلویزیون‌های مبتنی بر اندروید خبر دادند.

تلویزیون‌هایی که به طور بی‌سیم به اینترنت متصل می‌شوند و علاوه بر اینکه امکان اتصال به شبکه‌های آنلاین پخش فیلم را دارند، از برنامه‌های کاربردی که برای نصب بر روی این تلویزیون‌ها تهیه می‌شوند نیز بهره خواهند برد.

مسلماً تب اندروید به این زودی فروکش نخواهد کرد. هجوم بی‌سابقه شرکت‌ها برای تولید محصولات مبتنی بر اندروید رفته رفته طیف گسترده‌تری از محصولات شامل تلفن همراه، تلویزیون، نت‌بوک و تبلت را در بر می‌گیرد. به نظر می‌رسد همه چیز به کام

اندروید است و به سختی می‌توان تصور کرد سیطره این پدیده به سادگی قابل شکستن باشد. آدم آهنی بازیگوش سبز رنگ ما در مدت زمان کوتاهی که از تولدش می‌گذرد، نشان داده هر روز به دنبال غافلگیر کردن ما و سرک کشیدن به یکی دیگر از وسایل الکترونیکی ماست تا آن را نیز جولانگاه شیطنت‌های دوست داشتنی خود نماید.

یادداشت :

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

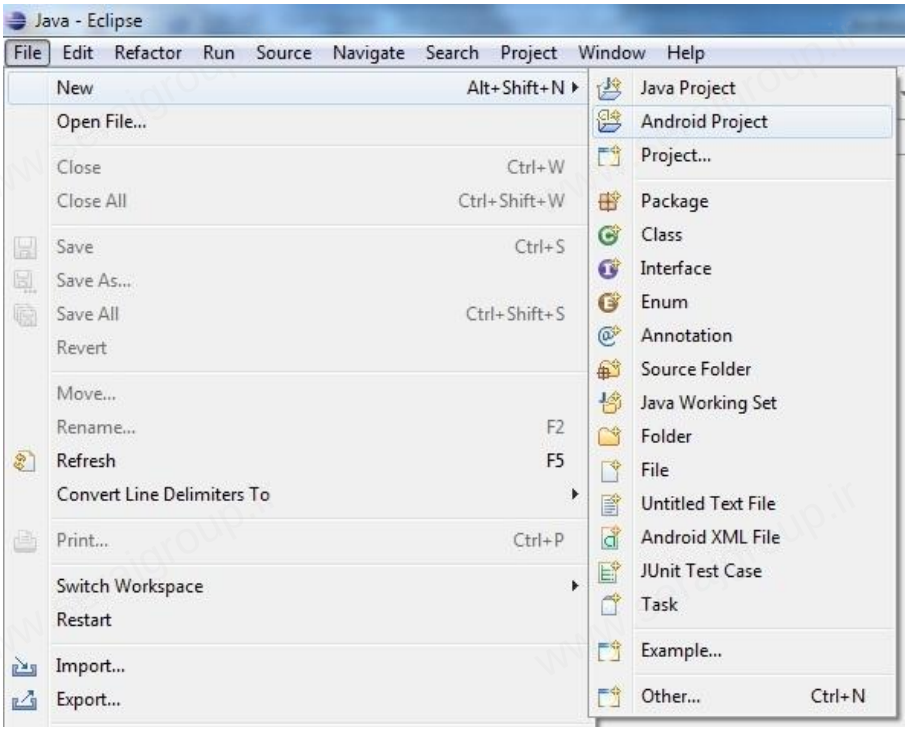
.....

فصل اول : شروع کار با اکلیپس

مطالبی که در این فصل با آن آشنا میشوید :

ایجاد پروژه جدید

بعد از نصب Eclipse در محلی از کامپیوتر خود ، این برنامه را اجرا کرده و از منوی File گزینه New را انتخاب کرده و سپس Android Project را انتخاب کنید.(شکل ۱-۱)



شکل ۲-۱

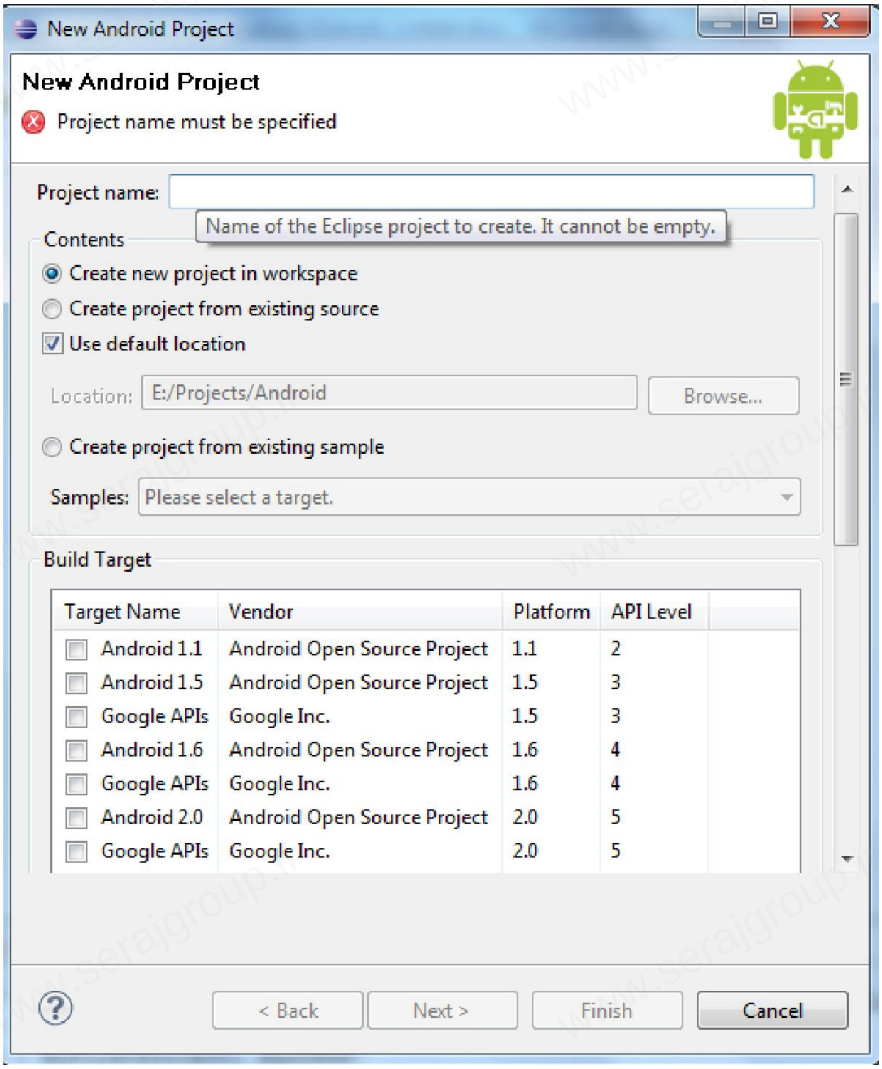
سپس فرم زیر را مشاهده میکنید.(شکل۱-۲)در این فرم بایستی نام پروژه را مشخص کنید. ونیز محلی که پروژه ذخیره میشود. و سپس نسخه[1] SDK را مشخص میکنید.در مورد نسخه های SDK اشاره به این مطلب ضروری است که اگر ما نسخه پایین تر را انتخاب کنیم برنامه ما بر روی گوشی های بیشتری اجرا میشود(از گوشی های قدیمی تا گوشی های جدید برنامه ما را اجرا میکنند).اما گوشی های جدید قابلیت های جدید دارند و این قابلیت ها در گوشی های قدیمی وجود ندارد و بدیهی است که SDK مربوط به گوشی های قدیمی هم این قابلیت ها را ندارند.پس در این صورت ناچارید که از نسخه های بالاتر استفاده کنید حتی به قیمت این که بسیاری از کاربرانی که گوشی های قدیمی دارند را از دست میدهید (یعنی برنامه شما برای آنها اجرا نمیشود).پس SDK هم باید با دقت انتخاب شود و با سبک سنگین کردن ورژن آن را مشخص کنید.بعد از مشخص کردن نام پروژه باید نام برنامه(Application Name) را هم مشخص نمایید.سپس نام بسته (Package Name) را مشخص کنید.(شکل ۱-۳)

توجه کنید که نام بسته بایستی حداقل دو قسمتی باشد و قسمت ها با نقطه[2] از هم جدا شوند.گزینه Min SDK Version هم به صورت خودکار توسط برنامه پر میشود ولی اگر پر نشد شما میتوانید از لیست نسخه های مربوط به SDK ، به ستون API Level مربوط به SDK ای که انتخاب کرده اید بروید و عدد آن را در این قسمت بنویسید.لزوم

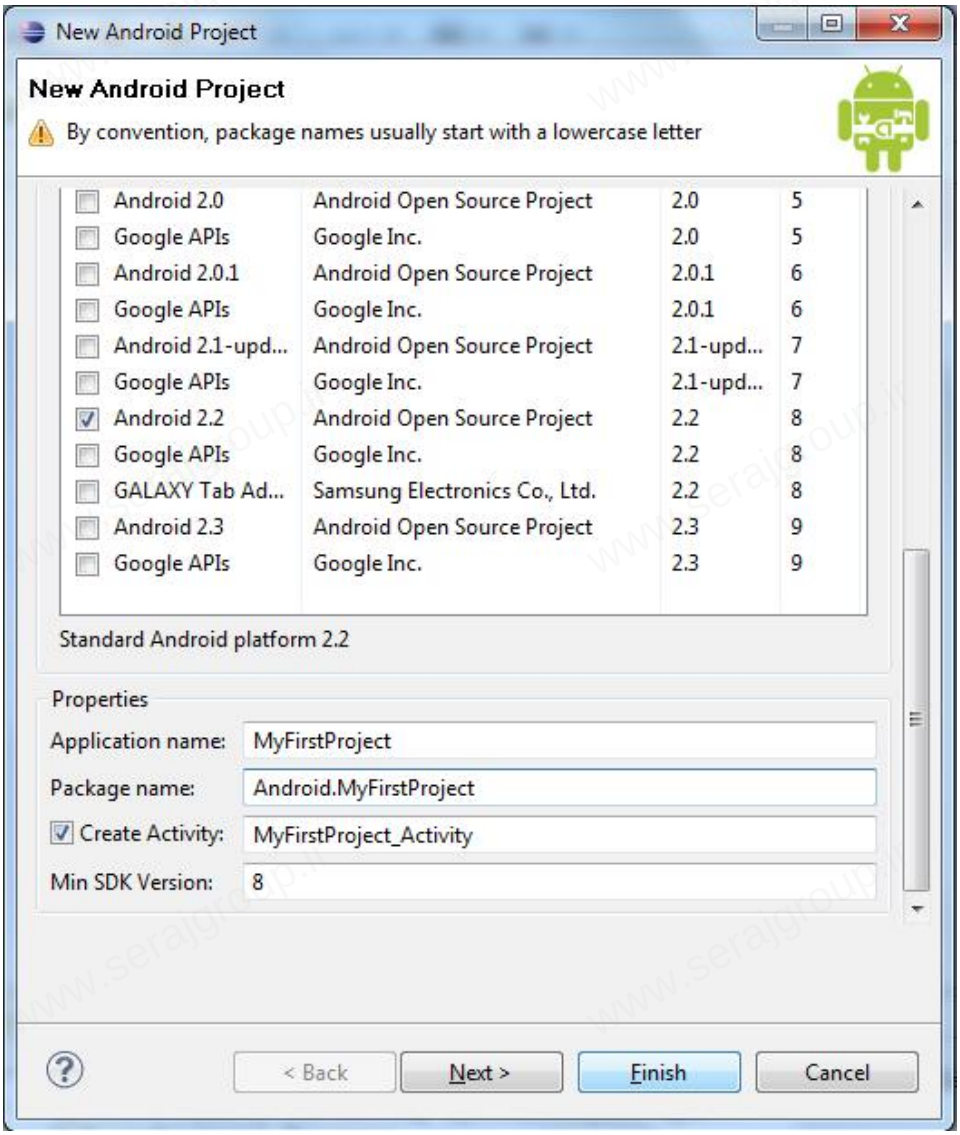
---

[1] Version(SDK Version)
[2] Dot

این کار به این خاطر است که این عدد همواره با برنامه شما خواهد بود تا در صورتی که شما آن را در مارکت ها برای نصب قرار دادید برنامه مارکت با بررسی این عدد حدس میزند(نتیجه میگیرد) که برنامه شما برای گوشی کاربری که قصد دانلود این برنامه و نصب آن را دارد قابل اجرا است یا خیر؟ برای مثال اگر شما این عدد را ۸ بدهید به این معنی است که اگر نسخه سیستم عامل اندروید گوشی مقصد کمتر از ۲.۲ باشد این برنامه بر روی آن گوشی قابل اجرا نمیباشد. به عبارت دیگر این برنامه بر روی گوشی های با نسخه ۲.۲ و ۲.۲ به بعد قابل اجرا میباشد.

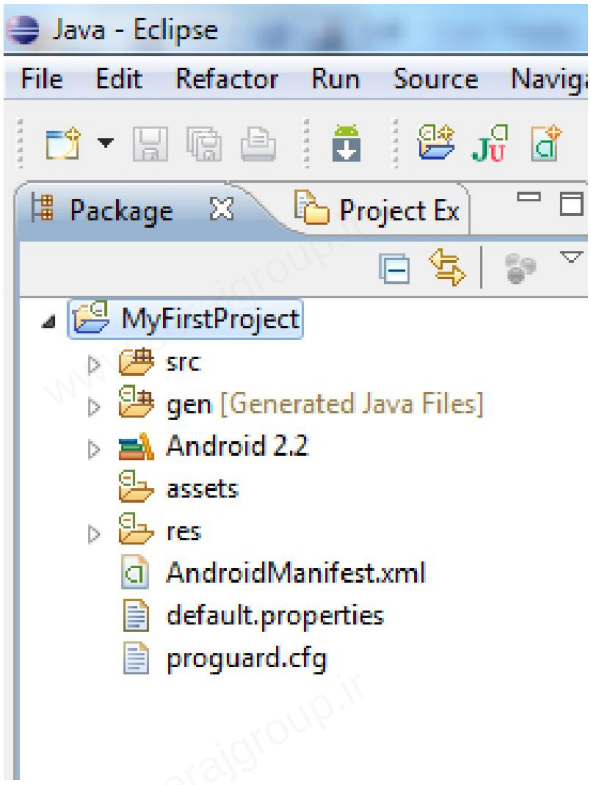

New Android Project
New Android Project
Project name must be specified
Project name:
Name of the Eclipse project to create. It cannot be empty.
Contents
Create new project in workspace
Create project from existing source
Use default location
Location: E:/Projects/Android
Browse...
Create project from existing sample
Samples: Please select a target.
Build Target
Target Name | Vendor | Platform | API Level
Android 1.1 | Android Open Source Project | 1.1 | 2
Android 1.5 | Android Open Source Project | 1.5 | 3
Google APIs | Google Inc. | 1.5 | 3
Android 1.6 | Android Open Source Project | 1.6 | 4
Google APIs | Google Inc. | 1.6 | 4
Android 2.0 | Android Open Source Project | 2.0 | 5
Google APIs | Google Inc. | 2.0 | 5
< Back
Next >
Finish
Cancel

شکل ۲-۲


New Android Project
New Android Project
By convention, package names usually start with a lowercase letter
Android 2.0 Android Open Source Project 2.0 5
Google APIs Google Inc. 2.0 5
Android 2.0.1 Android Open Source Project 2.0.1 6
Google APIs Google Inc. 2.0.1 6
Android 2.1-upd... Android Open Source Project 2.1-upd... 7
Google APIs Google Inc. 2.1-upd... 7
Android 2.2 Android Open Source Project 2.2 8
Google APIs Google Inc. 2.2 8
GALAXY Tab Ad... Samsung Electronics Co., Ltd. 2.2 8
Android 2.3 Android Open Source Project 2.3 9
Google APIs Google Inc. 2.3 9
Standard Android platform 2.2
Properties
Application name: MyFirstProject
Package name: Android.MyFirstProject
Create Activity: MyFirstProject_Activity
Min SDK Version: 8
< Back
Next >
Finish
Cancel

شکل ۲-۳

تذکر: نام بسته[1] بایستی یکتا[2] باشد. یعنی بر روی یک سیستم نباید دو پروژه دارای نام بسته یکسان باشند. و حرف اول نام بسته بهتر است با حرف کوچک[3] شروع شود. نام بسته ها را بایستی با دقت انتخاب کرد که اولاً تکراری نباشد و ثانیاً بعداً به هر دلیلی مجبور نشوید که آن را عوض کنید چونکه در کل پروژه از این نام استفاده شده است و بایستی این نام را در کل پروژه به صورت دستی عوض کنید. سپس از پنل Package Explorer برنامه اکلیپس میتوانید فایل های مربوط به پروژه را مشاهده نمایید. (شکل ۱-۴) این فایلها توسط ویزارد برنامه به صورت خودکار ایجاد شده اند. در جدول (۱-۱) انواع این فایل ها را مشاهده میکنید.



شکل ۲-۴

[1] Package name
[2] Unique
[3] Lowercase letter

نام فایل	هدف و کاربرد فایل
YourActivity.java	فایل به زبان جاوا که اولین activity از این فایل شروع میشود.
R.java	فایلی که به طور خودکار ایجاد میشود و هر چیزی در برنامه را به یک مقدار هگزا دسیمال نسبت میدهد
Android Library	فولدری که حاوی تمام فایل های مربوط به کلاس های SDK اندروید انتخابی است
Assets	شامل تمام فایل های مربوط به پروژه است.عکس ، و چند رسانه ای و...
Res	دایرکتوری منابع و فایل های برنامه که مربوط به ظاهر[1] برنامه است.

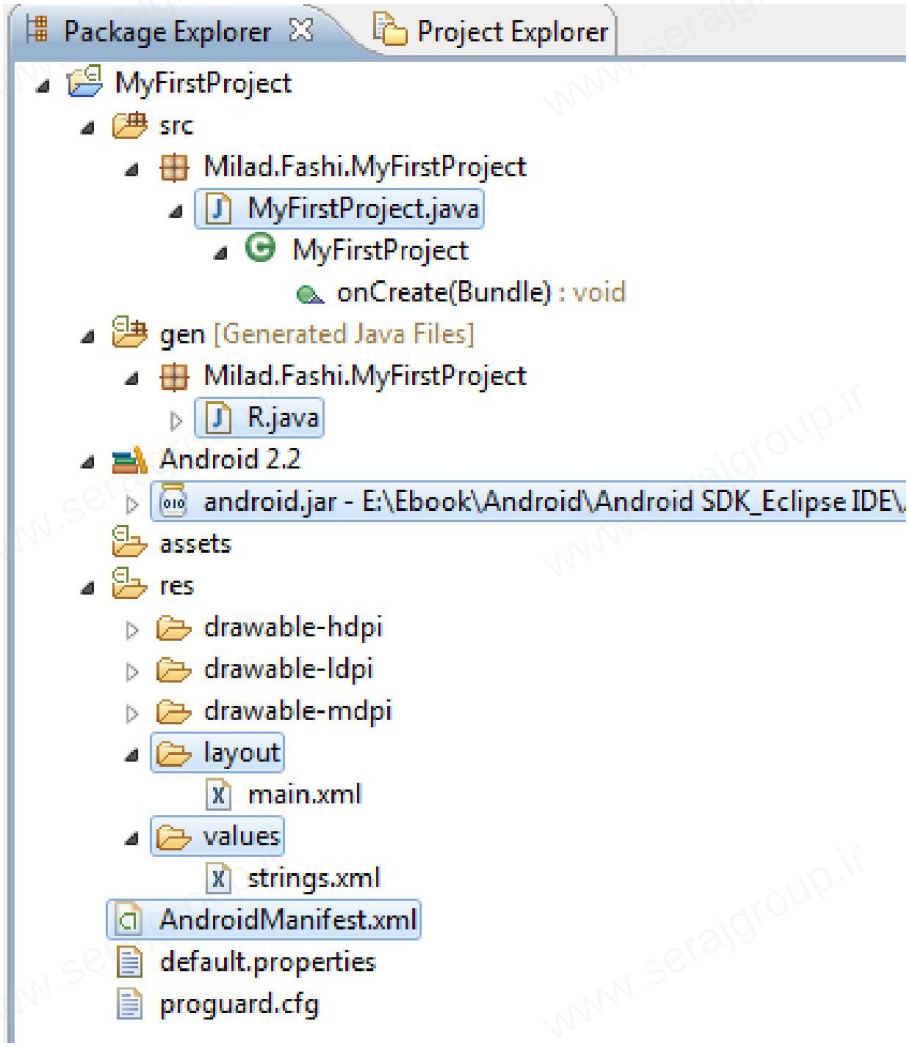
[1] User Interface

res/drawable	دایرکتوری مربوط به عکس های برنامه
res/layout	کدهای XML مربوط به ظاهر برنامه
res/values	محل ذخیره مقادیر String (رشته ها) و پیکربندی ها
AndroidManifest.xml	فایلی که در آن اطلاعاتی در مورد برنامه ذخیره میشود تا سیستم عامل از روی این فایل مشخصات برنامه را ببیند.

جدول ۲-۱

البته در جدول (۱-۱) به صورت خلاصه کاربرد و هدف از ایجاد این فایل ها را توضیح دادیم.در جلوتر شما با مفاهیم آن به صورت کلی تر آشنا میشوید و میتوانید بهتر اهداف این فایلها را درک کنید.در شکل (۱-۵) میتوانید فایل ها و دایرکتوری[1] های مربوط به پروژه که در جدول بالا هم به آن اشاره شد به صورت ساختار درختی مشاهده نمایید.

[1] Directory(Folder)



شکل ۲-۵

در فایل YourActivity.java که در پروژه ما نام آن MyFirstProject.java است.

اطلاعات زیر دیده میشود.(با دو بار کلیک بر روی نام این فایل در پنل مربوط به پروژه میتوانید اطلاعات آن را مشاهده کنید)

```
package Milad.Fashi.MyFirstProject;

import android.app.Activity;
import android.os.Bundle;

public class MyFirstProject extends Activity {
/** Called when the activity is first created.*/
    @Override
public void onCreate(BundlesavedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
    }

}
```

دقیقاً مانند جاوا کلاس های مورد نظر به پروژه با کلمه کلیدی import اضافه شده اند.

متد onCreate() برای اجرای برنامه در اولین اجرا کاربرد دارد. و باعث میشود که اولین اکتیویتی فعال شده و برای شما نشان داده شود.(در اندروید کل برنامه را که بر روی صفحه نمایش دیده میشود را میتوان یک اکتیویتی نامید). جلوتر در مورد انواع اکتیویتی ها و انواع متد های برنامه های اندروید توضیح میدهیم.فقط باید توجه کنید متد onCreate() همانند متد main() در برنامه های جاوا و سی است ولی در اندروید الزاماً برنامه از این متد شروع نمیشود! در ادامه کتاب بحثی در مورد Android Life Cycle وجود دارد که با این موضوع بهتر آشنا میشوید و میبنید که در اندروید برای شروع برنامه فقط یک متد نداریم و بسته به شرایط و موقعیت برنامه یکی از متد های خاص باعث شروع برنامه میشود و مانند جاوا به این شکل نیست که برنامه حتماً باید با متد Main() شروع شود و تا End این متد هم برنامه بایستی اجرا شود.

کلاس MyFirstProject هم از کلاس Activity ارث بری دارد و متد onCreate() از کلاس پدر[۱] بازنویسی[۲] شده است. در واقع متد onCreate() مربوط به کلاس Activity است و در کلاس ما هم بازنویسی شده است و به همین خاطر در انتها مقدار savedInstanceState برای آن فرستاده شده است.

```
super. onCreate(savedInstanceState);
```

در تکه کد بالا مقدار savedInstanceState برای این است که حالت فعلی برنامه حفظ شود. یعنی اگر برنامه به هر دلیلی به پشت صحنه برود و سپس دوباره به جلوی صحنه بیاید مقادیر درون فیلد های برنامه و حالت فعلی آن بایستی حفظ شود.مثلاً در حین اجرای برنامه ممکن است تلفن شما زنگ بخورد و برای مدتی برنامه به پروسه های پس زمینه برود.

در نهایت هم setContentView(R.layout.*main*) این متد باعث میشود که اطلاعات موجود در فایل main.xml در خروجی بر روی صفحه نمایش نشان میدهد. در واقع این متد مربوط به ساخت نمایش واسط کاربری است. و معمولاً در متد onCreate فراخوانی میشود.

---

[۱] دقت نمایید که کلاس پدر (parent) و super class به معنی کلاسی است که دیگر کلاس ها از آن مشتق(Derivation) میشوند. و این ارث بری با کلمه کلیدی Extends مشخص شده است.

[2] Override

فولدر دیگری که در پنل Package Explorer دیده میشود ، فولدر gen است.ما با این فولدر به عنوان برنامه نویس سرو کار نداریم.فایل مهمی که در این فولدر قرار دارد فایل R.Java است که توسط سیستم به صورت خودکار ایجاد میشود و دستکاری کردن این فایل بدون اینکه اطلاعاتی از نحوه تولید آن داشته باشید باعث خراب شدن پروژه میشود.در این فایل هر عنصری(فایل،تصویر،متن و هر منبع دیگری) که در پروژه استفاده شده است به آن یک عدد هگزا دسیمال نسبت داده شده است.این عدد را سیستم به عناصر استفاده شده در برنامه میدهد.واگر شما پروژه را تغییر دهید به صورت خودکار این فایل هم تغییر میکند.

```
/* AUTO-GENERATED FILE.  DO NOT MODIFY.
 * This class was automatically generated by the
 * aapt tool from the resource data it found.  It
 *should not be modified by hand.*/

package Milad.Fashi.MyFirstProject;

public final class R {
    public static final class attr {
    }
    public static final class drawable {
        public static final int icon=0x7f020000;
    }
    public static final class id {
        public static final int btn_Button1=0x7f050000;
    }
    public static final class layout {
        public static final int main=0x7f030000;
    }
    public static final class string {
        public static final int app_name=0x7f040001;
        public static final int hello=0x7f040000;
    }
}
```

در پنل Package Explorer (شکل ۵-۱)کتابخانه های[1] مربوط به Android 2.2 را مشاهده میکنید.این کتابخانه ها کلاس ها و توابع بنیادی را برای برنامه نویسی برای ماشین و سیستم عامل اندروید مهیا میکنند.

تذکر: ما در برنامه خود از gen/R.java و Android 2.2 Library استفاده میکنیم ولی با محتویات آن کاری نداریم و بهتر است که محتویات آن ها را دستکاری نکنید.

پوشه دیگری به نام asset (دارائی) در شکل ۵-۱ وجود دارد که در این پوشه میتوانید تمام تصاویر ، فایل های صوتی و تصویری و به عبارت دیگر هر چیزی که میخواهیم محتویات آن تغییر نکند و همواره ثابت باشد در این پوشه نگهداری میشود.برای مثال اگر شما بخواهید یک بازی برای اندروید طراحی کنید به این پوشه برای نگهداری عناصر چندرسانه ای احتیاج دارید.

پوشه دیگری به نام res در شکل ۵-۱ دیده میشود که دارای پنج زیر پوشه[2] است.

زیر پوشه های پوشه ی res این ها هستند :

۱. res/drawable-hdpi/ : در این پوشه تصاویری با کیفیت بالا و وضوح(Resolution) بالا قرار دهید. همانطور که از نام پوشه هم پیداست

[1] Library
[2] Sub Directory

(HDPI=high dot per inch) . اگر گوشی مقصد دارای LCD با کیفیت بالا باشد عکس های موجود در این پوشه برای او نمایش داده میشود.تصاویر موجود در این پوشه بایستی از نظر سایز و ابعاد(Dimension) بزرگتر و از نظر وضوح هم با کیفیت تر باشند.

۲. res/drawable-ldpi/ : یه نسخه از عکس هایی که در پوشه-drawable hdpi قرار داده اید را هم باید در این پوشه قرار دهید تا اگر گوشی مقصد دارای کیفیت تصویر پایینی باشد این عکس ها برای او نمایش داده شود.

۳. res/drawable-mdpi/ : یه نسخه از عکس هایی که در پوشه های drawable-hdpi و drawable-ldpi قرار داده اید را هم باید در این پوشه قرار دهید تا اگر گوشی مقصد دارای کیفیت متوسط[1] باشد این عکس ها برای او نمایش داده شود.

تذکر : برای تبلت هایی که سیستم عامل اندروید بر روی آنها نصب شده وشما قصد دارید برنامه یتان بر روی تبلت ها هم اجرا شود میتوانید به صورت دستی پوشه ای با نام res/drawable-xdpi ایجاد نمایید و فایل های تصویری بزرگتر و با کیفیت تر که مخصوص تبلت ها است را در این پوشه قرار دهید.

[1] Medium

۴. res/layout/ : در این پوشه اطلاعات مربوط به ظاهر برنامه مانند Text Box ، Button ، Label ، Check Box و Radio Button و ... قرار میگرد. که اصطلاحاً به آن رابط کاربری(UI)[1] میگوییم.اطلاعات این بخش به صورت xml نوشته میشود و توصیه میشودکه از محیط گرافیکی[2] استفاده نکنید و بهتر است به صورت دستی و با نوشتن کدهای xml رابط کاربری را طراحی کنید.فایلی که در فولدر layout قرار دارد و اطلاعات UI به صورت xml در آن ذخیره میشود main.xml نام دارد.

محتویات فایل main.xml در پروژه MyFirstProject :

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    >
<TextView
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/hello"
    />
<Button android:text="Finish" android:id="@+id/btn_finish"
android:layout_width="wrap_content"
android:layout_height="wrap_content"></Button>
</LinearLayout>
```

[1] User interface
[2] Graphic Layout

- ✓ خط اول این کدها هم که مربوط به اسناد xml است و بایستی نوشته شود. تگ <LinearLayout> هم تگی است که حاوی تگ های دیگر برنامه است . در واقع این تگ مانند سطرهای جدول عمل میکند.

- ✓ android:orientation="vertical" این عبارت مشخص میکند که جهت (orientation) کل برنامه به صورت عمودی[1] باشد.

- ✓ android:layout_width="*fill_parent*" : به این معنی است که پهنای برنامه به اندازه پهنای صفحه نمایش باشد.

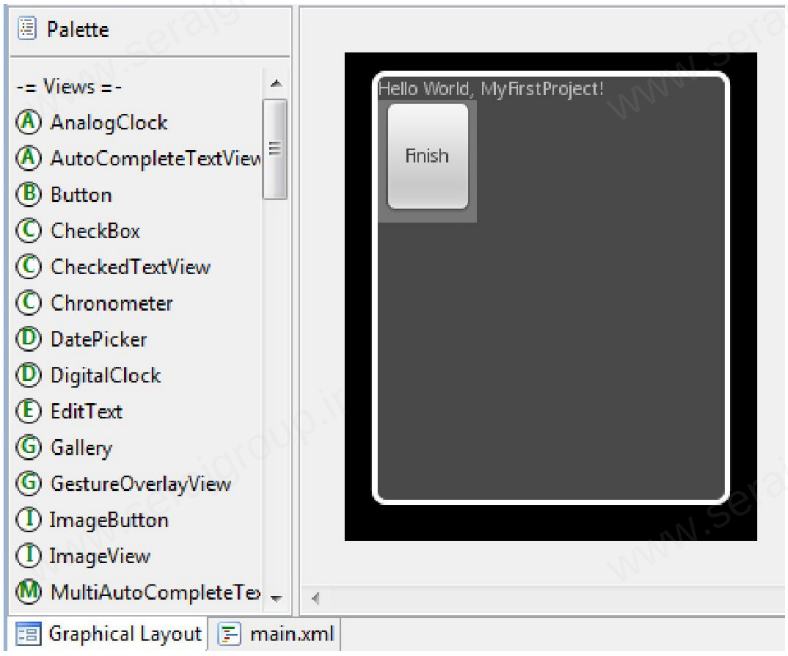
- ✓ android:layout_height="*fill_parent*" : به این معنی است که ارتفاع برنامه به اندازه ارتفاع صفحه نمایش شود و تمام صفحه را پرکند.

- ✓ برای Text viewوButton هم طول و عرض را داریم. ولی تفاوت های جزیی با طول و عرض برنامه دارند.برای مثال اگر عرض Text view برابر fill_parent شود پهنای آن تمام صفحه را پر نمیکند بلکه تمام یک سطر از LinearLayout را پر میکند.اگر هم برای ارتفاعش مقدار wrap_content قرار دهیم به این معنی است که حداکثر ارتفاع را متن داخل Text view مشخص میکند.شکل (۱-۶) رابط کاربری را به صورت گرافیکی[2] برای این کد نمایش

[1] Vertical(Portrait)
[2] Graphical Layout

میدهد.همانطور که میبینید با استفاده از tab هایی که وجود دارد میتوانید بین دو حالت کد و گرافیک جابجا شوید. با کلیک بر روی main.xml میتوانید به حالت کد switch کنید.

در بخش مربوط به طراحی رابط کاربری (User Interface Design) این مطالب با وسواس و جزییات بیشتری بیان میشود.



شکل ۲-۶

۵. Res/values : در این پوشه هم فایلی با نام strings.xml قرار دارد. با دو بار کلیک کردن بر روی این فایل پنجره ای با عنوان Resources باز میشود. در

این پنجره با کلیک بر روی زبانه strings.xml میتوانید محتویات این فایل را مشاهده کنید.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="hello">Hello World,
MyFirstProject!</string>

<string name="app_name">MyFirstProject</string>
</resources>
```

با استفاده از زبان xml و تگ های string مقادیر ثابتی در پروژه ذخیره کرده ایم. و هدف این است که در برنامه بتوانیم با استفاده از نامی که برای این تگ ها مشخص کرده ایم به مقادیر آنها دسترسی پیدا کنیم.(منظور از نام مقدار خصیصه[1]ی Name است).

در قسمت Resources هم میتوانید بدون کد نویسی، مقادیر ثابت را که میتواند از انواع مختلف زیر باشد را ایجاد کنید.

String, Integer Array, String Array, Color, Dimension, Drawable, Item, Style/Theme

با این کار ثوابت را از برنامه جدا کرده اید در نتیجه هم کد پیمانه ای تری (ماجولارتر[2]) نوشته اید و هم برای تغییر مقادیر میتوانید تنها تغییرات را در این فایل انجام دهید و در نتیجه نیاز به تغییر کل پروژه نمیباشد. این جدا سازی و دسته بندی ها از مزایای برنامه نویسی اندروید محسوب میشود .

---

[1] Attribute
[2] Modular

فایل دیگری که در Package Explorer شکل(۱-۵) دیده میشود فایل AndroidManifest.xml است.این فایل مهمی است و در آن اطلاعاتی در مورد برنامه ما نگهداری میشود و درقالب xml به صورت صریح برنامه ما را تشریح میکند.این فایل در واقع رابطی بین برنامه شما و جهان بیرون است.منظور این است که پس از انتشار فایل و قرار گیری آن بر روی مارکت ها (جهان بیرون) این سیستم ها از روی این فایل اطلاعاتی را در مورد برنامه بدست می آورند.در این فایل اطلاعاتی در مورد مجوز های دسترسی و ... هم دخیره میشود که جلوتر با جزییات بیشتری برسی میشود.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com
/apk/res/android" package="Milad.Fashi.MyFirstProject"
android:versionCode="1" android:versionName="1.0">

<uses-sdk android:minSdkVersion="8" />

<application android:icon="@drawable/icon"
android:label="@string/app_name">

<activity android:name=".MyFirstProject"
android:label="@string/app_name">

<intent-filter>
 <action android:name="android.intent.action.MAIN" />

<categoryandroid:name=
"android.intent.category.LAUNCHER" />

</intent-filter>
</activity>
</application>
</manifest>
```

android:versionCode="1" یک عدد صحیح[1] است و با هر تغییر و آپدیتی که در برنامه ایجاد میکنید بایستی یک واحد به عدد نسخه برنامه اضافه کنید تا برای انتشار در یک مارکت[2] (فروشگاه یا بازار آنلاین) اندروید آماده شود.

android:versionName="1.0" میتواند یک مقدار String هم بگیرد و میتوانید نام نسخه را وارد نمایید.مثلاً نسخه Ultimate از برنامه را با این خصیصه میتوانید مشخص نمایید.در محیط مارکت این Version Name به کاربری که قصد دانلود برنامه شما را دارد نمایش داده میشود.

android:minSdkVersion="8" در این مورد هم توضیح دادیم و گفتیم که مارکت با خواندن و بررسی عدد ۸ میتواند تشخیص دهد که برنامه ما با گوشی مقصد سازگاری دارد یا خیر؟

تگ Application دارای دو خصیصه Android:icon و Android:Lable است که به ترتیب آیکون برنامه و نام برنامه را مشخص کرده است. نوع آدرس دهی ها هم نسبی[3] است. و *@drawable* به این معنی است که به پوشه res/drawableرجوع کن و Icon را با توجه به نوع کیفیت تصویر موبایل اجرا کننده برنامه از پوشه مربوطه (LDPI یا MDPI یا HDPI) بخوان. نام برنامه را هم از فایل String.xml میخواند.(در

---

[1] Integer
[2] Market
[3] Relative

مورد Resources ها که در پوشه Value بود قبلاً توضیح داده شد). Intent در اینجا دارای دو خصوصیت Action و Category است و توسط این دو خصیصه حداقل بایستی یک Activity را به عنوان شروع کننده یا به اصطلاح Activity to Fire up مشخص کنیم. Launcher هم مشخص کننده اولین Activity است که باید اجرا (شروع[1]) شود.

در مورد Activity و Intent ها هم در فصل های آینده توضیح داده خواهد شد.

در فایل default. Properties هم اطلاعاتی قرار میگیرد که ما با آن کاری نداریم.

کد نویسی برای کنترل ها

در واژگان برنامه نویسی اندروید بهتر است به جای واژه کنترل از واژه ویجت (Widget) استفاده کنیم. برای اینکه برای یک ویجت از نوع دکمه(Button) کد بنویسید بایستی مراحل زیر را قدم به قدم انجام دهید.(البته اگر کد نویسی برای یک ویجت خاص مثل دکمه را یاد بگیرید میتوانید با کمی تغییر برای دیگر ویجت ها هم کد بنویسید و دلیل آن هم این است که همه ویجت ها از یک کلاس خاص به نام View مشتق میشوند)

[1] Launch

ابتدا بایستی بسته مربوط به کلاس های ویجت دکمه را به کد جاوا وارد کنید.(Import)

import android.widget.Button;

سپس در متد onCreate() برنامه بایستی کد زیر را بنویسید.(کد Bold شده)

```
public void onCreate(Bundle savedInstanceState) {
 super.onCreate(savedInstanceState);
 setContentView(R.layout.main);
 Button btn_finish=(Button) findViewById(R.id.btn_Finish);

}
```

در برنامه نویسی اندروید برای طراحی کلاس ها و متدها و تعریف پردازش ها و کارها و به طور کلی برای برنامه نویسی دستوری[1] از زبان جاوا و کلاس های مخصوص اندروید که به زبان جاوا نوشته شده اند استفاده میشود و برای کارهای خاص میتوان از کتابخانه هایی که به زبان C++ آماده شده اند هم استفاده کرد. و برای طراحی واسط کاربری از کدهای xml استفاده میشود. به کدهای جاوا ، کد منبع و به کدهای xml کد رابط کاربری هم میگویند. شما در کد پایین فقط ارتباط بین کد منبع[2] و کد رابط کاربری[3] را ایجاد کرده اید .

Button btn_finish=(Button) findViewById(R.id.*btn_Finish*);

[1] imperative
[2] Source Code
[3] User Interface

یعنی فقط بین کدهای جاوا و xml ارتباط برقرار کرده اید .در این کد شما یک متغیر از نوع دکمه ساخته اید.و دکمه ای که کد شناسایی[1] آن btn_finish است را با استفاده از تابع **findViewById** پیدا کرده اید و آن را به این متغیر نسبت داده اید. الگوی کلی آدرس دهی اشیایی که در طراحی استفاده کرده اید به شکل R.id.Object_Name است که ، R.Java همان فایلی است که توسط سیستم به صورت خودکار ایجاد میشود و در مورد آن مختصراً توضیحاتی داده شد ، و id هم که همواره باید نوشته شود و Object_Name نام عنصری است که قصد دسترسی به آن را دارید. نام عناصر(Object_Name) هم از طریق خصیصه android:id در کدهای xml تعیین میشود. کد زیر کد xml مربوط به دکمه Finish است :

```
<Button android:text="Finish" android:id="@+id/btn_finish"
android:layout_width="wrap_content"
android:layout_height="wrap_content"></Button>
```

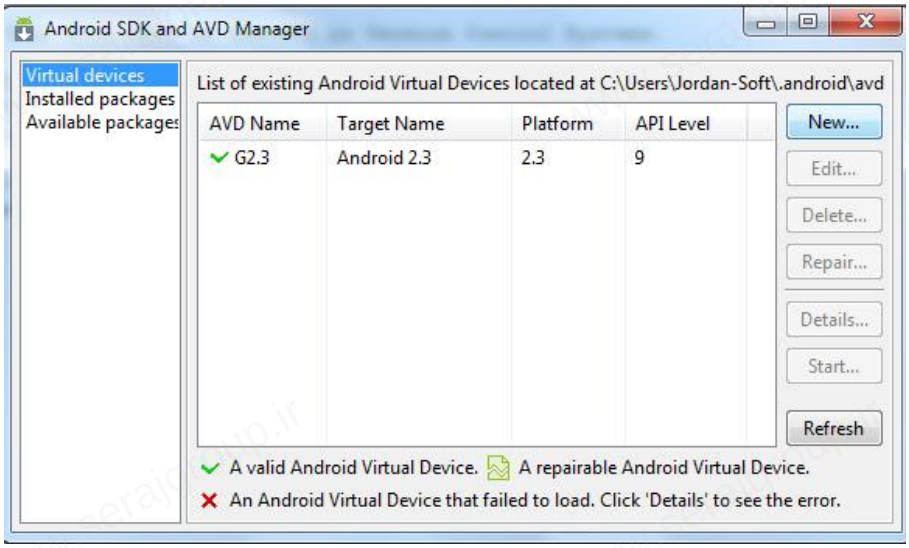
دقت کنید که الگوی[2] کلی نام گذاری عناصر به شکل "*@+id/Object_Name*" است.

نکته ریزی که در کد Button btn_finish=(Button) findViewById(R.id.btn_Finish); وجود دارد استفاده از Type Cast

[1] identification code
[2] Paradigm

اجرای برنامه و تنظیمات ماشین مجازی

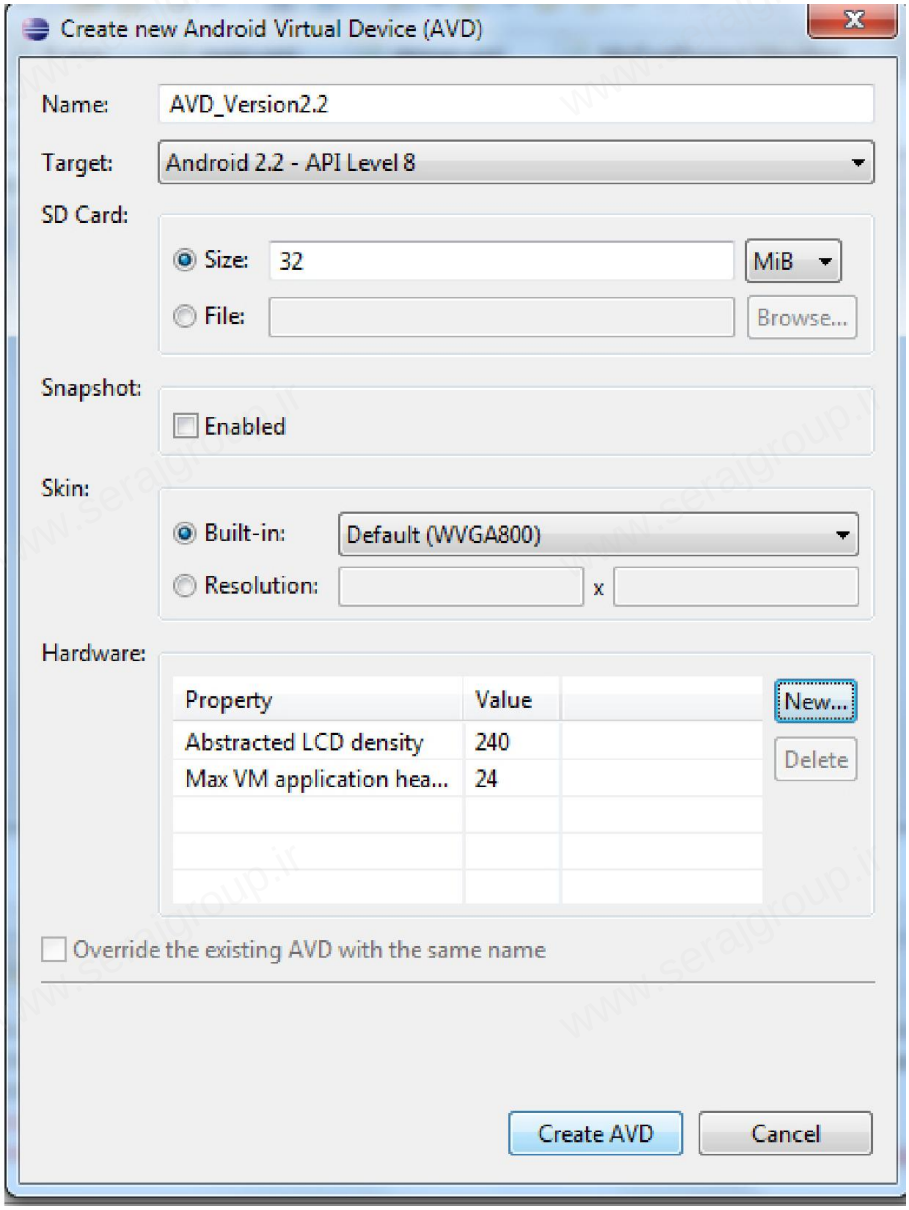
قبل از اجرای برنامه شما باید تنظیمات مربوط به شبیه ساز(ماشین مجازی[1]) را انجام داده باشید.برای این کار میتوانید از منو Window گزینه Android SDK And AVD Manager را انتخاب کنید.شکل۱-۷ برای شما نمایش داده میشود(البته قبلش باید AVD[2] و SDK[3] مربوط به اندروید را بر روی Eclipse نصب کرده باشید).



شکل ۲-۷ تنظیم ماشین مجازی دالویک

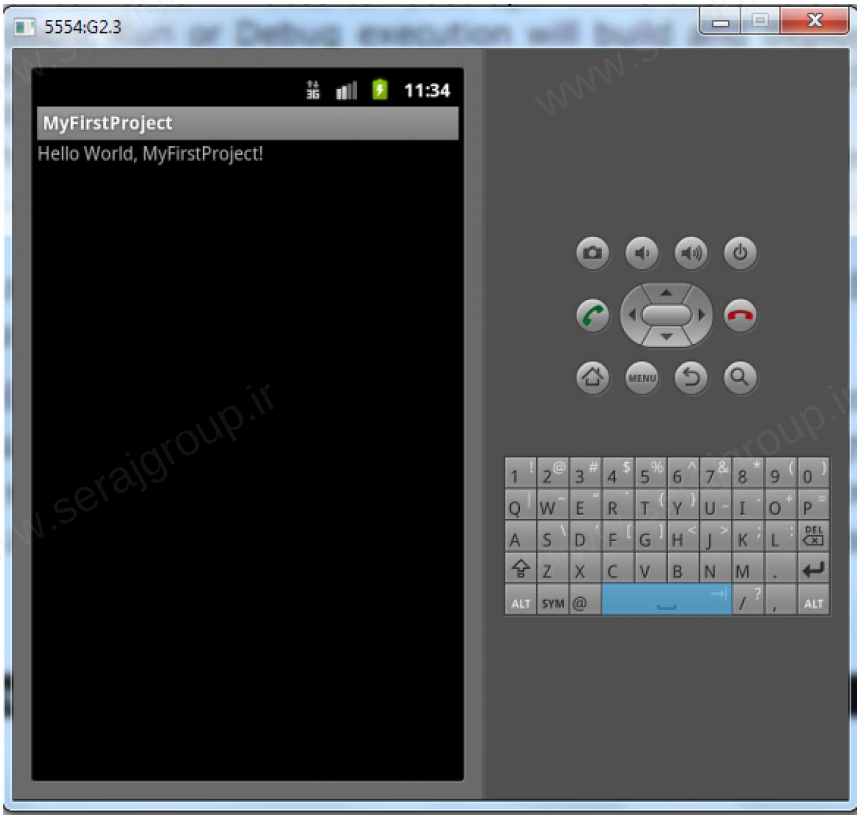
بر روی دکمه New کلیک کنید تا پنجره شکل ۱-۸ برای شما نمایش داده شود .

[1] Virtual Device(Emulator)
[2] Android Virtual Device
[3] Software Development kit


Create new Android Virtual Device (AVD)
Name:
AVD_Version2.2
Target:
Android 2.2 - API Level 8
SD Card:
Size:
32
MiB
File:
Browse...
Snapshot:
Enabled
Skin:
Built-in:
Default (WVGA800)
Resolution:
x
Hardware:
Property
Value
Abstracted LCD density
240
Max VM application hea...
24
New...
Delete
Override the existing AVD with the same name
Create AVD
Cancel

شکل ۲-۸

طبق شکل مقادیر را وارد نمایید.برای عدد Resolution هم میتوانید عدد مربوط به گوشی خود را وارد نمایید تا ماشین مجازی از نظر اندازه و دقت صفحه نمایش به اندازه های واقعی نزدیک تر شود. اندازه SD card هم میتواند هر عددی باشد و چون به صورت مجازی است لزومی ندارد که حتماً مضرب ۲ باشد.پس از مشخص کردن موارد فوق با کلیک بر روی Create AVD میتوانید ماشین مجازی خود را ایجاد کنید. برای اجرای برنامه میتوانید از منوی Run گزینه Run را انتخاب کنید یا کلید ترکیبی Control+F11 را فشار دهید. خروجی برنامه Hello World را در شکل ۱-۹ مشاهده میکنید.



شکل ۲-۹ خروجی برنامه در شبیه ساز

در اولین اجرا مراحل راه اندازی[1] شبیه ساز مقداری طول میکشد.به همین خاطر برای صرفه جویی در وقت بعد از اولین اجرای آن دیگر لازم نیست که آن را ببندید! بستن شبیه ساز دقیقاً مانند خاموش کردن موبایل است.پس دقیقاً مانند یک موبایل که دارای سیستم عامل اندروید است میتوانید از این ماشین مجازی(شبیه ساز) استفاده کنید.برای

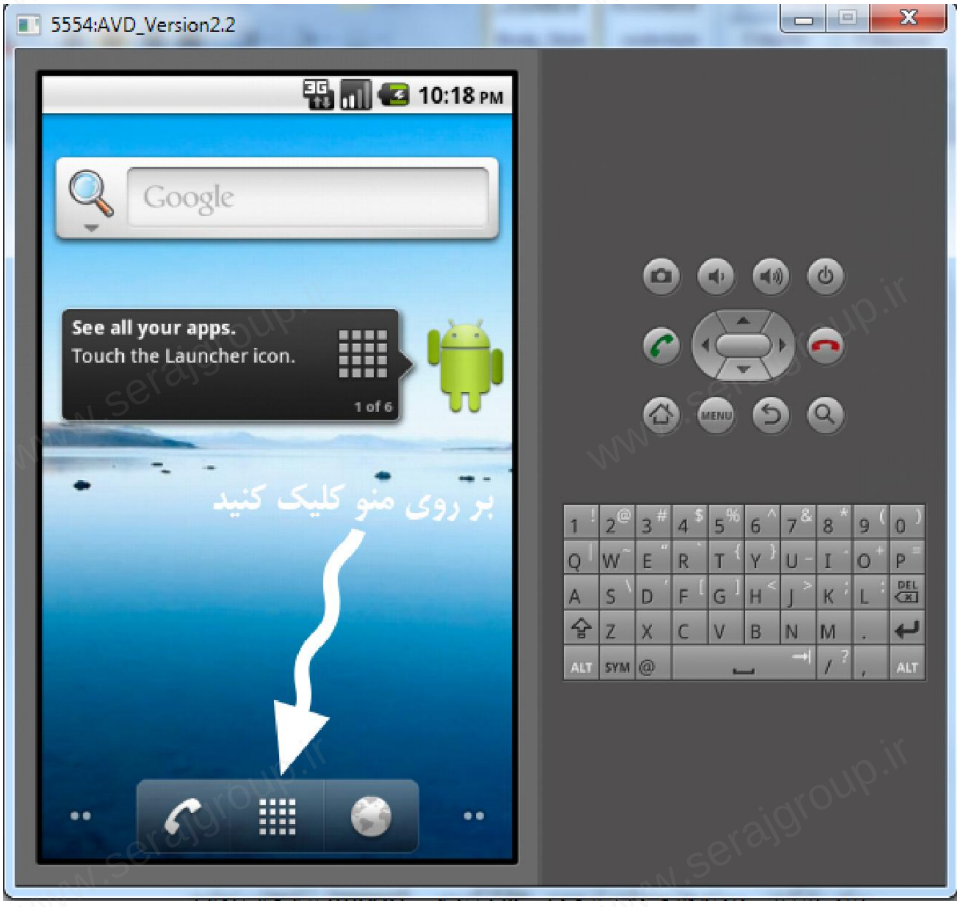
[1] Boot up

اجرای برنامه های خود میتوانید به منوی شبیه ساز رفته (شکل ۲-۱۰) و از لیست برنامه ها ، برنامه مورد نظر خود را انتخاب کنید(شکل ۲-۱۱).

تست برنامه بر روی گوشی

شما میتوانید برنامه را بر روی یک دستگاه موبایل واقعی تست کنید. برای اینکار موبایل خود (که نسخه سیستم عامل آن با نسخه برنامه سازگاری دارد) را بوسیله کابل USB به کامپیوتر خود وصل کنید و بعد به برنامه اکلیپس بروید و دکمه F5 را فشار دهید. برنامه اکلیپس به صورت خودکار دستگاه موبایل شما را شناسایی میکند و شکل ( ) را برای شما نمایش میدهد.(اگر که درایور USB ، مخصوص گوشی خود را نصب کرده باشید و تنظیمات اکلیپس را بدرستی انجام داده باشید این پنجره برای شما ظاهر میشود).سپس بر روی عنوان موبایل (عنوان گوشی که سازنده برای آن قرار داده) دوبار کلیک کنید تا برنامه به لیست برنامه های منوی گوشی شما اضافه شود. با باقی کار هم که شما آشنا هستید!

در این فصل با محیط برنامه نویسی برای اندروید و مثال ساده Hello World آشنا شدید.در فصل های بعد سعی میکنیم که با مثال های جذاب تر و کاربردی تری آشنا شوید ولی قبل از آن بایستی با مفاهیمی از قبیل : activities, intents, intent filters, services, Content Providers آشنا بشوید.

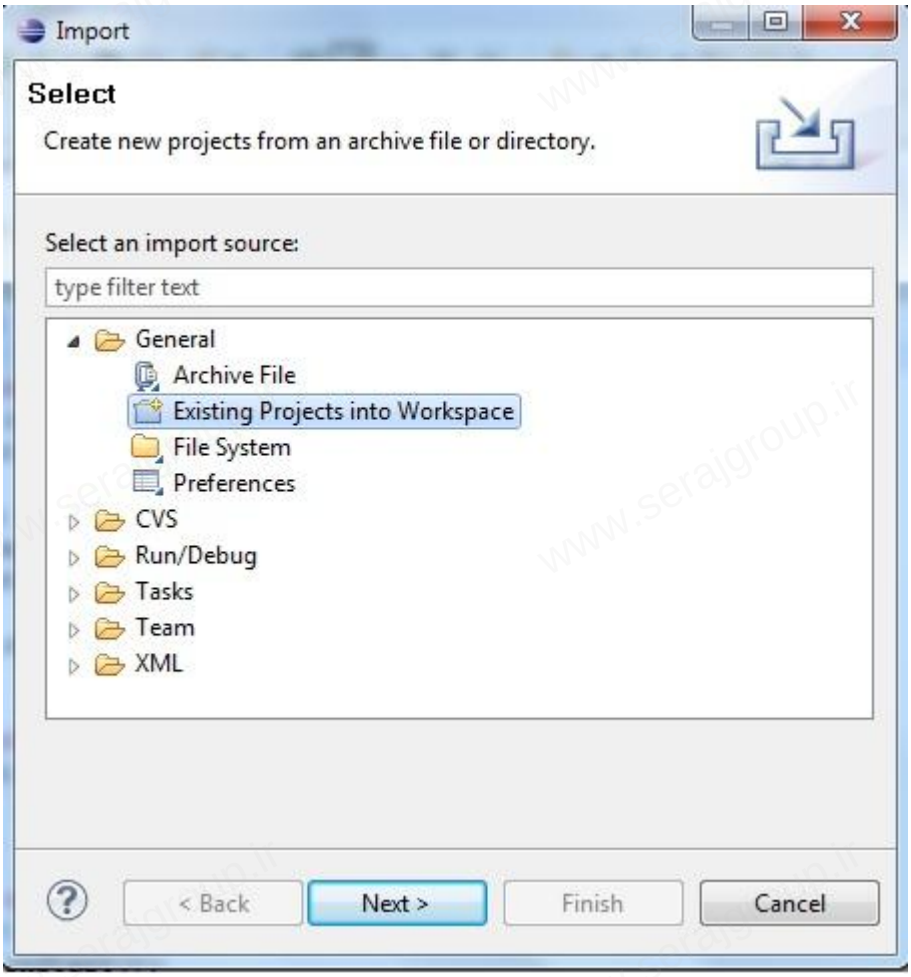

5554:AVD_Version2.2
10:18 PM
Google
See all your apps.
Touch the Launcher icon.
1 of 6
بر روی منو کلیک کنید
MENU


10:26 PM
Alarm Clock
Browser
Calculator
Camera
Car Home
Contacts
Custom Locale
Dev Tools
Email
Gallery
Messaging
Music
Phone
Settings
Spare Parts
Speech Recorder
برنامه شما
MyFirstProject

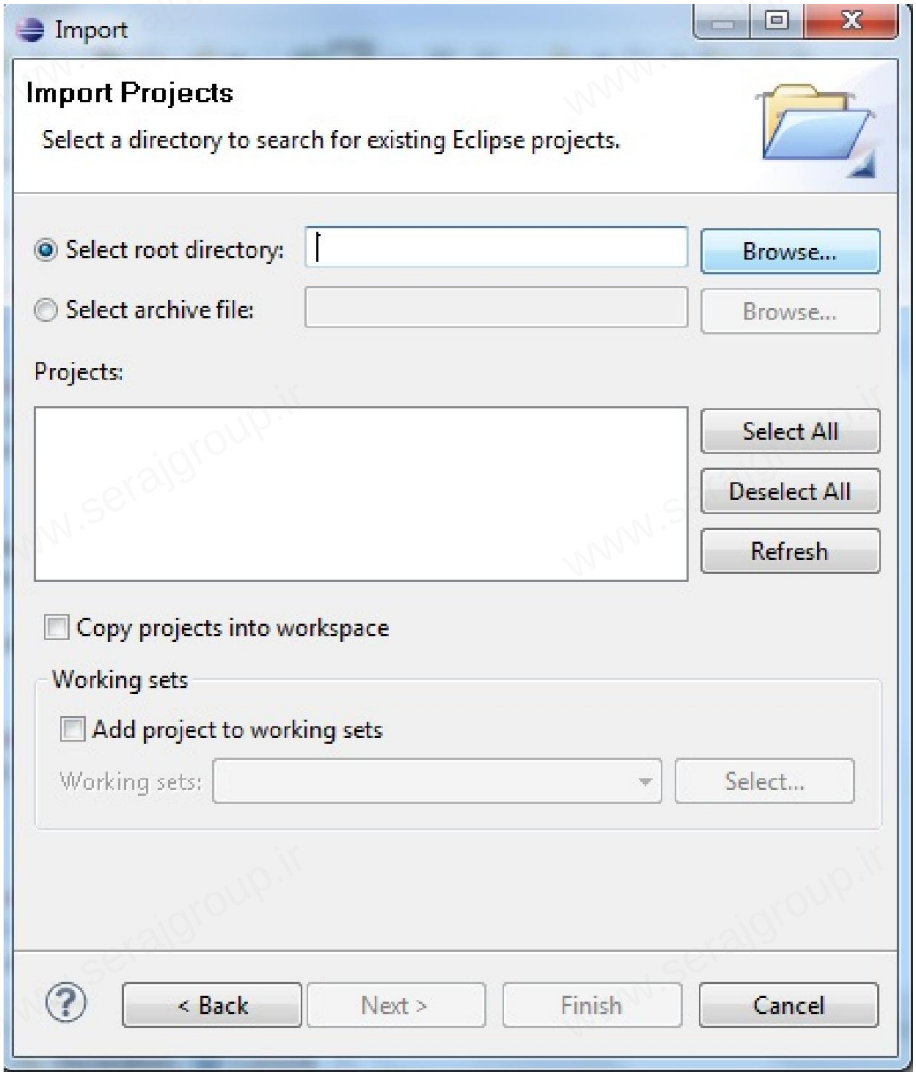
بازکردن پروژه

برای بازکردن پروژه ها میتوانید از منوی File گزینه ...Import را انتخاب نمایید.

پنجره ای برای شما باز میشود که طبق شکل ۲-۱۲ گزینه Existing Projects into Workspace را انتخاب نمایید.



سپس آدرس پروژه را طبق شکل ۱- ۱۱ با زدن دکمه Browse وارد نمایید. تا پروژه مورد نظر به لیست پروژه های موجود در Package Explorer اضافه شود .


Import
Import Projects
Select a directory to search for existing Eclipse projects.
Select root directory:
Browse...
Select archive file:
Browse...
Projects:
Select All
Deselect All
Refresh
Copy projects into workspace
Working sets
Add project to working sets
Working sets:
Select...
< Back
Next >
Finish
Cancel

یادداشت :

فصل دوم : مبانی برنامه نویسی اندروید

مطالبی که در این فصل با آن آشنا میشوید:

در این فصل با مفاهیم بنیادی و کلیدی برنامه نویسی برای اندروید آشنا میشوید.

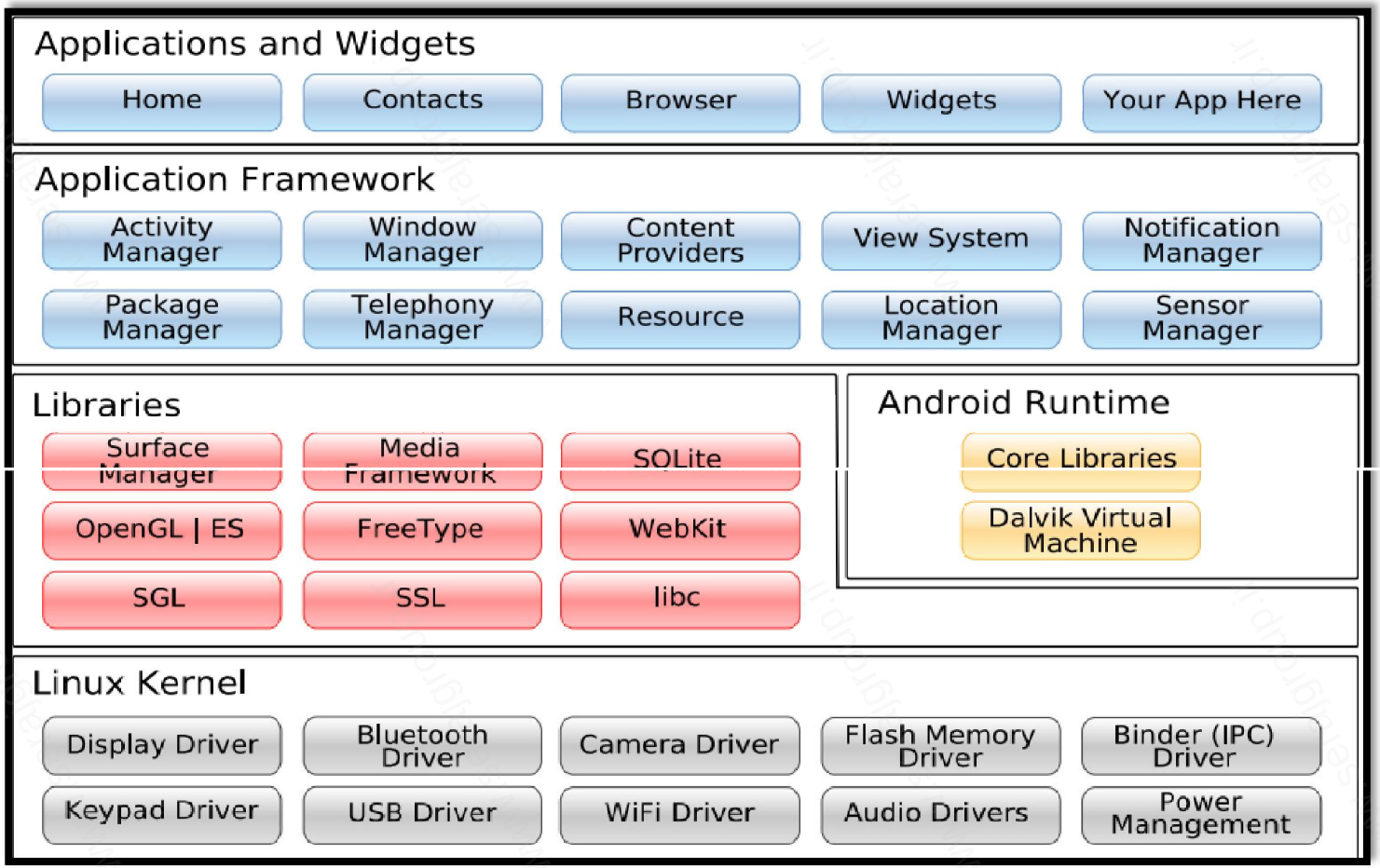
پیشنهاد من به شما این است که حتماً این فصل را مطالعه کنید تا با شناخت اصول اصلی اندروید ، برنامه هایی با کارایی بالا بسازید.

تصویری کلی از معماری اندروید

در شکل ( ) یک تصویر کلی از لایه ها و مؤلفه های معماری سیستم عامل اندروید میبیند.در معماری اندروید هر لایه از سرویس های لایه ی زیرین استفاده میکند.ما از پایین ترین لایه شروع میکنیم.

لایه های اصلی معماری اندروید به صورت زیر است که از پایین ترین لایه یعنی هسته لینوکس شروع میکنیم و هر کدام را جلوتر شرح میدهیم:

- Application and Widgets
- Application Framework
- Android Runtime
- Libraries
- Linux Kernel


Applications and Widgets
Home
Contacts
Browser
Widgets
Your App Here
Application Framework
Activity Manager
Window Manager
Content Providers
View System
Notification Manager
Package Manager
Telephony Manager
Resource
Location Manager
Sensor Manager
Libraries
Surface Manager
Media Framework
SQLite
OpenGL | ES
FreeType
WebKit
SGL
SSL
libc
Android Runtime
Core Libraries
Dalvik Virtual Machine
Linux Kernel
Display Driver
Bluetooth Driver
Camera Driver
Flash Memory Driver
Binder (IPC) Driver
Keypad Driver
USB Driver
WiFi Driver
Audio Drivers
Power Management

لایه هسته لینوکس(Linux Kernel)

اندروید بر روی یک فونداسیون و پایه قوی(Proven Foundation) به نام لینوکس بنا شده است.امروزه لینوکس در هر وسیله ای دیده میشود.(از یک ساعت مچی گرفته تا ابر رایانه[1] ها).

لایه لینوکس در واقع یک انتزاع از سخت افزار[2] را برای اندروید مهیا میکند، و به اندروید اجازه میدهد که از گستره مختلفی از پلتفرم های آینده را پشتیبانی کند. اندروید برای مدیریت حافظه[3] ، مدیریت پردازش ها[4] و سرویس های شبکه[5] و دیگر سرویس هایی که یک سیستم عامل ارائه میدهد از لایه ی لینوکس کمک میگیرد. کاربران گوشی های اندروید هرگز لینوکس را نمیبینند و در واقع از وجود لینوکس مطلع نمیشوند.حتی توسعه دهندگان نرم افزارها هم با این لایه درگیر نیستند ولی بایستی به عنوان یک برنامه ساز از وجود لایه لینوکس آگاه باشید. و به همین خاطر این لایه پایین ترین لایه است و نزدیک ترین لایه به سخت افزار و همانطور که شما تجربه کار با کامپیوتر را دارید ، همواره لایه ای بودن معماری ها برای راحت تر شدن کار کاربران نهایی و برنامه نویسان کاربردی است تا آن ها درگیر جزئیات سخت افزاری و پیچیدگی

---

[1] Super Computer
[2] Hardware Abstraction
[3] Memory Management
[4] Process Management
[5] Networking

های سیستم عامل نشوند. اما اگر میخواهید یک برنامه نویس سیستمی برای اندروید شوید بایستی کار با این لایه را یاد بگیرید.

اما چرا از لینوکس برای هسته گوشی استفاده شده است ؟ در موبایل ها همواره قابلیت اطمینان[۱] مهمتر از کارایی و بازدهی[۲] است. اگر چه امروزه یکی از استفاده های رایج از گوشی های موبایل برای برنامه هایی کاربردی از جمله ماشین حساب ، تقویم ، دیکشنری ، مالتی میدیا و ... است ولی هنوز مهمترین استفاده از موبایل ارتباط تلفنی[۳] است و نباید فراموش شود که ارتباط تلفنی هنوز مهمترین و اصلی ترین نیاز استفاده از موبایل است. و ویژگی هایی که لینوکس دارد و خاصیت قابل اطمینان بودن آن باعث شده که یک ارتباط تلفنی بدون خطا را تضمین کند.

لایه کتابخانه های اختصاصی

لایه ی بعدی ، که بر روی لایه ی هسته لینوکس قرار دارد ، لایه کتابخانه های اختصاصی[۴] است.این کتابخانه ها با استفاده از کدهای C یا C++ نوشته شده اند و برای یک سخت افزار با معماری خاص کامپایل شده اند . این کتابخانه ها مجموعه ای از کلاسهای C و C++ است که توسط کامپوننت های سیستم اندروید مورد استفاده قرار

---

[1] Reliability
[2] Performance
[3] Voice communication
[4] Native Libraries

میگیرند. همچنین استفاده از این قابلیت به توسعه دهندگان نیز داده شده است. در جدول ( )بعضی از این کتابخانه های اصلی را مشاهده میکنید.

نام کتابخانه	کاربرد
System C library	کتابخانه استاندارد سی که پایه آن همان کتابخانه های استاندارد سی از زمان یونیکس است(BSD[1]).
Media Libraries	کتابخانه های که برای ضبط و پخش، تصویر و صدا استفاده میشوند.و قادرند با انواع فرمت رایج مانندMP3,MP4,JPG,PNG کارکنند.
SGL	شامل موتورهای گرافیکی دو بعدی
WebKit Library_SSL[2]	شامل یک موتور برای مرورگرهای پیشرفته که باعث قدرت و امنیت بیشتر مرورگر اندروید شده است. که به آن Browser Engine هم میگویند.
3D libraries(OpenGL)	برای کار با Open GL طراحی شده است.
Free Type	برای ترجمه[3] کردن فونت های Bitmap و Vector (برداری)
SQLite	شامل یک موتور قوی[1] و سبک وزن[2] برای

[1] Barkley Software Distribution
[2] Secure Socket Layer
[3] Render

کار با پایگاه داده های رابطه ای[۳]	

لایه اندروید در زمان اجرا

یکی دیگر از لایه های معماری اندروید Android Runtime میباشد.که شامل ماشین مجازی دالویک[۴] و کتابخانه های هسته جاوا[۵] است.

دالویک چیست ؟ در سال های گذشته برای اینکه قابلیت های حمل برنامه ها[۶] بالا رود و برنامه ها مستقل از ماشین باشند، ماشین های مجازی مثل Net Common. language Runtime توسط مایکروسافت و Java Virtual Machine توسط شرکت سان طراحی شدند.گوگل هم برای رسیدن به این اهداف ماشین مجازی دالویک را طراحی کرد. دالویک یک ماشین مجازی است که توسط Dan Bornstein در شرکت گوگل طراحی و پیاده سازی شد . در طی فرایند ترجمه ، کد های برنامه شما به

---

[1] Powerful
[2] Light Weight
[3] RDB=Relational Data Base
[4] Dalvik Virtual Machine
[5] Core Java Libraries
[6] Portability

دستورات مستقل از ماشین[1] به نام بایت کد[2] تبدیل میشود که بر روی ماشین مجازی دالویک که روی موبایل ها قرار داده شده است اجرا میشود. در واقع دالویک یک ماشین مجازی جاوایی است که برای سیستم عامل اندروید و دستگاه های موبایل بهینه شده است تا برای مثال RAM و CPU و باطری کمتری مصرف کند و دلیل دیگری که از Dalvik VM به جای Java VM استفاده شد این بود که گوگل نمیخواست در قید و بند مجوزهای[3] شرکت Sun باشد و قصد داشتند که این بخش هم آزاد و open sourceباشد .

نرم افزارهای جانبی اندروید با استفاده از زبان جاوا نوشته میشوند و برای ارتباط با لایه های زیرین سیستم عامل میتوانند از کتابخانه های جاوایی اندروید استفاده کنند .بخش رابط کاربری سیستم عامل اندروید با زبان جاوا نوشته شده است و بسیاری از برنامه های اندروید هم با جاوا نوشته شده اند. اما این سیستم عامل ، Java Virtual machine ندارد . برای اجرای برنامه های جاوایی روی این سیستم عامل، کدهای جاوا به کدهای Dalvikتبدیل میشوند و سپس روی Dalvik virtual machine اجرا میشوند.. برنامه های جاوای معمولی(جاواهای قدیمی که برای گوشی های قدیمی ساخته شده اند) روی گوشی های اندروید دیگر اجرا نمیشوند و برای اجرای آنها میتوان با نصب نرم افزارهای شبیه ساز ماشین مجازی جاوا مانند j2ME MIDP Runner روی این سیستم عامل آن

[1] Machine Independent Instruction
[2] Byte Code
[3] License

فایل ها را هم قابل اجرا کرد.یعنی ماشین مجازی جاوا که مخصوص آنهاست را نصب کرد تا بتواند بایت کدها را حین اجرا به صورت خط به خط(مفسری[۱]) به کدهای قابل اجرا[۲] (ماشین کد[۳]) تبدیل کند.

لایه چارچوب برنامه (Application Framework)

با فراهم آوردن پلتفرم توسعه باز(open development platform) ، اندروید برنامه سازان را قادر کرده است تا به سرعت و به آسانی برنامه های کاربردیِ خلاقانه و قوی برای این پلتفرم فراهم سازند. توسعه دهندگان، آزادی کامل دارند تا از ویژگیهایی مانند دسترسی به سخت افزار، دسترسی به اطلاعات محلی (موقیت جغرافیایی)، اجرای سرویس های پس زمینه[۴] ، تنظیم زنگ ساعت، اضافه کردن اطلاعیه ها[۵] به نوار وضعیت و بسیاری بسیاری دیگر در برنامه هایی که میسازند، استفاده کنند.

توسعه دهندگان[۶] دسترسی کامل به همان چارچوبAPI هایی دارند که برنامه های هسته[۷] و اصلی دارند.(API= Application Program Interface در واقع یک سری توابع و کلاس هایی هستند که توسط سازنده سیستم عامل منتشر میشوند و ارتباط

---

[1] Interpreter
[2] Executable code
[3] Machine code
[4] background services
[5] Notifications
[6] Developers
[7] the core applications

دهنده ی برنامه های کاربردی با سیستم عامل و سرویس های آن است).معماری برنامه های کاربردی[1]بمنظور ساده سازیِ[2] استفاده مجدد[3] از کامپوننت ها طراحی شده است. هر برنامه ای می تواند قابلیت های خود را در اختیار دیگر برنامه ها قرار دهد و همچنین از قابلیت های دیگر برنامه ها استفاده کند (البته به محدودیت های امنیتی چارچوب هم بستگی دارد). این طرزکار مشابه به کاربر اختیار تعویض و جایگزینی[4] کامپوننت ها را می دهد. برای مثال در جاوا JDK شامل API هایی است که به برنامه نویس برای نوشتن برنامه های کاربردی کمک میکند. خود API هم از بسته هایی (Package) تشکیل شده است که این بسته ها حاوی کلاس هایی هستند که به برنامه نویس کمک میکنند. برای مثال شما برای کار با لیست ها و مجموعه ها ی با طول متغیر میتوانید از کلاسر ArrayList زیر استفاده کنید.این کلاس به شما کمک میکند که بدون اینکه وارد پیچیدگی های ساختمان داده ها بشوید بتوانید مجموعه ای از اشیا را ذخیره کنید. این سلسه مراتب را در زیر مشاهده میکنید.

JDK>>API>>Package>>Class

Example : import java.util.ArrayList

---

[1] The application architecture
[2] Facilitate
[3] Reuse
[4] Replacement

در جاوا در واقع Package ها کتابخانه ای از کلاس ها هستند.(Class library)

در این لایه ، سرویس ها و سیستم های زیر مشاهده میشوند که مهمترین بخش های این چارچوب به شرح زیر میباشد :

- (View System) : مجموعه قابل گسترشی از View ها که برای ساخت رابط کاربری برای برنامه های کاربردی استفاده میشوند، مانند : lists, grids, text boxes, buttons, embeddable web browser.
- مهیا کننده محتوا (Content Providers) : که برنامه ها را قادر میسازد تا به اطلاعات برنامه های دیگر مانند لیست تماس، دسترسی پیدا کنند یا حتی اجازه دسترسی به اطلاعات خود را به برنامه های دیگر دهند.(اشتراک و کپسوله سازی داده ها را انجام میدهد).
- مدیر منابع (Resource Manager) : اجازه دسترسی به منابع برنامه را فراهم میکند مانند دسترسی به رشته های محلی[1] ، تصاویر و فایل های مربوط به طرح برنامه(layout files). منابع برنامه هر چیزی است که در برنامه شما استفاده میشود و در واقع هر چیزی به غیر از کد های برنامه را میتوان جز منابع برنامه به شمار آورد.

[1] localized strings

- مدیر اطلاعیه(Notification Manager) : که از این طریق برنامه ها را قادر میکند تا هشدارهای خود را در نوار وضعیت نشان دهند.

- مدیر فعالیت (Activity Manager) : که مدیریت چرخه حیات[1] برنامه ها را در دست دارد و به نحوه اجرای برنامه ها نظارت میکند.

- مدیر موقعیت (Location Manager) : از این طریق میتوان سخت افزارهایی مثل GPS و شتاب سنج[2] را کنترل کرد.

لایه برنامه ها کاربردی و ویجت ها (Application and Widgets)

بالاترین لایه در دیاگرام معماری اندروید این لایه است. کاربران نهایی[3] فقط با این لایه در ارتباط هستند و از نرم افزارهای این لایه استفاده میکنند. نرم افزارهای این لایه شامل نرم افزارهای اختصاصی گوشی و بازی ها و نرم افزارهایی است که توسعه دهندگان ایجاد کرده اند و یا حتی نرم افزارهایی که شما بعد از خواندن این کتاب در آینده میسازید. اندروید به همراه بسته های مختلفی از جمله Email Phone dialer,

[1] Life Cycle
[2] Accelerometer
[3] End Users

SMS ,Calendar, Maps,Web Browser, Contacts ,Andrio Market ارایه میشود. تمام این برنامه ها با استفاده از زبان برنامه نویسی جاوا نوشته شده اند.این برنامه ها به کمک لایه های زیرین خصوصاً لایه Android Runtime اجرا میشوند.

برنامه های کاربردی و ویجت ها ابزارهای ارتباط و تعامل با کاربران نهایی هستند.برنامه کاربردی(Application) با ویجت (Widget) تفاوت دارد.برنامه های کاربردی تمام صفحه نمایش را در اختیار میگیرند و با کاربر تعامل دارند.ولی ویجت ها (که بعضاً به آن Gadgets هم میگویند) بر روی صفحه اصلی(Home Screen) قرار دارند و به اندازه ی یک مستطیل کوچک صفحه نمایش را در اختیار دارند. این کتاب بیشتر به ساخت برنامه های کاربردی میپردازد زیرا کاربردی تر و رایج تر هستند.ویجت ها مانند ساعت آنالوگ ، روزشمار و وضعیت آب و هوا و ... هستند.

مدیریت برنامه ها در اندروید

در اینجا قصد دارم که در مورد چگونگی وجود و عملکرد یک برنامه ، در سیستم عامل اندروید نکاتی را در قالب جملاتی کوتاه ولی پرمحتوا بیان کنم تا کمی بهتر با شیوه کاری آن آشنا بشوید.

- همانطور که دیدید سیستم عامل اندروید سیستمی مبتنی بر لینوکس و با قابلیت پشتیبانی از چندکاربر است، بدین صورت که هر برنامه به معنی یک کاربر در نظر گرفته میشود.

- بصورت پیش فرض، سیستم به هر برنامه ای یک کد احراز هویت[1] مخصوص به خودش را میدهد (این کد فقط برای سیستم قابل شناسایی است و برای برنامه شناخته شده نیست). سیستم برای تمام فایلهای برنامه مجوز صادر میکند، این کار باعث میشود تا تنها برنامه با آن کد هویتی خودش به فایلها دسترسی داشته باشد.و از این طریق باعث میشود که هر پردازش حریم خصوصی داشته باشد. و برنامه ها نتوانند به همدیگر دسترسی غیر مجاز داشته باشند .

- هر پردازشی بر روی ماشین مجازی خودش (virtual machine) اجرا میشود. بنابراین، اجرای کدهای یک برنامه از برنامه دیگر در شرایط مجزا (isolation) انجام میشود.

- بصورت پیش فرض، اجرای هر برنامهای بر روی پردازش لینوکسی[2] مربوط به خودش انجام میشود. اندروید، پردازش را زمانی اجرا میکند که کامپوننت نیاز به اجرا شدن داشته باشد (در ادامه به توضیح کامپوننت ها خواهیم پرداخت، نگران

[1] Linux User ID
[2] Linux Process

نباشید!)، پردازش را زمانی متوقف میکند (shuts down) که دیگر به آن نیازی نباشد یا زمانیکه سایر برنامه ها برای اجرا به حافظه بیشتر نیاز داشته باشند.

- بدین روش، اندروید اصل حداقل امتیاز (Principle of least Privilege)را اجرا میکند. بر اساس این اصل، هر برنامه ای، بصورت پیش فرض، تنها به کامپوننت هایی دسترسی خواهد داشت که برای اجرا به آنها نیاز داشته باشد و نه بیشتر. این روش محیطی بسیار امن خواهد ساخت، بدین ترتیب که یک برنامه تا زمانیکه مجوزهای لازم را نداشته باشد، نمیتواند به منابع سیستم دسترسی داشته باشد.

اشتراک داده ها

راه هایی وجود دارد که یک برنامه بتواند دیتاهای خود را با دیگر برنامه ها به اشتراک بگذارد و یا به سرویس هایی که سیستم میدهد، دسترسی داشته باشد:

- این قابلیت برای دو برنامه وجود دارد که بتوانند از یک کد هویتی مشترک استفاده کنند. بدین ترتیب هرکدام میتوانند از فایلهای دیگری استفاده نمایند. برای حفظ منابع سیستم، برنامه هایی که از یک کد هویتی استفاده میکنند میتوانند از یک ماشین مجازی استفاده کنند و بترتیب بر روی پردازشهای لینوکسی اجرا شوند (برنامه ها نیاز به sign شدن با یک گواهینامه [1] دارند.)

[1] Certificate

- هر برنامه میتواند درخواست مجوز برای دسترسی به دیتای دستگاه [۱] مانند مخاطبین های کاربر(user's contacts) ، پیامهایSMS ، حافظه جانبی(SD card) ، دوربین، بلوتوث و غیره را داشته باشد. تمامی این مجوزها در زمان نصب برنامه از کاربر مورد سئوال قرار خواهد گرفت.و کاربر در جریان این مجوز ها قرار میگیرد.این مجوز ها در فایل Manifest.xml

مؤلفه های برنامه(Application Components)

کامپوننت های[۲] برنامه، ضروری ترین بلوک های ساختمانی (building blocks) در ساخت برنامه های کاربردی می باشند. هر کامپوننت دارای خصوصیاتی است که از آن طریق میتوانید درخواست های خود را به سیستم اعلام کنید. ضرورتاً همه کامپوننت ها دریافت کننده دستورات کاربر نیستند و بعضی مواقع وابسته به یکدیگرند، اما هرکدامشان نقش مستقل خود را ایفا میکنند. از آنجا که کامپوننت ها مستقل از یکدیگر هستند، با استفاده از آنها قادریم رفتار کلی برنامه را مشخص کنیم.

---

[1] Device

[۲] به علت رایج شدن برخی کلمات در زبان فارسی در متن کتاب به جای واژه فارسی از معادل انگلیسی استفاده کردم.

چهار نوع مختلف از کامپوننت ها وجود دارد. هر نوع در خدمت یک هدف مشخص است و چرخه حیات (Lifecycle) مخصوص به خودش را دارد(در همین فصل با مفهوم چرخه حیات آشنا میشوید) که معلوم میکند کامپوننت چگونه ایجاد میشود و چگونه از بین میرود.

در اینجا به معرفی چهار نوع کامپوننت ها می پردازیم.

۱. **فعالیت ها (Activities) :** اکتیویتی بیانگر یک اسکرین کامل (صفحه نمایشگر) به همراه یک واسط کاربری است. بعنوان مثال، یک برنامه چک کردن ایمیل را درنظر بگیرید. در این برنامه ممکن است یک اکتیویتی برای نشان دادن لیست ایمیل های جدید داشته باشید، یک اکتیویتی دیگر برای نوشتن و ارسال ایمیل داشته باشید و یک اکتیویتی دیگر برای خواندن ایمیل داشته باشید که هرکدام مستقل از دیگری کار میکنند. بدین ترتیب، یک برنامه مجزا میتواند با هرکدام از این اکتیویتی ها ارتباط برقرار کند (اگر برنامه ایمیل اجازه اینکار را بدهد). بعنوان مثال، یک برنامه عکس برداری میتواند اکتیویتی ارسال ایمیل را بمنظور ارسال عکس گرفته شده، اجرا نماید.پس یک خاصیت جالب برنامه های اندرویدی این است که برنامه ها میتوانند برای انجام کارهای خود با یکدیگر بده بستان[1] داشته باشند. هر اکتیویتی یک زیرکلاس از کلاس

[1] Trade off

Activityاست که در مورد این کلاس مفصل در فصل مربوط به خودش توضیح داده خواهد شد.

۲. **سرویس ها (Services) :** پردازش[1] (فرایند) ای است که در پشت صحنه اجرا میشود و از دید کاربر پنهان است.در واقع این پردازش دارای رابط کاربری[2] نمیباشد.به عبارت دیگر سرویس کامپوننتی است که در پس زمینه[3] اجرا میشود تا یک کاری را برای مدت طولانی انجام دهد یا کاری را برای یک فرایند از راه دور (remote processes) انجام دهد. سرویس مانند اکتیویتی واسط گرافیکی ندارد. بعنوان مثال، یک سرویس میتواند موزیکی را در پس زمینه اجرا نماید درحالیکه کاربر مشغول نوشتن پیامک یا برنامه دیگری است. یا مثلاً سرویس مشغول دریافت دیتا در شبکه است درحالیکه کاربر مشغول انجام کار دیگریست. کامپوننت دیگری، مانند اکتیویتی، میتواند یک سرویس را اجرا کند تا با آن در تعامل باشد.Live Wallpaper ها هم تصاویر متحرکی هستند که در صفحه اصلی قرار میگیرند.و از یک سرویس برای ترسیم تصاویر متحرک[4] استفاده میکنند. هر سرویس یک زیرکلاس از کلاس Service است

---

[1] Process
[2] User interface
[3] background
[4] Animated Picture

که در مورد این کلاس مفصل در فصل مربوط به خودش توضیح داده خواهد شد.

۳. **مهیا کننده محتوا (Content Providers):** یک چارچوب[1] که شامل کلاس هایی اصلی برای ذخیره داده ها است. مهیا کننده محتوا به مدیریت دیتای به اشتراک گذاشته شده در برنامه هم می پردازد.دیتا میتواند به روش های مختلف ذخیره شود. مثلاً در فایل، در دیتابیس SQLite ، در وب، یا هر فضای موجود که برنامه توان دسترسی به آنرا داشته باشد. از طریق مهیا کننده محتوا، سایر برنامه ها میتوانند به جستجو و تغییر دیتای موجود بپردازند (اگر مهیا کننده محتوا اجازه دهد). از طریق Content provider ها دسترسی به تمامی اطلاعات ذخیره شده در گوشی – توسط برنامه های دیگر و یا برنامه ای که ما مینویسیم – امکان پذیر است. برای اینکه سایر برنامه ها بتوانند به اطلاعات دسترسی داشته باشند، معمولاً اطلاعات در فایل ها یا دیتابیس ذخیره میشوند. بعنوان مثال، در سیستم اندروید، مهیا کننده محتوایی قرار گرفته است که به مدیریت اطلاعات تماس کاربر می پردازد. بنابراین، هر برنامه ای با مجوز مناسب میتواند به اطلاعات تماس دسترسی پیدا کند و در آن به جستجو بپردازد.همچنین مهیا کنندگان محتوا برای خواندن از فایل و نوشتن در فایل[2]

---

[1] Framework
[2] Reading and Writing data

مناسب اند که ممکن است این فایل برای برنامه شما خصوصی باشد و قرار نباشد به اشتراک گذاشته شود. به عنوان مثال، برنامه نمونه Notepad از مهیا کننده محتوا به منظور ذخیره سازی فایل استفاده میکند.مهیا کننده محتوا زیرکلاسی از کلاس Content Provider است و می بایست مجموعه ای از APIها را بمنظور انجام ترنزاکشن ها(**Transaction**) اجرا کند.

۴. **دریافت کننده های اعلانات (Broadcast Receivers) :** یک دریافت کننده کامپوننتی است که به پخش اعلانات همگانی سیستم[۱] جواب میدهد. اصالت بسیاری از اعلانات به سیستم بر میگردد. بعنوان مثال، یک اعلان سیستمی اعلام میکند که صفحه نمایش خاموش است ، باتری کم است و یا عکس گرفته شد. همچنین برنامه ها قابلیت راه اندازی[۲] اعلانات را دارند. بعنوان مثال، اجازه دهند تا برنامه های دیگر بدانند که فرضاً دیتا در موبایل دانلود شده است و برای آنها قابل استفاده است. اگرچه دریافت کننده های اعلانات واسط گرافیکی ندارند، ولی میتوانند نوار وضعیتی[۳] برای خود داشته باشند تا کاربر را از اتفاق های رخ داده، باخبر کنند. معمولاً، دریافت کننده اعلانات مانند یک دروازه[۴] برای سایر برنامه ها است و برای انجام کارهای

[1] System-wide broadcast announcements
[2] Initiate
[3] Status bar
[4] Gateway

بسیار کوچک در نظر گرفته میشود. بعنوان نمونه، ممکن است سرویسی را راه اندازی نماید تا با توجه به اتفاقی که رخ میدهد کاری را انجام دهد.یک دریافت کننده اعلانات بعنوان زیر کلاسی از کلاس Broadcast Receiver است و هر اعلان بصورت یک شیء (Intent object) تحویل داده میشود.

در اندروید هر برنامه ای قادر به اجرای کامپوننتی از برنامه ای دیگر است. بعنوان مثال، اگر بخواهید کاربر قادر باشد تا بوسیله دوربین موبایل تصویری بگیرد، بجای آنکه خودتان برنامه ای بنویسید تا عمل عکسبرداری را انجام دهد، ممکن است برنامه دیگری وجود داشته باشد که این کار را برای شما انجام دهد و شما از نتیجه آن استفاده کنید. احتیاجی به کد نوشتن و لینک کردن برنامه ها نیست. بجای آن خیلی راحت میتوانید اکتیویتی مورد نظرتان را اجرا کنید تا عمل عکسبرداری را انجام دهد. وقتی عکس گرفته شد، عکس به برنامه شما تحویل داده خواهد شد تا هرطور که نیاز دارید از آن استفاده کنید. از دید کاربر، دوربین و عمل عکسبرداری بخشی از برنامه شما است که اجرا شده است.

وقتی سیستم کامپوننتی را اجرا میکند، خط پردازش برنامه آغاز میشود (اگر تا حالا اجرا نشده باشد) و کلاسهای مورد نیاز برای اجرای برنامه فراخوانی میشوند. بعنوان مثال، اگر برنامه شما اکتیویتی برنامه دوربین را اجرا نماید تا عکس بگیرد، اکتیویتی در خط پردازشی قرار میگیرد که متعلق به برنامه عکسبرداری است، نه در خط

پردازش برنامه شما. بنابراین، برخلاف انجام پردازش ها در سایر سیستمهای دیگر، برنامه های اندروید یک نقطه شروع معین (single entry point) ندارند.(مثل نقطه شروع ()main در زبانهایی مثل C++ و جاوا)

از آنجاییکه سیستم، اجرای هر برنامه را در خط پردازش مجزا از برنامه های دیگر انجام میدهد و هر برنامه ممکن است مجوزهای مختلف داشته باشد، برنامه شما نمیتواند بطور مستقیم کامپوننتی از دیگر برنامه ها را اجرا کند. اما سیستم اندروید میتواند اینکار را برای شما انجام دهد. بنابراین، برای اجرای اکتیویتی برنامه ای دیگر، باید پیامی [1] به سیستم ارسال کنید تا قصد (intent) شما برای اجرای آن کامپوننت خاص را معلوم کند. و سیستم کامپوننت مورد نظر را برایتان اجرا میکند.

اجرای کامپوننت ها(Activating Components)

سه نوع از چهار نوع کامپوننت ها (اکتیویتی ها، سرویس ها و دریافت کننده ها) با پیام هایی (Asynchronous Message) که به Intent معروفند، اجرا می شوند. Intentکامپوننت ها را در زمان اجرا بهم وصل میکند (اینطوری میتوانید فکر کنید

[1] Message

که کامپوننتی یک درخواستی را ارسال میکند و منتظر جواب می شود)، خواه کامپوننت ها مربوط به برنامه شما باشند یا نباشند.

یک اینتنت با ساخت یک شیء از Intent بوجود می آید؛ با اینکار پیامی تعریف میشود که بوسیله آن میتوان یک کامپوننت خاص یا نوع خاصی از کامپوننت را فعال کرد. یک اینتنت میتواند صریح[1] یا ضمنی[2] باشد.

در اکتیویتی ها و سرویس ها، یک اینتنت معرف عملی برای اجراست (بعنوان مثال برای "دیدن" یا "ارسال" یک چیزی) و ممکن است چیزی را معرفی کند که کامپوننت برای اجرا نیاز به آن داشته باشد. بعنوان مثال، اینتنت ممکن است درخواستی را به یک اکتیویتی ارسال کند که آن اکتیویتی یک تصویری را نمایش دهد یا یک صفحه وبی را باز کند. در بعضی موارد، میتوانید یک اکتیویتی را اجرا کنید تا یک نتیجه ای را دریافت کنید، که در اینصورت، اکتیویتی نیز نتیجه را درغالب یک اینتنت برگشت میدهد (بعنوان مثال، میتوانید یک اینتنت ارسال کنید تا کاربر شماره تماس مورد نظرش را انتخاب کند و این شماره در قالب یک اینتنت به شما برگشت داده شود. بعنوان مثال دیگر، یک اعلان دریافت میکنید که باتری خیلی کم است. این عمل فقط شامل یک رشته "battery is low" است).

---

[1] Explicit
[2] Implicit

نوع آخر کامپوننت، یعنی مهیا کننده محتوا، توسط اینتنت اجرا نمیشود. بلکه، زمانی فعال میشود که درخواستی از یک Content Resolver برسد. وظیفه Content Resolver سر وسامان دادن به ترنزاکشن های ارسالی به مهیا کننده محتواست. این کار به امنیت ارسال و دریافت دیتا کمک میکند زیرا یک سطح دسترسی عادی بین مهیا کننده محتوا و کامپوننت درخواست دهنده را حذف کرده است.

متد های اجرای کامپوننت ها

متدهای مختلفی بمنظور اجرای هر نوع کامپوننت وجود دارد:

۱. یک اکتیویتی با ارسال یک اینتنت به startActivity() اجرا میشود. وقتی که میخواهید اکتیویتی یک نتیجه را به شما برگرداند میتوانید از ارسال اینتنت به متد startActivityForResult() استفاده نمایید.

۲. یک سرویس با ارسال یک اینتنت به startService() یا bindService() اجرا میشود.

۳. یک اعلان[1] توسط ارسال یک اینتنت به متدهایی مثل sendBroadcast() sendOrderedBroadcast()، sendStickyBroadcast() آماده سازی میشود.

۴. با فراخوانی متد query() در یک Content Resolver میتوان در یک مهیا کننده محتوا پرس و جو (query) انجام داد.

فایل مانیفست(The Manifest File)

قبل از اینکه یک سیستم اندروید قادر به اجرای یک کامپوننت باشد، سیستم باید بداند کامپونتی وجود دارد .

این کار با خواندن فایلی بنام AndroidManifest.xmlکه معروف به فایل مانیفست است، انجام میشود. برنامه شما باید تمام کامپوننت ها را در این فایل معرفی کند، که آدرس این فایل دایرکتوری اصلی پروژه تان است. مانیفست شامل قسمت های مختلفی است که کامپوننت ها هم در یکی از این قسمتها میبایست معرفی شوند. چند نمونه از چیزهایی که باید معرفی شوند در زیر آمده اند:

۱. تعریف هر مجوزی[1] که کاربر برای کار با برنامه به آن نیاز دارد، مانند دسترسی به اینترنت یا درخواست دسترسی به اطلاعات تماس کاربر.

[1] Broadcast Receiver

۲. تعریف حداقل سطح دسترسی API ی برنامه، بسته به اینکه برنامه از چه API ای استفاده میکند.

۳. تعریف خصوصیات سخت افزاری و نرم افزاری که توسط برنامه مورد استفاده قرار میگیرد. مانند، دوربین، بلوتوث و صفحه مولتی تاچ.

۴. کتابخانه های API استفاده شده در برنامه مانند استفاده از Google Maps library و موارد دیگر.

تعریف کامپوننت ها در فایل مانیفست

مهمترین وظیفه مانیفست آگاه کردن سیستم اندروید از کامپوننت های برنامه است. بعنوان مثال کد زیر نحوه تعریف یک کامپوننت از نوع اکتیویتی را در مانیفست نشان میدهد.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest...>
  <application android:icon="@drawable/icon" …
    <activity android:name=".MyFirstProject"
          android:label="@string/app_name" ...>
    </activity>
              ...
```

[1] Permission

```
    </application>
</manifest>
```

در تگ <application>، صفت android: icon به یکی از منابع سیستم (که در اینجا آیکون است) اشاره میکند .و برای برنامه یک آیکون تعریف کند.

در تگ(المان) <activity>، صفت android: name اسم کلاس اکتیویتی را مشخص میکند و صفت android: label رشته ای تعریف میکند که به کاربر نشان داده میشود و این رشته به نام اکتیوتی اصلی برنامه اشاره میکند.

تمام کامپوننت های برنامه می بایست بطریق زیر معرفی شوند:

۱. <activity>المانهای اکتیویتی ها

۲. <service>المانهای سرویس ها

۳. <receiver>المانهای دریافت کننده های اعلانات

۴. <provider>المانهای مهیا کننده محتوا

نکته مهم اینکه اکتیویتی ها، سرویس ها و مهیا کننده های محتوایی که در کد[1] برنامه مورد استفاده قرار داده اید ولی در فایل مانیفست تعریف نکرده اید، برای

[1] Source Code

سیستم اندروید قابل شناسایی نیستند ، در نتیجه ، هرگز هم اجرا نمی شوند. اما، دریافت کننده اعلانات علاوه بر اینکه میتواند در مانیفست تعریف شود، بطور داینامیک هم میتواند در کد برنامه ساخته شود.

دریافت کننده اعلانات میتواند بعنوان یک شیء از BroadcastReceiver() و در سیستم، با فراخوانی متد RegisterReceiver() ثبت شود.

معرفی کردن قابلیت های کامپوننت ها در فایل مانیفست

همانطور که در بالا توضیح داده شد، برای اجرا کردن کامپوننت ها (اکتیویتی ها، سرویس ها و دریافت کننده های اعلانات)، باید از اینتنت استفاده شود. باید به صراحت[1] نام کامپوننت مورد نظر (نام کلاس کامپوننت) به اینتنت اعلام شود تا عمل مورد نظر شما اجرا شود. با این حال، قدرت حقیقی اینتنت در مفهوم قدرت عمل (intent actions)آن نهفته است. در این حالت، شما فقط نوع کاری که به آن نیاز دارید را اعلام میکنید (درصورت نیاز میتوانید دیتای مورد نیاز برای آن عمل را نیز ضمیمه کنید) ، و اجازه دهید سیستم برای شما کامپوننت مورد نظرتان را انتخاب کند و آنرا اجرا نماید. اگر چندین کامپوننت مختلف وجود داشت که همگی

[1] Explicit

میتوانستند عمل خواسته شده توسط اینتنت را انجام دهند، حالا کاربر میتواند انتخاب کند که کدام برنامه اجرا شود.

روشی که سیستم در پیش میگیرد تا کامپوننت مورد نظر را انتخاب کند تا به اینتنت جواب دهد بدینگونه است که در مانیفست هر برنامه ای قسمتی به نام intent filtersوجود دارد. سیستم آنرا میخواند و وقتی اینتنتی ارسال میشود با اینها مقایسه میکند، اگر اینتنت با اینتنت فیلتر همخوانی داشت، آن برنامه را اجرا میکند.

وقتی شما کامپوننتی را در مانیفست برنامه تان معرفی میکنید، بصورت اختیاری میتوانید اینتنت فیلتر را – که توانایی های برنامه شما را نشان میدهد – نیز معرفی نمایید. بنابراین با این کار شما برنامه تان را قادر میسازید در صورتیکه برنامه ای دیگر نیاز به استفاده از قابلیت های برنامه شما را داشت، بتواند برنامه تان را اجرا کند و از نتیجه برنامه شما استفاده کند. برای معرفی اینتنت فیلتر به کامپوننت تان باید المان <intent-filter>را بصورت زیر گروه در قسمت معرفی کامپوننت، در فایل مانیفست بیان کنید.

بعنوان مثال، یک برنامه ایمیل با یک اکتیویتی برای ارسال ایمیل، ممکن است اینتنت فیلتری در مانفستش تعریف شده باشد که به اینتنت Send پاسخ دهد (بمنظور ارسال ایمیل). حال یک اکتیویتی در برنامه شما میتواند یک اینتنت ایجاد نماید و عمل send را درخواست کند(ACTION_SEND) ، سپس سیستم وقتی

اینتنت شما را با فیلترهای موجود چک میکند به برنامه ایمیل میرسد و آنرا اجرا میکند.(وقتی شما دستور startActivity() را صادر کنید).

معرفی کردن ملزومات برنامه ها در فایل مانیفست

موبایل های بسیار متنوعی بر مبنای اندروید ساخته شده اند اما همه آنها دارای خصوصیات و تواناییهای یکسان نیستند. بمنظور جلوگیری از نصب برنامه هایتان بر روی تلفن هایی که حداقل نیازهای برنامه تان را ندارند، بسیار مهم است که بصورت دقیق و شفاف ملزومات مورد نیاز برنامه تان ذکر شود تا فقط موبایل هایی که حداقل نیاز برنامه تان را تامین میکنند، توانایی نصب برنامه شما را داشته باشند. این کار با معرفی سخت افزار مورد نیاز و البته نرم افزار در فایل مانیفست انجام میشود. بسیاری از تعاریف فقط برای اطلاع رسانی است و توسط سیستم خوانده نمی شود، اما در سرویس های خارجی مانند مارکت اندروید خوانده میشود تا در هنگام جستجویی که کاربران در مارکت میکنند، برنامه ها فیلتر شوند.

بعنوان مثال، اگر برنامه شما نیاز به دوربین داشته باشد و از API های معرفی شده در سری ۲,۱ استفاده کند(API Level 7) ، می بایست در فایل مانیفست بعنوان ملزومات معرفی شود. بدین ترتیب، موبایل هایی که دوربین ندارند و یا از API های

پایینتر از این سری استفاده میکنند، قادر به نصب برنامه نخواهند بود.و در نتیجه این برنامه توسط مارکت برای آن دسته از گوشی ها نمایش داده نمیشود.

شما همچنین میتوانید بگویید که برنامه تان از دوربین استفاده میکند، ولی دوربین مساله اساسی نیست. در اینصورت، برنامه در ابتدای اجرا شدن چک میکند تا ببیند موبایل دوربین دارد یا نه ؟ و اگر موبایل مقصد دوربین نداشت اگر برنامه شما از خصوصیتی استفاده میکند که به دوربین احتیاج دارد، آن خصوصیت را غیر فعال میکند.

موارد زیر خصوصیات بسیار مهمی است که در حین طراحی و برنامه نویسی باید مد نظر قرار دهید:

اندازه صفحه و تراکم (Screen size and density): بمنظور طبقه بندی موبایل ها بر اساس نوع صفحه نمایشگر، اندروید به معرفی دو خصوصیت برای هر موبایل می پردازد: اندازه صفحه[1] (ابعاد فیزیکی صفحه) و تراکم صفحه (تراکم فیزیکی پیکسل ها در صفحه نمایش که با dpi[2] مشخص میشود). برای ساده سازی معرفی انواع صفحات، سیستم اندروید صفحات را به گروه های مختلف تقسیم بندی کرده است تا راحتتر مورد انتخاب واقع شوند.

---

[1] Dimension
[2] Dots Per Inch

تقسیم بندی براساس اندازه صفحه: کوچک(small) ، معمولی(normal) ، بزرگ (large)و خیلی بزرگ(extra large) .

تقسیم بندی براساس تراکم صفحه: تراکم کم(low density) ، تراکم متوسط (medium density)، تراکم زیاد (high density) و تراکم خیلی زیاد (extra high density)

بصورت پیش فرض، برنامه شما با تمام انواع صفحات و تراکم های مختلف سازگار است، علت این است که سیستم اندروید تنظیمات مناسب را برای واسط کاربری (UI layout)شما و منابع تصویری (image resources) برنامه انجام میدهد. اما، شما میبایست برنامه تان را براساس یک اندازه صفحه خاص و تصاویر را براساس تراکم صفحه طراحی و انتخاب کنید. در فایل مانیفست با استفاده از المان <supports-screens>میبایست اندازه صفحه نمایشگر پشتیبانی شده، ذکر شود.

تنظیمات ورود (Input configurations): ابزارهای مختلف ورودی های متفاوتی برای دریافت اطلاعات از کاربر معرفی کرده اند. چند نمونه از ورودی های مختلف، صفحه کلید سخت افزاری، گوی مسیر[1] و کلیدهای چند جهته[2] هستند. اگر برنامه شما نوع خاصی از ورودی را پشتیبانی میکند باید در فایل مانیفست با

[1] Trackball
[2] Five-way navigation pad

استفاده از المان <uses-configuration>معرفی شود. اما بندرت اتفاق میافتد که برنامه به یک ورودی خاص نیاز داشته باشد.

ویژگی های دستگاه (Device features): خصوصیات سخت افزاری و نرم افزاری متنوعی بر روی دستگاه های مختلف اندروید وجود دارد، مانند دوربین، حسگر نور، بلوتوث، نسخه خاصی از OpenGL و یا صفحه لمسی. هزگز نباید تصور کنید یک ویژگی خاص بر روی تمامی ابزارهای مختلف وجود دارد، بنابراین میبایست در فایل مانیفست با استفاده از المان <uses-feature>به معرفی ویژگیهای خاصی که در برنامه تان استفاده شده است، بپردازید.

نسخه پلتفرم(Platform Version): اغلب، ابزارهای مختلف اندروید از نسخه های متفاوتی استفاده میکنند، مانند اندروید ۱,۶ و اندروید ۲,۳ در نسخه های جدیدتر از API هایی استفاده شده است که در نسخه های قدیمی وجود نداشته است. بمنظور معرفی API های قابل دسترس، هر نسخه از پلتفرم یک سطح API مخصوص دارد، بعنوان مثال اندروید 1.0 شامل API Level 1 است و اندروید 2.3 شامل API Level 9 است. اگر از API هایی استفاده میکنید که بعد از نسخه 1.0 به پلتفرم اضافه شده است، میبایست با استفاده از المان <uses-sdk>حداقل سطح APIرا مشخص کنید.

شاید فکر کنید که رعایت کردن مسائل بالا وقت گیر است و چندان مهم نمیباشد ولی باید بگویم که رعایت این نکات مهم است و بهتر است به نکات بالا توجه کنید و در برنامه مورد استفاده قرار دهید. علت این است ، زمانیکه شما برنامه تان را در مارکت اندروید قرار میدهید، مارکت از این تعاریف استفاده میکند تا برنامه ها را برای موبایل های مختلف فیلتر کند. بدین ترتیب، برنامه شما برای موبایل هایی قابل دسترس خواهد بود که کلیه ملزومات برنامه شما را پشتیبانی کنند.

تعریف مجوزها (Define Permission) : مجوز های مورد نیاز برنامه ، حین نصب برنامه به کاربر معرفی میشوند. برخی از مجوزهای رایج را در زیر میبینید:

- INTERNET : دسترسی به اینترنت
- READ_CONTACTS : فقط خواندن اطلاعات مخاطبان[1]
- WRITE_CONTACTS : مجوز نوشتن در طلاعات مخاطبان
- RECEIVE_SMS : نظارت کردن بر پیامک های ورودی[2]
- ACCESS_COARSE_LOCATION : مجوز دسترسی به وسایلی مانند wifi

[1] Contacts
[2] Incoming SMS(Text Message)

- ACCESS_FINE_LOCATION : مجوز دسترسی به وسایلی مانند GPS

برای مثال ، برای نظارت بر پیامک های ورودی ، شما میتوانید آن را به این صورت در فایل مانیفست تعریف کنید :

```
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
package="com.google.android.app.myapp" >
<uses-permission android:name="android.permission.RECEIVE_SMS" />
</manifest>
```

برای اطلاعات بیشتر میتوانید به اینترنت متصل شوید و مبحث Android security model را از سایت زیر مطالعه کنید :

http://d.android.com/guide/topics/security/security.html

اکتیویتی ها(Activities)

همانطور که اشاره شد، هر اکتیویتی یک کامپوننت از برنامه است که صفحه نمایش را در اختیار میگیرد تا کاربران بتوانند با برنامه ارتباط[1] برقرار کنند، مانند شماره گرفتن برای تماس، عکس گرفتن، ارسال ایمیل و یا دیدن نقشه. به هر اکتیویتی برای نمایش واسط کاربری[2] یک پنجره (در نمایشگر) اختصاص داده خواهد شد. پنجره

[1] Interact
[2] user interface

عموماً کل صفحه نمایش را در بر میگیرد، اما ممکن است کوچکتر از آن هم باشد و بر روی سایر پنجره ها شناور[1] باشد.

نحوه مدیریت حافظه اکتیویتی ها

هر اپلیکیشن یا برنامه معمولاً از چندین اکتیویتی تشکیل میشود که با هم در ارتباط هستند. بطور معمول، یک اکتیویتی در هر برنامه اکتیویتی اصلی است (با نام Main Activity شناخته میشود) و زمانیکه کاربر برنامه را برای بار اول اجرا کرد به او نشان داده میشود. هر اکتیویتی برای انجام وظیفه محول شده به آن میتواند اکتیویتی های دیگر را فراخوانی کند تا کارهای دیگری انجام دهد. هر بار که یک اکتیویتی جدیدی شروع بکار میکند[2]، اکتیویتی قبلی متوقف[3] میشود . اما سیستم اکتیویتی قبلی را در استک[4] نگه میدارد. بنابراین، وقتی اکتیویتی جدیدی اجرا میشود، اکتیویتی قبلی به داخل استک Push میشود و همچنین اکتیویتی های قبلی که در داخل استک قرار داشتند نیز هرکدام یک سطح به پایین تر حل داده خواهند شد. همانطور که میدانید مکانیزم کاری استک هم بصورت "آخر وارد شده، اول خارج شود (LIFO[5]) است". بنابراین، زمانیکه کار کاربر با اکتیویتی فعلی تمام شود و بر

---

[1] float
[2] start
[3] stop
[4] stack
[5] last in, first out

روی دکمه "Back" کلیک کند، اکتیویتی قبلی از داخل استک بیرون کشیده میشود (Pop) و ادامه آن اجرا خواهد شد . چون که ذاتاً پشته با این نوع عملیات سازگاری و همخوانی دارد از ساختمان داده[1] پشته استفاده شده است.

در اندروید فقط یک برنامه در پیش زمینه(foreground) قرار دارد و این برنامه تمام صفحه نمایش را به جز نوار وضعیت[2] در اختیار میگیرد. برای مثال وقتی که شما گوشی خود را روشن میکنید اولین برنامه ای که بر روی صفحه نمایش قرار میگیرد صفحه اصلی(Home Application) است. حالا فرض میکنیم که کاربر برنامه پلیر را اجرا میکند.در این صورت صفحه اصلی به پس زمینه میرود(به داخل استک پوش میشود) و برنامه پلیر به پیش زمینه می آید. بعد برنامه گالری را اجرا میکند.باز هم برنامه پلیر به پس زمینه میرود و برنامه گالری در پیش زمینه قرار میگیرد.اگر شما موزیکی را پخش کرده باشید این موزیک همواره شنیده میشود (حتی موقعی که برنامه گالری را اجرا میکنید).پس این خود نشان میدهد که با اجرا برنامه جدید برنامه های قبلی هم در حالت اجرا هستند ولی در پس زمینه اند و از دید شما خارج اند.اندروید تمام برنامه های اجرایی را در محلی ذخیره میکند.ما این محل را پشته برنامه ها (Application-Stack) می نامیم. وقتی کاربر دکمه

[1] Data Structure
[2] Status line

بازگشت[1] را فشار میدهد برنامه های ذخیره شده در استک خارج میشوند.همواره آخرین برنامه خارج شده هم صفحه اصلی(Home Program) است چونکه اول وارد پشته برنامه ها شده است.و اگر دکمه Back را هنگامی که در صفحه اصلی هستید(صفحه اصلی در پیش زمینه است) فشار دهید هیچ اتفاقی نمی افتد چون که پشته خالی است و هیچ برنامه ای در آن ذخیره نشده است. از دید کاربران اجرای برنامه ها در اندروید خیلی شبیه به بازگشت هایی است که در مرورگرها[2] صورت میگیرد. یعنی همانند صفحات وب با زدن دکمه بازگشت به صفحه قبلی برمیگردیم.

دو حالت ممکن است باعث حذف اکتیویتی از پشته شود.حالت اول زمانی است که کاربر تقاضای بستن برنامه ها را داشته باشد[3] و حالت دوم اینکه به هر دلیلی فضای کافی برای واردن شدن یک برنامه به حافظه وجود نداشته باشد(برنامه جدیدی که به دستور کاربر اجرا شده است). در این صورت سیستم عامل به طور خودکار یکی از اکتیویتی ها را قربانی میکند و از حافظه پشته حذف میکند(این اکتیویتی ممکن است اکتیویتی باشد که اخیراً کمترین دسترسی به آن وجود داشته باشد).

زمانیکه با اجرای یک اکتیویتی، عمل اکتیویتی دیگری متوقف میشود، این تغییر از طریق روش های پاسخگویی در چرخه حیات اکتیویتی[4] شناخته میشود. چندین

---

[1] Back
[2] Web Browser
[3] On User Demand
[4] Activity’s lifecycle callback methods

روش پاسخگویی برای هر اکتیویتی که ناشی از تغییر در وضعیت آن است، وجود دارد. یک اکتیویتی در طول حیات خود ممکن است که اتفاقات زیر برایش رخ دهد :

- سیستم در حال ساخت اکتیویتی است(Creating activity)
- سیستم در حال متوقف کردن آن است(Stopping activity)
- سیستم درحال ازسرگیری اکتیویتی است (Resuming activity)
- سیستم درحال ازبین بردن آن است (Destroying activity).

بنابراین، هر روش پاسخگویی فرصتی را دراختیارتان میگذارد تا بسته به وضعیت حال حاضر سیستم، عمل خاصی که مد نظرتان است را انجام دهید. بعنوان نمونه، زمانیکه برنامه در توقف است، اکتیویتی میبایست آبجکت[1]های حجیم مانند ارتباطات شبکه و دیتابیس را آزاد کند. زمانیکه فعالیت اکتیویتی مجدداً از سر گرفته میشود، میتوانید دوباره منابع مورد نیازتان را دوباره سازی کنید و به ادامه کاری که قبلاً میکردید بپردازید. تمام این انتقال ها بین وضعیت های مختلف اکتیویتی بخشی از چرخه حیات اکتیویتی است.

[1] Object

در ادامه این بخش مفاهیم پایه راجع به ساخت و استفاده از اکتیویتی شرح داده میشود. همچنین، چرخه حیات اکتیویتی بطور کامل مورد بررسی قرار میگیرد تا بتوانید به مدریت وضعیت های مختلف اکتیویتی در چرخه حیات آن بپردازید.

مدیریت چرخه حیات اکتیویتی

مدیریت چرخه حیات اکتیویتی تان با اجرای متدهای پاسخ دهی، بمنظور ساخت و گسترش یک برنامه قدرتمند و قابل انعطاف بسیار حیاتی است. چرخه حیات یک اکتیویتی بطور مستقیم تحت تاثیر سایر اکتیویتی های مر تبط به آن، وظیفه خودش و وضعیتش در پشته (stack) است.

هر اکتیویتی ضرورتاً در یکی از وضعیت های زیر است و از آنجا که همه ی اکتیویتی های شما از کلاس Activity مشتق شده اند و در این کلاس متدهایی برای این وضعیت ها وجود دارد ؛ شما میتوانید با دوباره نویسی[1] کردن این متدها برای اکتیویتی خود رفتار های[2] مناسب برای حالت های مختلف تعریف نمایید.

وضعیت های مختلف یک اکتیویتی در دوره حیات آن :

[1] Override
[2] Behavior

- **در حال ساخت (*onCreate*)** : وقتی که اکتیویتی برای اولین بار اجرا میشود.شما میتوانید در این وضعیت مقادیر اولیه آن را تعیین کنید.

- **در حال شروع (*onStart*) :** زمانی که اکتیویتی ظاهر میشود و کاربر آن را مشاهده میکند.

- **در حال از سرگرفتن (*Resumed*) :** اکتیویتی در صفحه نمایش ظاهر است (foreground)و در دسترس کاربر می باشد.این وضعیت جای مناسبی برای شروع انیمیشن ها و موزیک ها است. (این وضعیت در بعضی مواقع "در حال اجرا" نامیده می شود).

- **در حال توقف (*Paused*):** زمانی که اکتیویتی به پس زمینه میرود و در پشت برنامه دیگری که کل یا بخشی از صفحه نمایشگر را به خود اختصاص داده است، پنهان است. اکتیویتی در حال توقف نیز بطور کامل زنده است (شیئ اکتیویتی در حافظه قرار دارد) و تنها زمانیکه سیستم به شدت به حافظه احتیاج داشته باشد و حافظه آزاد کم باشد، سیستم نسبت به ازبین بردن آن اقدام میکند.

- **در حالت قطع (*Stopped*):** در این حالت اکتیویتی بصورت کامل توسط اکتیویتی دیگری پنهان شده است (اکنون اکتیویتی ما در پس زمینه (background)است). یک اکتیویتی stop شده نیز هنوز زنده است (زیرا هنوز

یک شیئ از Activity در حافظه باقی مانده است) ولی ارتباطش با مدیر پنجره قطع است. بنابراین، در دسترس کاربر نیست و میتواند توسط سیستم زمانیکه به حافظه نیاز است، نابود شود.

- **در حالت شروع مجدد(*onRestart*) :** زمانی این متد اجرا میشود که اکتیویتی از حالت توقف به حالت شروع مجدد میرود و دوباره اکتیویتی ظاهر میشود و به پیش زمینه می آید.

- **در حالت نابود کردن(onDestroy) :** قبل از اینکه اکتیویتی نابود شود این متد فراخوانی میشود. اما تضمینی برای فراخوانی این متد وجود ندارد.یعنی اگر محدویت حافظه وجود داشته باشد و حافظه پر شده باشد ، متد ()**onDestroy** ممکن است هرگز فراخوانی نشود. و خود سیستم عامل پروسه را خاتمه[1] دهد.

- **در حالت ذخیره وضعیت(*onSaveInstanceState*)** : اندروید این متد را فراخوانی میکند تا وضعیت اکتیویتی را ذخیره کند. برای مثال موقعیت مکان نما[2] را در یک Text Field ذخیره و حفظ میکند.معمولاً شما نیاز ندارید که این متد

[1] Terminate
[2] Cursor

را دوباره نویسی[1] کنید زیرا به صورت پیش فرض به نحوی پیاده سازی شده است که وضعیت اکتیویتی را به صورت خودکار ذخیره میکند.

- **در حالت بازگردانی وضعیت قبلی(*onRestoreInstanceState*):** این متد زمانی فراخوانی میشود که اکتیویتی میخواهد وضعیت قبلی خود را که توسط متد ()onSaveInstanceState ذخیره شده، دوباره بدست آورد . این متد هم طوری پیاده سازی شده که به صورت خودکار وضعیت قبلی اکتیویتی را برمیگرداند تا وضعیت رابط کاربری حفظ شود. برای مثال شما در حال پر کردن یک فرم هستید که در همین لحظه تلفن شما زنگ میزند و بعد از این که صحبت تلفنی شما تمام شد ، انتظار دارید که مقادیر را که در فرم نوشته اید ذخیره شده باشند و به وضعیت قبل از زنگ خوردن تلفن برگردید.این دو متد به طور خودکار این کار را انجام میدهند.

چه اکتیویتی در توقف باشد و چه بطور کامل قطع شده باشد، سیستم به راحتی میتواند آنها را از حافظه پاک کند. اینکار یا با اجرای متد ()finish انجام میشود یا خیلی ساده با پاک کردن اثر آن در حافظه. زمانیکه اکتیویتی مجدداً اجرا شود، همه چیز میبایست دوباره ساخته شود.در همین فصل با مثالی عملی با چرخه حیات اکتیویتی ها بیشتر و بهتر آشنا میشوید.

---

[1] Override

ساخت اکتیویتی

برای ساخت یک اکتیویتی، باید یک زیر کلاس از Activity بسازید (یا یک زیر کلاس از کلاسهای موجود). در زیر کلاستان، میبایست متدهای پاسخگویی را اجرا کنید تا سیستم بداند در وضعیت های مختلف چه کارهایی باید انجام دهد، مثلاً زمانیکه اکتیویتی درحال ساخته شدن است، درحال توقف است، درحال از سرگیری کار است و یا در حال از بین رفتن است .دو تا از مهمترین متدهای پاسخگویی در ذیل معرفی شده اند.

onCreate() : بصورت اجباری باید این متد را اجرا کنید. سیستم این متد را زمانیکه درحال ساخت اکتیویتی است، صدا میزند. درحین اجرا، شما میبایست کامپوننت هایی را که در اکتیویتان معرفی کرده اید، مقدار دهی (initialize) کنید. از همه مهمتر، اینجا محلی است که متد setContentView() بمنظور اجرا و ساخت واسط گرافیکی و کاربری برای این اکتیویتی، میبایست معرفی شود.با این متد به سیستم دستور میدهیم که User Interface اکتیویتی را نشان دهد.

onPause() : سیستم زمانی این متد را اجرا میکند که کاربر در حال ترک این اکتیویتی است (ترک اکتیویتی لزوماً به معنی خارج شدن از حافظه استک و نابود

شدن اکتیویتی نیست). اینجا جایی است که معمولاً باید تغییرات مهم مانند قطع موزیک، اجرا شود و یا اینکه بعضی از منابعی که برنامه از آن استفاده میکند را آزاد کنیم. مثلاً اگر از دوربین استفاده میکنیم آن را آزاد کنیم و در هنگام از سر گیری اکتیویتی دوباره آن را بازپس بگیریم.

معرفی کردن اکتیویتی در مانیفست

به منظور استفاده از اکتیویتی در برنامه، قبل از هرچیز اکتیویتی باید به سیستم معرفی شود. برای این منظور، فایل مانیفست برنامه را باز کنید و المان <activity>را بعنوان فرزند[1] المان <application>قرار دهید. بعنوان مثال:

```
<manifest …>
        <application …>
                <activity android: name=".ExampleActivity" />
                …
        </Application…>
</manifest>
```

[1] Child

استفاده از اینتنت فیلتر (intent filters)

هر اکتیویتی ممکن است با استفاده از المان <intent-filter> چندین اینتنت فیلتر را به منظور معرفی، چگونگی فعال کردن این برنامه توسط برنامه های دیگر، معرفی کند.

زمانیکه شما یک برنامه را با ابزار اندروید (SDK) می سازید، یک اکتیویتی که اکتیویتی اصلی شماست با یک اینتنت فیلتر ساخته میشود. این اکتیویتی با عنوان Mainمعرفی میشود و طبقه بندی آن اجرا کننده (launcher) است. تعریف اینتنت فیلتر مانند تکه کد زیر است:

```
<activity android:name=".MyFirstProject"
        android:label="@string/app_name">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category  android:name
              ="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
```

المان <action> میگوید که این اولین اکتیویتی است که باید اجرا شود. المان <category> میگوید که این اکتیویتی میبایست در قسمت اجراشونده های سیستم لیست شود (تا به کاربر اجازه داده شود برنامه را اجرا کند).

اگر دوست ندارید اجازه دهید تا برنامه های دیگر برنامه شما را اجرا کنند، به فیلترهای دیگر نیازی ندارید. تنها یک اکتیویتی باید بصورت main معرفی شود و در طبقه اجراشوندها[1] قرار گیرد، مانند مثال بالا. بنابراین برنامه شما برای سایر برنامه های دیگر قابل دسترس نخواهد بود و باید بصورت صریح اجرا شود.

اما، اگر میخواهید که اکتیویتی تان بصورت ضمنی به سایر اکتیویتی ها جواب دهد، میبایست اینتنت فیلترهای[2] بیشتری به اکتیویتان معرفی کنید.

برای هر نوع منظوری[3] که میخواهید اکتیویتان اجرا شود، باید یک المان <intent-filter> داشته باشید که شامل المان <action> باشد و بصورت اختیاری میتوانید المانهای <category> و <data> را نیز به آن اضافه کنید. این المانها نوع المانی که برنامه باید به آن پاسخ دهد را معلوم میکند.

---

[1] Launcher
[2] Intent Filter
[3] Intent در لغت به معنی منظور ، نیت ، قصد است.

اجرای اکتیویتی

برای اجرای یک اکتیویتی باید متد startActivity() فراخوانی شود و همراه آن میبایست یک اینتنت فرستاده شود تا معلوم کند چه برنامه ای را میخواهید اجرا کنید. اینتنت دو کاربرد دارد، اگر میدانید دقیقاً چه اکتیویتی را میخواهید اجرا کنید میتوانید آن اکتیویتی را با استفاده از اینتنت اجرا کنید. اگر نام اکتیویتی را نمیدانید (مثلاً در شرایطی که میخواهید از اکتیویتی برنامه دیگری استفاده کنید)، میبایست نوع فعالیتی که مورد نظرتان است را شرح دهید. در این شرایط سیستم اکتیویتی مناسب را برای شما پیدا میکند (که میتواند اکتیویتی پیشنهادی، مربوط به برنامه دیگری باشد). همچنین، اگر اکتیویتی که قرار است اجرا شود نیاز به اطلاعات اولیه داشته باشد، اینتنت میتواند حجم کوچکی از اطلاعات را نیز حمل کند.

وقتی مشغول کار بر روی برنامه خود هستید، معمولاً میدانید به چه اکتیویتی هایی نیاز دارید و اکتیویتی هایتان چه قابلیت هایی دارند. بنابراین برای اجرای یک اکتیویتی میدانید نام آن چیست. بنابراین خیلی واضح – با استفاده از اسم کلاس – باید بگویید چه اکتیویتی بایست اجرا شود. برای مثال، در کد زیر نشان میدهیم چطور یک اکتیویتی، اکتیویتی دیگری با نام SignInActivity را صدا میزند.

```
Intent  intent  = new Intent(this , SignInActivity.class);
startActivity(intent);
```

اما، زمانی نیز ممکن است اکتیویتی شما نیاز داشته باشد کارهای دیگری مانند ارسال ایمیل یا ارسال SMS با استفاده از اطلاعات برنامه تان، انجام دهد. در این شرایط، برنامه شما ممکن است اکتیویتی مورد نیاز برای انجام این کار را نداشته باشد. بنابراین، درعوض شما میتوانید از قدرت و امکانات سایر برنامه ها که بر روی وسیله نصب هستند، استفاده کنید تا درخواست مورد نظرتان را انجام دهند. اینجاست که پی به قدرت واقعی اینتنت ها میبریم. میتوانید یک اینتنت بسازید و بگویید نیاز به چه کاری دارید و سیستم میگردد و اکتیویتی مورد نیازتان را در میان برنامه های موجود پیدا میکند. اگر چندین اکتیویتی مختلف برای اجرای منظور شما موجود باشد، کاربر میتواند انتخاب کند کدام برنامه اجرا شود. بعنوان مثال، اگر بخواهید یک ایمیل ارسال کنید، میتوانید بصورت زیر اینتنت تان را بسازید.

```
Intent intent = new Intent (Intent.ACTION_SEND);
intent.putExtra(Intent.EXTRA_EMAIL , recipientArray);
startActivity(intent);
```

EXTRA_EMAIL که به اینتنت اضافه شده، یک آرایه رشته ای است که در آن آدرس های ایمیل نگهداری میشود. اطلاعات به این آدرس ها ارسال میشود. وقتی برنامه ارسال ایمیلی به این اینتنت پاسخ میدهد، ابتدا آدرس ایمیل ها که در آرایه ذخیره شده است را میخواند و آنها را در قسمت "to:" ایمیل قرار میدهد. سپس، اکتیویتی برنامه ارسال ایمیل اجرا میشود و بعد از اتمام کار به اکتیویتی شما برمی گردد.

اجرای یک اکتیویتی برای دریافت نتیجه

بعضی مواقع نیاز است تا نتیجه ای را از اکیتیویتی که اجرا کرده اید دریافت کنید. در این شرایط، بجای اجرای اکتیویتی با متدstartActivity() ، اکتیویتی را باید با استفاده از متد startActivityForResult() اجرا کرد. سپس برای دریافت جواب یا نتیجه از اکتیویتی مورد درخواست، متد پاسخگویی onActivityResult() میبایست اجرا شود.

وقتی اکتیویتی کارش تمام شد، نتیجه را به اینتنت در متد startActivityForResult()باز میگرداند.

به عنوان مثال، فرض کنید میخواهید کاربر مشخصات یکی از افرادی که در دفتر تلفن موبایل خود دارد را انتخاب کند و شما در اکتیویتی تان عملی را بر مبنای آن انجام دهید. کد پایین نشان میدهد که چگونه میتوانید یک اینتنت بسازید و از نتیجه استفاده کنید.

```
Private void pickContact(){
//Create an intent to "pick" a contact, as defined by the content provider URI
Intent intent;
intent = new Intent(Intent .ATION_PICK,Contacts.CONTENT_URI);
startActivityForResult(intent,PICK_CONTACT_REQUEST);
}
@Override
Protected void onActivityResult(int  requestCode,int  resultCode, Intent data) {
```

```
/* if the request went well(ok) and the request was
        PICK_CONTACT_REQUEST */

If(resultCode == Activity.RESULT_OK  && requestCode ==
PICK_CONTACT_REQUEST) {

//Perform a query to the contact ' s content provider for the contact 's
name

Cursor cursor = getContentResolver().query(data.getData())

new String[] {Contacts.DISPLAY_NAME},null,null,null);

if(cursor.moveToFirst()) { //True if the cursor is not empty

        int cloumnIndex =
        cursor.getColumnIndex(Contacts.DISPLAY_NAME);

        String name = cursor.getString(columnIndex);

        //Do Something with the selected contact 's name …

}
```

تکه کد فوق نشان دهنده منطقی است که در استفاده از متد onActivityResult() بمنظور استفاده از جواب برگشت داده شده، میبایست بکار گرفته شود. اولین شرط بررسی میکند که آیا درخواست موفقیت آمیز انجام شده یا نه. اگر انجام شده بود، result Code برابر با RESULT_OK خواهد بود. همچنین، بررسی میکند که آیا پاسخ شناخته شده ای به درخواست داده شده است یا خیر. در این حالت requestCode برابر با پارامتری میشود که توسط startActivityForResult() ارسال شده است. از اینجا به بعد، کد نتیجه

بازگشتی از اکتیویتی را با استفاده از query در دیتای برگشت داده شده، مورد استفاده قرار میدهد(پارامتر data).

آن چیزی که اتفاق میافتد این است که Content Resolver یک پرس و جو (query)در تامین کننده محتوا (content provider) انجام میدهد، که این عمل باعث برگشت داده شدنCursor میشود تا دیتای پرس و جو شده قابل خواندن شود.

خاتمه دادن به اکتیوتی

با فراخوانی متد ()finish میتوانید به فعالیت یک اکتیویتی خاتمه دهید.

شما همچنین میتوانید به فعالیت یک اکتیویتی دیگر را که قبلاً اجرا کرده بودید، با استفاده از متد ()finishActivity خاتمه دهید.

هشدار :در بسیاری از مواقع، مستقیماً با استفاده از این دستور نباید یک اکتیویتی را متوقف کنید. بر اساس آنچه در چرخه حیات اکتیویتی گفته خواهد شد، سیستم اندروید بجای شما به مدریت حیات اکتیویتی می پردازد. بنابراین شما نیازی نیست فعالیت اکتیویتی را متوقف کنید. از این متد تنها زمانی باید استفاده شود که قطعاً نمی خواهید کاربرتان به این اکتیویتی بازگردد.

پیاده سازی پاسخگوهای چرخه حیات

وضعیت های مختلف اکتیویتی در بالا توضیح داده شد و گفته شد که پاسخگوهای[1] مختلفی برای وضعیت های مختلف اکتیویتی وجود دارد. تمام متدهای پاسخگویی مانند قلاب هستند که شما میتوانید آنها را برای شرایط مختلفی که ممکن است برای اکتیویتی اتفاق بیافتد، برنامه ریزی کنید. کد پایین نشان دهنده اسکلت و چارچوب اکتیویتی برای پاسخگویی به شرایط مختلف است.

کد برنامه چرخه حیات:

```
public class ExampleActivity extends Activity {
        @override
        public  void onCreate(Bundle  savedInstanceState) {
        super.onCreate (savedInstanceState);
        //The activity is being created.
}
        @override
        protected void onStart() {
                super.onStart();
                Toast.makeText(this, "onStart", 1).show();
        }
        @Override
```

[1] callback

```
protected void onRestart() {
        super.onRestart();
        Toast.makeText(this, "onRestart", 1).show();
}
@Override
protected void onResume() {
super.onResume();
Toast.makeText(this, "onResume", 1).show();
}
@Override
protected void onPause() {
        super.onPause();
        Toast.makeText(this, "onPause", 1).show();
}
@Override
protected void onStop() {
        super.onStop();
        Toast.makeText(this, "onStop", 1).show();
}
@Override
protected void onDestroy() {
        super.onDestroy();
        Toast.makeText(this, "onDestroy", 1).show();
```

}

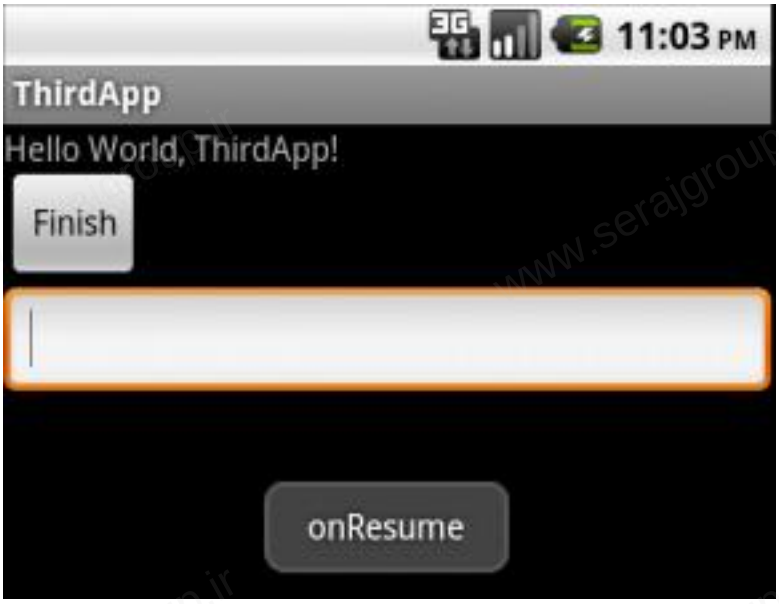
توجه: همانطور که در مثال فوق می بینید، برای اجرای هرکدام از متدهای فوق، قبل از هر کاری می بایست کلاس اصلی (super class) صدا زده شود.

```
Toast.makeText(this, "onDestroy", 1).show();
```

تکه کد بالا باعث میشود که متن داخل کوتیشن به اندازه یک ثانیه بر روی صفحه نمایش نشان داده شود(عدد ۱ در کد Interval یا فاصله زمانی است).

در شکل های ( )خروجی های برنامه مربوط به چرخه حیات را مشاهده میکنید.اگر این تصاویر را با فاصله ی یک ثانیه از هم تصور کنید.دقیقاً خروجی برنامه به همین صورت است.


10:47 PM
ThirdApp
Hello World, ThirdApp!
Finish
onStart


11:03 PM
ThirdApp
Hello World, ThirdApp!
Finish
onResume


10:54 PM
ThirdApp
Hello World, ThirdApp!
Finish
با کلیک بر روی دکمه Finish


Email
Gallery
Messaging
Music
Phone
Settings
Spare Parts
Speech Recorder
ThirdApp
onPause

تذکر۱ : با اجرای برنامه برای اولین بار متد های زیر اجرا میشوند :

1.onCreate() 2.onStart() 3.onResume()

تذکر۲ : زدن دکمه Back معادل با کلیک بر روی دکمه Finish برنامه است. و باعث اجرای متدهای زیر میشود :

1.onPasue() 2.onStop() 3.onDestroy()

تذکر۳: کلیک بر روی دکمه خانه (Home key) باعث اجرای متد های زیر میشود :

1.onPause() 2.onStop()

تذکر ۴: با چرخاندن صفحه نمایش متدهای زیر اجرا میشود :

1.onPause() 2.onStop() 3.onDestroy()

4.onCreate() 5.onStart() 6.onResume()

برای چرخاندن صفحه نمایش شبیه ساز میتوانید دکمه ترکیبی Control+F11 را فشار دهید و یا دکمه ۷ یا ۹ قسمت ماشین حساب[1] (کلید Num Lock باید روشن باشد) کیبرد را فشار دهید. یک راه سریع برای تست کد های خود چرخاندن صفحه نمایش[2] است. با چرخاندن صفحه نمایش متدهای فوق اجرا میشود ، که اگر شما در کد برنامه خود به خوبی وضعیت های برنامه را ذخیره و کنترل کرده باشید برنامه به درستی کار میکند. و وضعیت های خود را حفظ میکند.

[1] Keypad
[2] Change Screen Orientation

با کلیک بر روی دکمه خانه (Home key) متد ()onDestroy اجرا نمیشود. به همین خاطر برنامه در حافظه باقی میماند و از بین نمیرود (برنامه به لیست برنامه های موجود در پشت صحنه اضافه میشود و اگر برنامه Task Manager را اجرا کنید از حضور برنامه در حافظه مطمئن میشوید) .

با نگه داشتن دکمه Home key میتوانید لیست برنامه های اخیر را ببینید و برنامه را دوباره اجرا کنید.که در این صورت متدهای زیر اجرا میشوند :

1.onRestart() 2.onStart() 3.onResume()

همه این متدها با هم، چرخه حیات اکتیویتی را تشکیل میدهد. با اجرای این متدها میتوانید شاهد سه حلقه تودرتو در چرخه حیات اکتیویتی باشید:

۱. طول عمر اکتیویتی **(Entire Lifetime):** طول عمر یک اکتیویتی با فراخوانی متد ()onCreate شروع می شود و تا فراخوانی متد ()onDestroy ادامه پیدا میکند. وضعیت عمومی اکتیویتی (مانند حالت گرافیکی و واسط کاربری اکتیویتی) باید در متد ()onCreate تعریف شود. و تمام منابع(Resource) استفاده شده باید در متد ()onDestroy آزاد شوند. بعنوان مثال اگر اکتیویتی تان شامل نخی[1] باشد

[1] Thread

که در پشت صحنه به منظور دانلود دیتا از شبکه دارد اجرا میشود، این thread می بایست در متد ()onCreate ساخته شود و در متد()onDestroy متوقف شود.

۲. طول عمر قابل رؤیت (Visible Lifetime): فاصله بین فراخوانی متدهای ()onStart و ()onStop است. در طی این زمان، برنامه برای کاربر قابل مشاهده است و میتواند با آن کار کند. بعنوان مثال، متد ()onStop زمانی صدا زده میشود که اکتیویتی جدیدی اجرا شده و اکتیویتی ما دیگر قابل دیده شدن و در دسترس نیست. در فاصله اجرای این دو متد میتوانید از منابع مورد نیازتان در اکتیویتی تان استفاده کنید تا به کاربر نشان داده شود. بعنوان مثال، شما میتوانید در متد ()onStart یک BroadcastReceiver را به منظور آگاه شدن از تغییرات انجام شده در محیط کاربری، ثبت[1] کنید. هچنین برای «خارج کردن آن از ثبت»[2] باید در متد ()onStop اینکار را انجام دهید. در طی طول عمر اکتیویتی، سیستم ممکن است چندین بار متدهای ()onStart و ()onStop را اجرا کند. البته این عمل بدستور کاربر انجام میشود چراکه بعضاً ممکن است چندین بار بین برنامه های مختلف و برنامه شما حرکت کند.

۳. طول عمر قابل دسترس بودن (Foreground Lifetime): این طول عمر اکتیویتی فاصله بین فراخوانی متدهای ()onResume و ()onPause است. در طی این

[1] register
[2] unregister

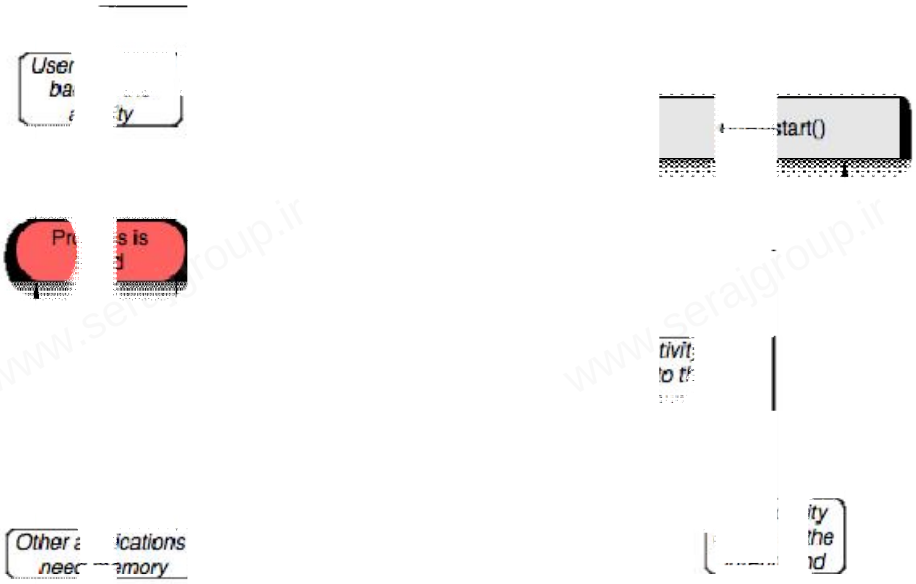
مدت، اکتویتی بر روی سایر اکتیویتی ها در حال اجرا شدن است و صفحه نمایشگر را دراختیار دارد و همچنین کاربر میتواند با آن ارتباط داشته باشد. اکتیویتی میتواند چندین بار بین این دو وضعیت حرکت کند، مثلاً متد()onPause زمانی فراخوانی میشود که موبایل بخواهد به sleep برود و یا یک پیامی در صفحه نمایشگر به کاربر نشان داده شود. از آنجا که این تغییر وضعیت ها اغلب رخ میدهد، بمنظور جلوگیری از تلف کردن وقت کاربر، کدهای این قسمتها نباید سنگین باشد تا کاربر را منتظر نگهدارد.

شکل( ) نشان دهنده این حلقه ها و مسیرهایی است که یک اکتیویتی ممکن است در حرکت بین وضعیت های مختلف طی کند. در این شکل مستطیل ها بیان کننده متدهای پاسخگویی به وضعیت های مختلف است که میتوانید آنها را کد نویسی کنید تا پاسخ مورد نظرتان را به وضعیت های مختلف بدهید.میتوان از بحث چرخه حیات این نتیجه گیری را کرد که مهمترین متد برنامه ی ما متد ()onCreate است و همیشه باید نوشته شود.برای اینکه اهمیت متد های دیگر را هم بفهمیم ، اگر به شکل نگاه کنید متوجه میشوید که اگر برنامه ما از ()onCreate شروع شود نهایتاً به متد ()onResume میرسد و اگر برنامه به حالت Stop برود باز برنامه در شروع مجدد(()onRestart) به متد ()onResume میرسد. لذا در هر صورت خط اجرای برنامه به متد ()onResume میرسد و این متد همیشه اجرا میشود.پس متد ()onResume هم از متد های مهم

است.متد onPause() هم به همین صورت متد مهمی است.چون که برنامه ما اگر قرار است که نابود شود و یا متوقف شود در هر صورت متد onPause() اجرا میشود.پس در برنامه هایی که نوشته میشود حداقل باید کد مربوط به متدهای onCreate(),onResume(),onPause() را بنویسید.

در جدول ( ) مجدداً متدهای پاسخگویی چرخه حیات را این بار با جزئیات بیشتری تشریح میکند. همچنین موقعیت و حلقه هرکدام را نسبت به سایر موقعیت ها و حلقه های دیگر نشان میدهد. همچنین نشان میدهد که آیا سیستم میتواند اکتیویتی را بعد از اینکه متد پاسخگویی آن اجرا شد، نابود کند.

ستون سوم جدول با عنوان " قابل نابود شدن بعد از اجرا؟ " ، نشان میدهد که آیا سیستم میتواند اکتیویتی را بعد از انجام وظیفه اش در آن متد (بدون اینکه حتی یک خط دیگر از کدهای نوشته شده برای اکتیویتی را اجرا کند)، نابود کند.


start()
s is
Other a
ications
emory
ity
the
id

متد	توضیح	قابل نابود شدن بعد از اجرا؟	متد بعدی
onCreate()	این متد زمانیکه اکتیویتی برای اولین بار ساخته میشود، اجرا میشود. در اینجا شما باید تمام خصوصیات ثابت برنامه تان مانند ساخت Viewها، قرار دادن اطلاعات در لیست و غیره را انجام دهید. این متد دارای یک آبجکت است که حاوی مجموعه ای از اطلاعاتِ وضعیت قبلی اکتیویتی است، البته درصورتیکه وضعیت قبلی آن ذخیره شده باشد (SavingActivityState را جلوتر بیان میکنیم). همیشه بعد از آن متد onStart() باید اجرا شود.	خیر	onStart()
onRestart()	این متد زمانی اجرا میشود که اکتیویتی بخواهد از حالت قطع یا توقف کامل (Stop) خارج شود، همیشه بعد از آن متد onStart() باید اجرا شود.	خیر	onStart()
onStart()	درست قبل از اینکه اکتیویتی برای کاربر قابل مشاهده باشد، فراخوانی می شود.	خیر	onResume() یا onStop()

onResume()	این متد دقیقاً قبل از اینکه اکتیویتی بخواهد با کاربر تعاملش را آغاز کند، فراخوانی میشود. در این زمان، اکتیویتی در بالاترین نقطه انباره (Stack)قرار دارد و میتواند با کاربر در تعامل باشد . همیشه بعد از آن متدonPause() باید اجرا شود.	خیر	onPause()
onPause()	این متد زمانی فراخوانی میشود که اکتیویتی دیگری میخواهد اجرا شود .بطور معمول از این متد برای انجام کارهایی مانند ذخیره اطلاعاتی که هنوز ذخیره نشده اند، توقف انیمیشن و یا سایر کارهایی که CPU را بخودش درگیر کرده است، استفاده میشود. هر کاری که قرار است انجام شود باید بسیار سریع صورت گیرد زیرا تا زمانیکه وظیفه این متد انجام نشود، اکتیویتی بعدی اجرا نمیشود. اگر اکتیویتی مجدداً بخواهد به نمایش درآید .متد بعد از این onResume() است و یا اگر بخواهد از دید و دسترس کاربر پنهان شود متد بعدی onStop()است.	بلی	onResume() یا onStop()
onStop()	زمانی اجرا میشود که اکتیویتی دیگر در دسترس کاربر نیست. این متد	بلی	onRestart() یا onDestroy()

	ممکن است به دلیل اینکه اکتیویتی دیگری (چه اکتیویتی ای که قبلاً اجرا شده و الان در استک است و چه اکتیویتی جدید باشد) کارش را از سر گرفته است و دستور اینکار را داده ، اجرا میشود . اگر اکتیویتی میخواهد برگردد تا در دسترس کاربر قرار گیرد متد بعد از آن ()onRestart است و یا اگر اکتیویتی بخواهد نابود شود ()onDestroy است.		
onDestroy()	این متد دقیقاً قبل از اینکه اکتیویتی بخواهد نابود شود، اجرا میشود. بعبارت دیگر این متد آخرین فراخوانی است که اکتیویتی دریافت خواهد کرد. این فراخوانی ممکن است به این دلیل باشد که اکتیویتی به پایان انجام وظیفه اش رسیده است (کسی متد ()finish را اجرا کرده است)، یا سیستم بمنظور در اختیار گرفتن فضای بیشتر نیاز دارد تا بعضی از اکتیویتی ها را نابودکند. متد ()isFinishing وجه تمایز دو سناریوی فوق است.	بلی	متدی اجرا نمیشود.

متدهایی که جواب منفی در ستون ” قابل نابود شدن بعد از اجرا؟ ” داده اند، در مقابل نابود شدن توسط سیستم حفاظت شده اند. بنابراین، وقتی اکتیویتی به متد ()onPause پاسخ می دهد و نابود نمیشود، حتماً به متد ()onResume برخواهد

گشت و دیگر این متد قابل نابود شدن نیست تا زمانیکه مجدداً متد ()onPause فراخوانی شود.

زمانیکه یک اکتیویتی ساخته میشود، متد onPause() آخرین متدی است که ضمانت شده تا قبل از نابودی اکتیویتی حتماً اجرا شود. اگر در حالت اضطرار ، سیستم مجبور شود تا حافظه را بازیابی[1] کند، متدهای onStop() و onDestroy() ممکن است فراخوانی نشوند و تضمینی در ارجای این متد های وجود ندارد. بنابراین، به منظور ذخیره داده های مهم (مانند مشخصات کاربر) بر روی سیستم، میبایست در متد onPause() این کار را انجام دهید. اما، در مورد اینکه چه اطلاعاتی باید در طی فرایند این متد نگهداری شود، خیلی محتاط باشید زیرا هر عملی که سرعت پردازش را پایین بیاورد عمل انتقال به اکتویتی بعدی را کندتر میکند.

توجه: اکتیویتی هایی که از لحاظ تکنیکی قابل نابود شدن نیستند، فقط در زمانی که منابع سیستم به شدت پایین باشد، توسط سیستم نابود میشوند. نابودی اکتیویتی ها در بخش پردازش و نخ ها (Processes and Threading) با جزئیات بیشتری شرح داده شده است.

ذخیره کردن وضعیت اکتیویتی

در "مدیریت چرخه حیات اکتیویتی" گفتیم که وقتی اکتیویتی متوقف (Pause) می شود یا به حالت قطع (Stop) می رود، وضعیت اکتیویتی حفظ می شود. علت این است

[1] Recover

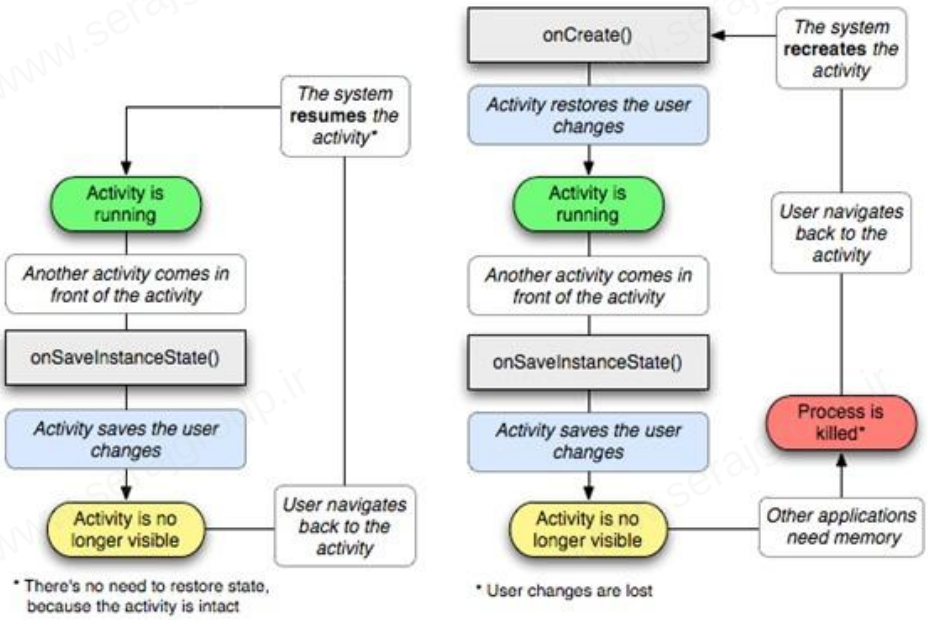
که یک شیئ [1] از Activity هنوز در حافظه باقی مانده است و تمام اطلاعات در مورد سایر اکتیویتی های مرتبط با این اکتیویتی و وضعیت فعلی هنوز در دسترس است. بنابراین، هر تغییری که کاربر در اکتیویتی بوجود آورد در حافظه نگهداری می شود. بنابراین، زمانیکه اکتیویتی مجدداً به صفحه نمایش بر میگردد (Resume) آن تغییرات مشاهده می شود.

اما زمانیکه سیستم، اکتیویتی را بمنظور بازیابی حافظه نابود می کند، شیئ Activity نیز نابود می شود. بنابراین، سیستم قادر نخواهد بود بسادگی و مانند حالت قبل اکتیویتی را مجدداً اجرا (Resume) نماید. در عوض سیستم، زمانیکه کاربر مجدداً بخواهد به آن برگردد، میبایست شیئ اکتیویتی را مجدداً بسازد . کاربر، هنوز نمی داند که سیستم شیئ اکتیویتی قبلی را نابود کرده و این یک شیئ جدید است و بنابراین انتظار دارد که اکتیویتی، همان اکتیویتی قبلی باشد. در این وضعیت، شما باید مطمئن شوید که اطلاعات مهم راجع به وضعیت اکتیویتی ذخیره می شود. این عمل با اضافه کردن متدی دیگر که به شما اجازه می دهد تا وضعیت اکتیویتی را ذخیره کنید، انجام می شود. زمانیکه کاربر مجدداً اکتیویتی را اجرا کند، سیستم اکتیویتی جدید را باتوجه به وضعیت قبلی اکتیویتی می سازد.

[1] object

onSaveInstanceState() متدی است که شما می توانید با آن وضعیت فعلی اکتیویتی تان را ذخیره کنید. سیستم این متد را قبل از اینکه اکتیویتی نابود شود و به شیئ Bundle فرستاده شود، فرا میخواند.Bundle جایی است که شما می توانید اطلاعات مربوط به وضعیت اکتیویتی تان را مانند نام ها با استفاده از متدputString() ، ذخیره کنید. حال اگر سیستم اکتیویتی شما را نابود کند و کاربر به اکتیویتی برگردد، سیستم Bundle را به onCreate() ارسال میکند و بنابراین با این روش شما می توانید وضعیت قبلی اکتیویتی تان را که با استفاده از متد onSaveInstanceState() ذخیره کرده بودید، بازیابی کنید. اگر اطلاعاتی برای بازیابی موجود نباشد، مقدارBundle ی که به onCreate() ارسال شده است، تهی[1] خواهد بود.

[1] Null



شکل ۲ – دو حالتی که برای اکتیویتی زمانیکه می خواهد مجدداً اجرا شود اتفاق می افتد. اول، زمانیکه اکتیویتی متوقف می شود و سپس مجدداً اجرا می شود و وضعیت آن مانند قبل است (شکل سمت چپ). دوم، اکتیویتی نابود می شود و سپس دوباره سازی می شود و سیستم می بایست مجدداً وضعیت سیستم را مانند قبل نشان دهد (شکل سمت راست).

هیچ ضمانتی وجود ندارد که قبل از نابودی اکتیویتی متد onSaveInstanceState() اجرا شود. علت این است که در بعضی از موارد نیازی نیست که وضعیت اکتیویتی ذخیره شود (مانند زمانیکه کاربر با استفاده از دکمه

برگشت (BACK key) اکتیویتی شما را ترک می کند، چراکه کاربر به صراحت میگوید که میخواهد از برنامه خارج شود). اگر قرار بر فراخوانی متد فوق باشد، همیشه قبل از متد ()onStop و احیاناً قبل از متد ()onPause فراخوانی می شود.

اما اگر شما هیچ کاری نکنید و از متد ()onSaveInstanceState استفاده نکنید بعضی از وضعیت های اکتیویتی توسط متد()onSaveInstanceState که در کلاس Activity بصورت پیش فرض وجود دارد، بازیابی می شود. بطور خاص، متد ()onSaveInstanceState برای هر View در چارچوب (layout) برنامه اجرا می شود تا هر المان از وضعیت خودش باخبر باشد. تقریباً تمام ویجت[1] ها در چارچوب سیستم اندروید این متد را هرجا که نیاز باشد، اجرا می کنند. مثلاً هر زمان که تغییر در واسط کاربری اتفاق بیافتد این رخ داد با استفاده از متد فوق در سیستم ذخیره می شود و در زمان مورد نیاز اکتیویتی با استفاده از این اطلاعات، بازسازی می شود. بعنوان مثال، ویجت EditText متون وارد شده به آن توسط کاربر را ذخیره میکند یا مثلاً ویجت CheckBox انتخاب شدن یا نشدن گزینه ها را ذخیره میکند. تنها کاری که شما باید بکنید این است که یک ID منحصر بفرد با استفاده از صفت android:id به هر ویجتی که میخواهید وضعیتش را ذخیره کنید اختصاص دهید. اگر ویدجتی ID نداشته باشد قادر به ذخیره وضعیت خودش نخواهد بود.

[1] Widget

اگرچه اجرای متد onSaveInstanceState() بصورت پیش فرض اطلاعات مفیدی را در مورد تغییرات واسط کاربری ذخیره میکند ؛ ولی ممکن است شما نیازمند ذخیره کردن اطلاعات بیشتر باشید در این صورت بایستی این متد را دوباره نویسی[1] کنید. بعنوان مثال، شاید بخواهید تعداد مراجعات کاربر به اکتیویتی را نیز بدانید که این کار با دوباره نویسی متد انجام پذیر است.

از آنجاکه اجرای پیش فرض متد onSaveInstanceState() کمک میکند تا وضعیت واسط کاربری را ذخیره کنید، اگر شما بخواهید این متد را بمنظور ذخیره اطلاعت بیشتر، دوباره نویسی کنید، همیشه و قبل از هرکاری میبایست سوپر کلاس (super class)را فرخوانی کنید.

توجه: از آنجاکه اجرای متد onSaveInstanceState() هیچگاه ضمانت[2] نشده است، بایستی از این متد فقط برای ذخیره وضعیت اکتیویتی (یا وضعیت واسط کاربری) استفاده شود و هیچگاه نباید برای ذخیره دیتا(اطلاعات کاربری) استفاده شود. در عوض شما میتوانید از متدonPause() بمنظور ذخیره سازی اطلاعاتتان استفاده کنید (مثلاً اطلاعاتی که باید در دیتابیس ذخیره شوند).

[1] Override
[2] Guarantee

یک راه خوب و سریع بمنظور تست اینکه آیا برنامه شما قابلیت ذخیره سازی موقعیتش را دارد این است که وسیله (مثلاً موبایل) خود را بچرخانید، بنابراین صفحه نمایش تغییر جهت[1] می دهد. وقتی صفحه تغییر جهت داد، سیستم ابتدا اکتیویتی را نابود میکند و سپس آنرا دوباره سازی میکند. اینکار برای اعمال منابع جایگزین برای این حالت از صفحه نمایش که ممکن است موجود باشد، انجام می شود.فقط به همین دلیل، خیلی مهم است که اکتیویتی شما بطور کامل وضعیت قبلیش را بعد از دوباره سازی بازیابی کند. چراکه کاربر بسیاری مواقع برای استفاده از سایر برنامه ها نیاز دارد تا موبایل خود را بچرخاند.

اداره کردن تغییرات پیکربندی

بعضی مواقع تغییر در پیکره بندی[2] در موقع اجرا برنامه[3] اتفاق میافتد (مانند چرخش صفحه، دسترسی به کیبورد و زبان). وقتی این چنین تغییری بوجود آمد، سیستم اندروید اکتیویتی که در حال اجراست را مجدداً راه اندازی (restarts) میکند (متد onDestroy() را فرامیخواند و بلافاصله متدonCreate() را اجرا میکند). رفتار راه اندازی مجدد بمنظور کمک به برنامه شما طراحی شده است تا با تغییر جدید

[1] Orientation
[2] Configuration
[3] Runtime

هماهنگ شود و اگر نیاز به منابع جدیدی داشت که در برنامه شما به آن اشاره کرده بودید، از آنها استفاده شود. اگر برنامه تان را جوری بنویسید که این تغییر را اداره کند، برنامه شما قابلیت انعطاف[1] بیشتری در برخورد با اتفاقات غیر منتظره پیدا خواهد کرد.

بهترین راه برای اداره تغییرات پیکربندی (مانند تغییر در جهت صفحه نمایش، ذخیره وضعیت برنامه) استفاده از متد های زیر است :

- onSaveInstanceState()
- onRestoreInstanceState()
- onCreate()

هماهنگ کردن اکتیویتی ها

وقتی یک اکتیویتی، اکتیویتی دیگری را اجرا میکند، چرخه حیات هرکدام دستخوش تغییر می شود. اکتیویتی اول متوقف (Pause) و سپس قطع (Stop) می شود و دومین اکتیویتی ساخته می شود. ممکن است هر دو اکتیویتی بخواهند به یک فایل دسترسی داشته باشند یا اطلاعاتی را به اشتراک بگذارند، خیلی مهم است که توجه کنید که

[1] Flexibility

اکتیویتی اول تا زمانیکه اکتیویتی دوم کاملاً ساخته نشده است به حالت قطع نمی رود. بلکه، پردازش های ساخت اکتیویتی دوم با توقف اکتیویتی اول همپوشانی[1] پیدا میکند.

در اینجا مراحل عملیات انجام شده بر روی دو اکتیویتی A و B نشان داده می شود:

۱. متد onPause() اکتیویتی A اجرا می شود.

۲. متدهای onCreate()، onStart() و onResume() از اکتیویتی B بترتیب اجرا می شود. (اکتیویتی دوم به کاربر نشان داده می شود).

۳. حال اگر اکتیویتی A مدت طولانی باشد که کاربر از آن استفاده نکرده باشد، متد onStop() آن اجرا می شود.

این چرخه قابل پیش بینی به شما در اداره وضعیت های مختلف اکتیویتی کمک خواهد کرد. بعنوان مثال، اگر باید چیزی در پایگاه داده[2] بنویسید که اکتیویتی دوم از آن استفاده کند، باید آنرا در وضعیت onPause() اکتیویتی اول بنویسید، قبل از اینکه متدonStop() اکتیویتی اجرا شود.

تبادل اطلاعات بین اکتیویتی ها

[1] Overlap
[2] Data Base

در جابجایی بین اکتیویتی ها دو حالت وجود دارد. حالت اول اینکه اکتیویتی های اول و دوم با هم رابطه ای ندارند. در این حالت با استفاده از Intent و startActivity()، اکتیویتی دوم را صدا میزنیم و بعد از آن صفحه نمایش در اختیار اکتیویتی دوم قرار خواهد گرفت (و همینطور اگر به همین شیوه اکتیویتی های دیگر صدا زده شوند). حالت دوم اینکه اکتیویتی دوم برای انجام پردازشی فراخوانده شود و پس از انجام پردازش نتیجه محاسبات (یا پردازش) را به اکتیویتی اول برگرداند. در این حالت میبایست از تابع

startActivityForResult()برای فراخوانی اکتیویتی دوم استفاده شود. در این پروژه سه اکتیوتی خواهیم داشت. در اکتیویتی اول دو دکمه قرار دارد که یکی صرفاً اکتیویتی

دیگر را فراخوانی میکند و دیگری اکتیویتی سوم را به منظور پردازش اطلاعات صدا میزند و منتظر دریافت نتیجه می ماند.شکل ( ) طراحی واسط کاربری مربوط به اکتیویتی اول را نشان میدهد.

بدلیل طولانی بودن، قسمتی از کد XML مربوط به این اکتیویتی را در زیر مشاهده میکنید. لطفاً برای دیدن سایر قسمتهای کد برنامه کامل به لوح فشرده مراجعه نمایید.

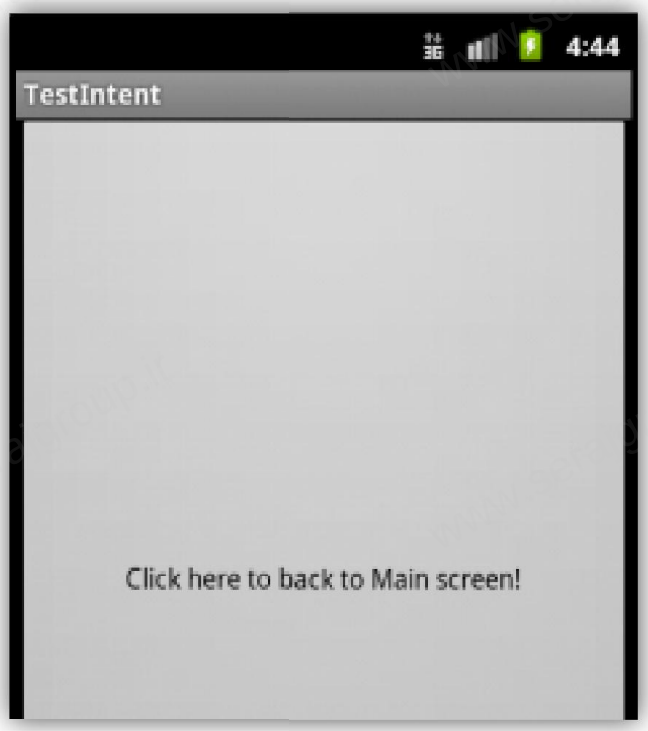
```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >


    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dip" >

        <TextView
            android:id="@+id/tvName1"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentLeft="true"
            android:layout_centerVertical="true"
            android:text="Number 1:"
```

```
            android:paddingRight="10dip" />

        <EditText
            android:id="@+id/etNumber1"
            android:layout_toRightOf="@+id/tvName1"
            android:layout_width="200dip"
            android:layout_height="wrap_content"
            android:layout_alignParentRight="true"
            android:layout_centerHorizontal="true"
            android:inputType="number"
            android:singleLine="true"
            android:imeOptions="actionNext" />

    </RelativeLayout>


    <RelativeLayout
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dip" >

        <TextView
            android:id="@+id/tvName2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_alignParentLeft="true"
            android:layout_centerVertical="true"
            android:text="Number 2:"
            android:paddingRight="10dip" />

        <EditText
            android:id="@+id/etNumber2"
            android:layout_toRightOf="@+id/tvName2"
            android:layout_width="200dip"
            android:layout_height="wrap_content"
            android:layout_alignParentRight="true"
            android:layout_centerHorizontal="true"
            android:inputType="number"
            android:singleLine="true"
            android:imeOptions="actionDone" />

    </RelativeLayout>
```

ادامه دارد...

از آنجاکه سه اکتیویتی داریم و هر اکتیویتی به فایل XML خود برای قرار گرفتن در صفحه نمایش احتیاج دارد، در قسمت res/layout دو فایل دیگر با نامهای next.xmlو sum.xml ساخته ایم. main.xml هم که بصورت پیشفرض زمانیکه برنامه از طریق ویزارد ساخته میشود، ایجاد شده است. کد فایل next.xml بصورت زیر است. در این فایل فقط یک دکمه داریم که با کلیک بر روی آن میخواهیم به صفحه اصلی برنامه برگردیم. وشکل گرافیکی آن را در شکل ( ) مشاهده میکنید.

```
<?xml version="1.0" encoding="utf-8"?>

<Button

xmlns:android="http://schemas.android.com/apk/res/android"
    android:id="@+id/btnBack"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:text="Click here to back to Main screen!" />
```

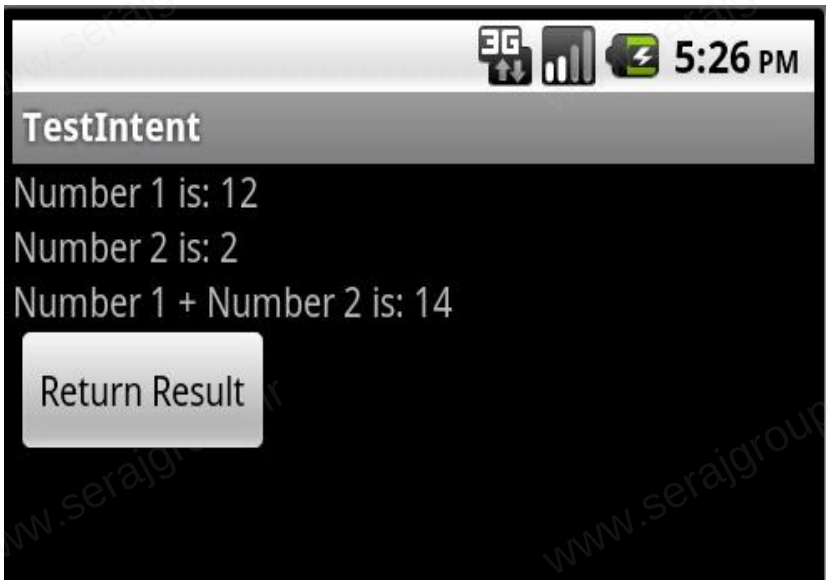


علت اینکه دکمه تمام صفحه را گرفته است این است که مقدار خصوصیت های layout_width و layout_height را «fill parent» قرار داده ایم.

فایل sum.xml شامل سه TextView و یک دکمه است که اعداد وارد شده در EditBoxهای اکتیویتی اول را نشان میدهد، همچنین نتیجه جمع آنها را نیز محاسبه میکند و درنهایت نتیجه را به کلیک بر روی دکمه به اکتیویتی اول بر میگرداند. کد زیر طراحی واسط کاربری را نشان میدهد.

```
<?xml version="1.0" encoding="utf-8"?>

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical" >

    <TextView
        android:id="@+id/tvNumber1"
        android:layout_width="wrap_content"
      android:layout_height="wrap_content" />

    <TextView
        android:id="@+id/tvNumber2"
        android:layout_width="wrap_content"
     android:layout_height="wrap_content" />

    <TextView
        android:id="@+id/tvNumber3"
        android:layout_width="wrap_content"
      android:layout_height="wrap_content" />

    <Button
        android:id="@+id/btnResult"
        android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="Return Result" />

   </LinearLayout>
```

شکل ( ) نمای گرافیکی فایل Sum.xml را نشان میدهد.



خب، در اینجا کار طراحی گرافیکی به پایان رسیده و نوبت به لینک کردنlayout ها به کد برنامه میرسد. با اکتیویتی اصلی شروع میکنیم. کد کامل آن را در زیر مشاهده میکنید.

```
1.  public class TestIntentActivity extends Activity {
2.  EditText etNumber3;
3.  @Override
4.  public void onCreate(Bundle savedInstanceState) {
5.  super.onCreate(savedInstanceState);
6.  setContentView(R.layout.main);

7.  final EditText etNumber1 =
8.  (EditText) findViewById(R.id.etNumber1);

9.  final EditText etNumber2 =
10. (EditText) findViewById(R.id.etNumber2);

11. etNumber3 =
12. (EditText) findViewById(R.id.etNumber3);

13. Button btnNext =
```

```
14.  (Button) findViewById(R.id.btnNext);

15.  btnNext.setOnClickListener(new OnClickListener(){
16.  @Override
17.  public void onClick(View arg0) {
18.  Intent intent = new
19.  Intent(TestIntentActivity.this, NextActivity.class);
20.  startActivity(intent);
21.  }
22.  });
23.  Button btnSum = (Button)
     findViewById(R.id.btnResult);
24.  btnSum.setOnClickListener(new OnClickListener(){
25.  @Override
26.  public void onClick(View arg0) {
27.  Intent intent = new
28.  Intent(TestIntentActivity.this, SumActivity.class);
29.  intent.putExtra("Number1",etNumber1.getText().toString());
30.  intent.putExtra("Number2",etNumber2.getText().toString());
31.  startActivityForResult(intent, 1);
32.  }
33.  });
34.  }

35.  @Override
36.  protected void onActivityResult(int requestCode, int
     resultCode, Intent data)
37.  {
38.  if (resultCode == RESULT_OK && requestCode == 1) {
39.  if (data.hasExtra("Result")){
40.  etNumber3.setText(Integer.toString(data.getExtras().g
     etInt("Result")));
41.  }
42.  }
43.  }
44.  }
```

توضیح کدهای اکتیویتی اصلی برنامه :

در خطوط زیر EditText ها را به کد لینک کرده ایم.

```
final EditText etNumber1 =
(EditText) findViewById(R.id.etNumber1);

final EditText etNumber2 =
```

```
(EditText) findViewById(R.id.etNumber2);

etNumber3 = (EditText) findViewById(R.id.etNumber3);
```

از خطوط ۱۳ تا ۲۲ btnNext را معرفی و آنرا به layout متصل کرده ایم. همچنین گفته ایم که زمانیکه بر روی آن کلیک شد تابع NextActivity را اجرا کند. در ابتدای مطلب اشاره کردیم که اگر اکتیویتی دوم مستقل از اکتیویتی اول قرار باشد که اجرا شود؛ با استفاده از startActivity() اینکار انجام میشود که در خط ۲۰ استفاده شده است.

از خطوط ۲۳ تا ۳۴ دکمه btnSum را معرفی و به layout لینک کرده ایم. همچنین با استفاده از دستور Intent.putExtra() متغییرهای موردنیازمان را به Intent اضافه کرده ایم. ازآنجاکه قرار است اکتیویتی بعدی نتیجه جمع دو عدد را برگرداند، میبایست دو عدد به آن تحویل داده شود که این دو عدد در خطوط ۳۹ و ۴۰ به اینتنت اضافه شده اند. putExtra دو آرگومان دارد، اولی کلید[1] و دومی مقدار[2] نام دارد. به کلید هر نامی که دوست دارید میتوانید نسبت دهید. در گیرنده از این کلید برای دسترسی به مقدار آن استفاده میشود. در خط ۳۱ ملاحظه میکنید که اینبار startActivityForResult() استفاده شده است. این تابع دو آرگومان دارد، اولی همان اینتنت است و دومی کدی

[1] Key
[2] Value

است که به آن اختصاص میدهیم. این کد یک عدد است که خودمان انتخاب میکنیم و نباید تکراری باشد.

در خطوط ۳۵ تا ۴۲ تابع onActivityResult() را داریم. این تابع زمانی فراخوانی میشود که از اکتیویتی دوم به اول برگشته باشیم. در خط ۳۸ گفته ایم اگر اکتیویتی دوم با موفقیت خاتمه پذیرفته است (RESULT_OK) و کد ارسال اطلاعات به اکتیویتی دوم ۱ بوده است، خطوط ۳۹ و ۴۰ را اجرا کن. در اینجا نیازی به چک کردن کد نیست چون فقط یک startActivityForResult() داریم ولی اگر شما در برنامه تان بیشتر از یکبار در جاهای مختلف از این تابع استفاده کردید بنابراین نیاز دارید تا کدهای مختلف تعریف کنید تا زمانیکه به این اکتیویتی برگشتید بدانید از کجا به اینجا آمده اید.

در خط ۴۰ هم نوشته ایم مقداری که در کلید Result وجود دارد را در EditBox قرار دهد. این مقدار در اکتیویتی دوم تعریف شده و مقدارش تعیین میشود. دقیقاً مانند همین کاری که در خطوط ۲۹ و ۳۰ انجام دادیم.

حال نوبت به اکتیویتی دوم (SumAvtivity) میرسد. کد آن را در زیر ملاحظه میکنید.

```
1.  public class SumActivity extends Activity {
2.  @Override
3.  public void onCreate(Bundle savedInstanceState) {
4.  super.onCreate(savedInstanceState);
5.  setContentView(R.layout.sum);
6.  Bundle extras = getIntent().getExtras();
7.  if(extras == null) {
8.  return;
9.  }
```

```
10.  String value1 = extras.getString("Number1");
11.  String value2 = extras.getString("Number2");
12.  TextView tvNumber1 = (TextView)
     findViewById(R.id.tvNumber1);
13.  TextView tvNumber2 = (TextView)
     findViewById(R.id.tvNumber2);
14.  TextView tvNumber3 =(TextView)
     findViewById(R.id.tvNumber3);

15.  tvNumber1.setText("Number 1 is: " + value1);
16.  tvNumber2.setText("Number 2 is: " + value2);
17.  final int i = Integer.parseInt(value1);
18.  final int j = Integer.parseInt(value2);
19.  tvNumber3.setText("Number 1 + Number 2 is: " +
     Integer.toString(i+j));
20.  Button btnResult = (Button)
     findViewById(R.id.btnResult);
21.  btnResult.setOnClickListener(new OnClickListener() {
22.  @Override
23.  public void onClick(View v) {
24.  Intent intent = new Intent();
25.  intent.putExtra("Result", i+j);
26.  setResult(RESULT_OK, intent);
27.  SumActivity.this.finish();
28.  }
29.  });
30.  }
31.  }
```

توضیح کدهای SumAvtivity : در خط ۵ گفته ایم که layout این اکتیویتی را sum.xml قرار دهد.

از آنجاکه این اکتیویتی مستقل از اکتیویتی اول نیست و قرار است محاسبه انجام دهد بنابراین در خطوط ۶ تا ۱۲ چک کرده ایم که آیا اینتنتی که ما را به اینجا آورده است مقدار اولیه همراهش بوده یا خیر؟ اگر نه که به کار این اکتیویتی خاتمه دهد. ولی از

آنجا که ما قبلاً در اکتیویتی اول با استفاده از کلیدهای Number1 و Number2 مقادیری را به این اکتیویتی ارسال کرده ایم، بنابراین در خطوط ۱۰و۱۱ مقادیر این دو کلید را دریافت میکنیم و در متغییری ذخیره میکنیم.

در خطوط ۱۲و۱۳و۱۴ سه TextView ای که در فایل XML معرفی شده را به کد متصل میکنیم.

در خطوط ۱۵ تا ۱۹ به هرکدام ازTextView ها مقداری داده ایم تا ببینیم آیا مقادیری که از اکتیویتی اول به اینجا آمده درست است یا خیر.

در خطوط ۲۰ تا ۲۹ مانند اکتیویتی اول عمل کرده ایم. ابتدا دکمه را به XML لینک کرده ایم و گفته ایم زمانیکه بر روی آن کلیک شد یک اینتنت بساز. نام کلید آنرا Resultقرار بده و مقدار آنرا نتیجه جمع دو عددی که از اکتیویتی اول آمده است قرار بده. سپس از آنجا که نتیجه محاسبه کامل انجام شده آرگومان اول SetResult را RESULT_OK قرار بده و اینتنت را برای اکتیویتی اول بفرست. (البته همانطور که در خط ۲۴ ملاحظه میکنید در آرگومان Intent مشخص نکرده ایم که چه اکتیویتی میبایست اجرا شود چون در خط ۲۷ اکتیویتی دوم را بسته ایم و اتوماتیک وار سیستم عامل ما را به اکتیویتی اول یعنی اکتیویتی که در حال حاضر در بالای پشته قرار دارد بر میگرداند. این فقط بدین منظور نوشته شده که بدانید اینتنت با متغییر برای اکتیویتی بعد ارسال میشود صرف نظر از اینکه اکتیویتی بعد به این متغییر نیاز داشته باشد یا خیر).

نکته مهم اینکه در putExtra فقط چندین نوع متغییر هستند که میتوانید قرار دهید مانند انواعint ، String، boolean **و...** برای اطلاعات بیشتر مطالب سایت اندروید علی الخصوص "Intents and Intent Filters" را مطالعه فرمایید.

اکتیویتی سوم هم که دیگر نیاز به توضیح ندارد چراکه فقط یک دکمه دارد و زمانیکه بر رویش کلیک میشود، آن اکتیویتی بسته میشود. کد آنرا در زیر مشاهده میکنید.

```
1.  public class NextActivity extends Activity {
2.  @Override
3.  public void onCreate(Bundle savedInstanceState) {
4.  super.onCreate(savedInstanceState);
5.  setContentView(R.layout.next);
6.  Button btnBack = (Button) findViewById(R.id.btnBack);
7.  btnBack.setOnClickListener(new OnClickListener() {
8.  @Override
9.  public void onClick(View v) {
10. NextActivity.this.finish();
11. }
12. });
13. }
14. }
```

اگر با خواندن کدهای موجود در پروژه بالا احیاناً مطالب را متوجه نشدید، اصلاً نگران نباشید. میتوانید جلوتر بروید و برنامه های دیگر را یاد بگیرید و بعد این پروژه را بررسی کنید.خصوصاً اینکه اگر خود برنامه را که در داخل لوح فشرده است را تست کنید و خروجی را ببینید بهتر است. و نیز در محیط توسعه نرم افزار اکلیپس، کد ها دارای تورفتگی هستند و خود این موضوع باعث میشود که شما کدها را بهتر ببینید و بهتر درک کنید.و همواره از کار با محیط عملی غافل نشوید.

یادداشت :

فصل سوم : طراحی واسط کاربری با XML

آن چه را که در این فصل فرا میگیرید:

منابع برنامه(Application Resources)

برنامه های اندروید صرفاً از کد تشکیل نشده اند ، بلکه از ترکیب چیزهای مختلفی ساخته میشوند. برنامه برای ساخته شدن نیاز به منابعی دارد که از کد منبع[1] مجزا هستند، مانند تصاویر، فایلهای صوتی و هرچیزی که در برنامه مورد توجه کاربر قرار میگیرد. بعنوان مثال، میبایست انیمیشن ها، منوها، استایل ها، رنگ ها و طرح گرافیکی اکتیویتی مورد استفاده کاربر با فایل هایXML ، تعریف شوند. همچنین استفاده از این منابع مختلف در برنامه، از این جهت که نیاز به تغییر در سورس کد برنامه را ندارد، قابلیت بروزرسانی برنامه را بسیار ساده تر میکند. همچنین شما را قادر میسازد تا برنامه خود را برای ابزارهای دیگر با صفحات متفاوت و زبان های متفاوت، بهینه سازی کنید.

منابع، المانهای خارجی هستند که ما در برنامه خود بکار میگیریم. آنها در فولدر "res" ذخیره میشوند و میتوانند فایلهای تصویری، صوتی مانند عکس، فیلم، انیمیشن و موزیک باشند. همچنین فایلهای XML که شامل طراحی های پوسته، واسط کاربری و تم ها هستند، در این پوشه ذخیره میشوند. وقتی عملیات کامپایل انجام میشود، آنها بطور کامل از طریق کدهای جاوا قابل دسترسی هستند.

[1] Source Code

به هر منبعی که به برنامه تان اضافه میکنید، یک عدد منحصربفرد توسط SDK بعنوان کد هویتی[1] تعلق میگیرد که شما میتوانید از آن کد در سورس برنامه یا از منابع دیگری که در فایل های XML تعریف شده اند، بعنوان مرجعی به منبع اصلی استفاده کنید. بعنوان مثال اگر برنامه شما یک فایل تصویری با نام logo.png دارد که در مسیر /res/drawable ذخیره شده ، SDK یک کد هویتی با نام R.drawable.logo به منبع تصویری اختصاص میدهد که شما میتوانید از این برای رفرنس در کد برنامه تان استفاده کنید و از آن در واسط گرافیکی برنامه استفاده کنید.

پس همانطور که گفته شد یکی از مهمترین ویژگی های اندروید، جدا بودن منابع از سورس کد برنامه است، قابلیتی است که به شما مجوز استفاده از منابع در طرح بندی موبایل های مختلف را میدهد. بعنوان مثال، با تعریف یک رشته در فایلXML ، میتوانید آنرا به زبانهای مختلف ترجمه کرده و در فایلهای مختلف ذخیره کنید. سپس، براساس زبانی که در دایرکتوری منابع برنامه تعریف کرده اید، مثلاً زبان فرانسه-/res/values FRو زبان انتخاب شده توسط کاربر، سیستم اندروید زبان مناسب را در برنامه شما نشان خواهد داد.

[1] Identification Code

اندروید از توصیف کننده های [1] متنوعی برای منابع جایگزین [2] شما پشتیبانی میکند. توصیف کننده، یک رشته کوچک است که شما آنرا در نام دایرکتوری منابع برنامه وارد میکنید تا براساس نیاز از آنها استفاده شود. بعنوان مثالی دیگر، شما اغلب مجبور میشوید، بسته به جهت (orientation) موبایل و اندازه آن، حالت های گرافیکی مختلفی برای اکتیویتی بسازید. بعنوان مثال، وقتی جهت موبایل در حالت عمودی[3] است، ممکن است بخواهید دکمه ها بحالت عمودی نشان داده شوند ولی زمانی که صفحه یا جهت موبایل در حالت افقی[4] قرار گرفته است، دکمه ها بصورت افقی قرار گرفته باشند. بنابراین، سیستم بصورت خودکار طرح گرافیکی مناسب را بسته به جهت موبایل انتخاب میکند.

View ها و Layout ها

Viewیک عضو مستقل (single object) در واسط کاربری است. View در داخل Layout قرار میگیرد و از ترکیب View ها ، ساده ترین تا پیچیده ترین واسط های گرافیکی و غیر گرافیکی کاربری ساخته میشود. بعنوان مثال، برچسب (Label) یک نوع View است که در اندروید به آن TextView میگویند.

---

[1] qualifiers
[2] alternative resources
[3] portrait orientation
[4] landscape orientation

شما میتوانید با ساخت یک زیر کلاس از View ویجت مورد نیاز خودتان را بسازید و در اکتیویتی از آن استفاده کنید.اندروید بصورت پیش فرض چندین و چند view را آماده کرده است تا در طراحی واسط کاربری مورد استفاده قرار گیرد. کلاس Viewیک کلاس پایه[1] برای ویجت ها است و همه ی ویجت ها از کلاس View ارث بری دارند. ویجت ها[2] ، کنترل هایی هستند که شکل گرافیکی دارند و به اندازه ی یک مستطیل کوچک صفحه نمایش را در اختیار دارند و با کاربر در تعامل هستند. برای مثال Button,EditText,TextView و ... نمونه هایی از ویجت ها هستند.برای کار کردن با هر ویجت باید با استفاده از دستور import، کلاسهای ویجت مورد نظر خود را به برنامه اضافه کرد. مثال :

```
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Button;
```

چیدمان ها[3] ، شامل مجموعه ای از view ها هستند که از کلاس ViewGroup مشتق[4] میشوند و نحوه قرار گیری view ها را در صفحه مشخص میکنند. Layout ها حتی میتوانند به صورت تو در تو بکار گرفته شوند. هر زمانیکه یک فایل XML برای واسط

---

[1] Base Class
[2] Widgets
[3] Layouts
[4] Derived

کاربری ایجاد میکنیم، نیاز به Layout داریم تا طراحی ما را در خودش نگه دارد. Layout ها بر اساس نحوه نگهداری المان های طراحی ، انوع مختلفی دارند :

- اگر view ها را بصورت خطی، پشت سر هم قرار دهد ، به آن Linear Layout میگویند.
- اگر View ها را به صورت جدولی نگه دارد ، به آن Grid Layout میگویند(مانند صفحه شطرنج به صورت مشبک View ها را در درون خود قرار میدهد) .
- اگر View ها را نسبت به هم کنار هم قرار دهد ، به آن Relative Layout میگویند.

در واقع میتوان گفت که کلمه اکتیویتی معادل کلمه فرم[1] در واژگان مربوط به برنامه های دسکتاپ[2] است وبرای این که اکتیویتی دارای واسط کاربری باشد باید به آن یک View (معمولاً یک چیدمان[3]) تخصیص[4] دهیم.

---

[1] Form
[2] Desktop Application
[3] Layout
[4] Assign

ابزار طراحی واسط کاربری

در این کتاب برای طراحی واسط کاربری از محیط اکلیپس استفاده میکنیم.اگر شما با این محیط راحت نیستید میتوانید واسط کاربری (فایل xml) خود را با استفاده از ابزارهای زیر طراحی کنید:

- Eclipse ADT UI Designer
- Droid Draw
- Asset Studio

به راحتی میتوانید با جستجوی این برنامه ها در اینترنت و دانلود و نصب آنها واسط کاربری خود را با این نرم افزار ها طراحی کنید و از امکانات این برنامه ها هم به عنوان ابزار کمکی استفاده نمایید.

متد طراحی واسط کاربری

واسط کاربری برنامه خود را میتوانید به دو شیوه[۱] رویه ای[۲] و توصیفی[۳] طراحی کنید. در متد رویه ای شما میتوانید کد واسط کاربری برنامه خود را در قسمت کدهای برنامه (کدهای جاوا) بنویسید

---

[1] Method
[2] Procedural
[3] Declarative

در متد توصیفی شما میتوانید با استفاده از کدهای XML واسط کاربری برنامه را ایجاد کنید.از این جهت به آن توصیفی میگویند که XML از دسته زبان های توصیفی است.زبان XML شبیه به زبان توصیفی HTML[1] است.در زبان های توصیفی تعیین میکنید که چه چیزی را میخواهید در صفحه ببینید نه این که چه کاری را میخواهید انجام بدهید.

از کدام متد استفاده کنید بهتر است ؟ هر دوی این متدها صحیح هستند ولی توصیه گوگل این است که از متد توصیفی(کدهای XML) استفاده شود. مزیت های اسفاده از این روش این است که کدهای XML نسبت به کدهای جاوا کوتاه ترند و درک و فهم آن از کدهای جاوا ساده تر است و در ورژهای بعدی هم امتحال تغییر این نوع کد کمتر است.از طرف دیگر اگر ما طرحهای گرافیکی خود را بر اساس XML تولید کنیم و در قسمت سورس کد برنامه صرفاً به عملکرد (Functionality) برنامه بپردازیم ؛ بدین سبب از پیچیدگی برنامه کاسته ایم.

اگر کدهای واسط کاربری را با متد توصیفی(به صورت XML) بنویسید در قسمت کدهای جاوا صرفاً به عملکرد (functionality) برنامه میپردازید و بدین سبب پیچیدگی برنامه کاهش میابد.

[1] Hyper Text Markup Language

اندروید برای راحت تر شدن کار برنامه نویسان سورس کد برنامه را از محیط طراحی جدا کرده است.اگر چه شما ، ولی بهتر و راحتر است که واسط کاربری نرم افزار را به صورت جداگانه در فایل موجود در پوشه Res/Layout با استفاده از کدهای XML طراحی کنید.

معمول ترین راه به منظور طراحی چیدمان و قرار دادن view ها در آن، استفاده از XML است که بعنوان منبع [1] در برنامه قرار خواهد گرفت. شما همچنین میتوانید در کد برنامه هم یک view بسازید و با قرار دادن آن در ViewGroup از آن استفاده کنید. سپس ViewGroup را با استفاده از setContentView() بعنوان واسط کاربریِ اکتیویتی معرفی کنید.همانطور که قبلاً هم گفتیم این متد معمولاً در متد onCreate() فراخوانی میشود.

اتفاقات(Events)

Eventها اتفاقاتی هستند مانند کلیک کردن یا لمس کردن، که برای دریافت این اتفاقات متد Listener آن اتفاق باید تعریف شود .Event اتفاقی است که از بیرون رخ میدهد؛ چیزی است که کنترل آن دست ما نیست؛ یا حتی چیزیست که ما نمیدانیم کی

[1] resource

اتفاق میافتد. بدین منظور کنترلر های اتفاق (Events controllers) مورد استفاده قرار میگیرند. این امکان وجود دارد که برای هر اتفاقی که رخ میدهد به سیستم بگوییم که چه کاری انجام دهد. معمول ترین اتفاق دریافت لمس صفحه نمایشگر است که برای دریافت آن Listener مورد نیاز باید تنظیم شود.

از اینجا به بعد آموزش برنامه نویسی در محیط اکلیپس به پایان میرسد وشما با محیط موسینک آشنا میشوید.

فصل چهارم : شروع کار با موسینک

موسینک چیست ؟

موسینک (MoSync) یک کیت توسعه نرم افزار(SDK[1]) است. این SDK برای نصب و اجرا بر روی سیستم عامل های ویندوز و اپل مکینتاش ساخته شده است و تمامی ویژگی های IDE معروف Eclipse را دارد.

اما خود SDK چیه و به چه منظوری ساخته شده؟

SDK یک نرم افزار سیستمی است که برنامه نویسان از آن برای ساخت پروژه هایشان استفاده می کنند. یک SDK شامل مجموعه ای از ابزارهای لازم برای برنامه نویسی است تا برنامه نویس به کمک آن بتواند برای یک پلتفرم و سیستم عامل خاص برنامه بنویسد. هر SDK از چند مؤلفه ساخته شده است و معمولاً شامل ابزارهای زیر است.

Editor,Compiler,Linker,Libraries,Emulator,...

MoSync یک SDK متن باز[2] است که شما را قادر میسازد تا با یک کد پایه برای پلتفرم های مهم و رایج موبایل برنامه بنویسید.یعنی با یک کد برای چند پلتفرم مختلف و ناسازگار برنامه بنویسید. به این ویژگی Cross-Platform بودن میگویند.موسینک توسط یک شرکت نرم افزاری در استکهلم سوئد ساخته شده است.در موسینک میتوان با استفاده از زبان هایی مثل C++,Java Script(jQuery Mobile),HTML5 برنامه

[1] Software Development Kit
[2] Open Source

هایی ایجاد کنید که بر روی پلتفرم های مختلف قابل اجراست.این یعنی جادوگری برای موبایل![1]

پلتفرم های مهم و رایجی که با موسینک میتوان برای آن برنامه ساخت ، اندروید ، iPhone و ویندوز موبایل و ... است.

برنامه هایی که با MoSync ساخته میشوند با برنامه هایی که با SDK های ویژه و مخصوص پلتفرم های خاص نوشته شده اند تفاوتی ندارد.یعنی اگر شما برنامه ای را با MoSync SDK بسازید تمیز دادنی با برنامه ای نیست که با Android SDK یا iPhone SDK و یا Windows phone SDK نوشته شده است. پس خیالتان از این بابت راحت باشد و با این SDK هم میتوانید برنامه های استاندارد برای پلتفرم های رایج مذکور بسازید بدون اینکه برنامه نقصی داشته باشد.

این SDK قابلیت کار کردن با ویژگی های سخت افزاری جدید ، موبایل های امروزی را دارد.درواقع API های لازم برای دسترسی به ویژگی هایی مثل گرافیکی ، ارتباطی ، منطقه ای[2] ، اطلاعات مخاطبان و سنسورها و ... را دارد. از واسط کاربری موبایل مقصد(که به آن NativeUI میگویند) پشتیبانی میکند و توانایی کار با OpenGL را دارد.

---

[1] Mobile Sorcery
[2] Location(GPS)

در گذشته ابزارهای برنامه نویسی موبایل جالب نبودند.برای توسعه نرم افزار های موبایل شما باید با ابزارها و زبان های ناآشنا و عجیب غریب کار میکردید ولی شما برای توسعه نرم افزار توسط این SDK میتوانید از زبان های رایجی مثل HTML5,JavaScript,C/C++ استفاده کنید. شما میتوانید کدهای خود را با HTML5 و JavaScript بنویسید ویا از C,C++ استفاده کنید و یا از هر دوی این روش ها استفاده نمایید!

کدهای خود را میتوانید در یک IDE بنویسید که شباهت زیادی به Eclipse دارد و تمام ویژگی های اکلیپس[1] را دارد.کامپایلر آن هم شبیه به GCC 3.4.6 است و تنها عقب بندی[2] آن تغییر کرده و مخصوص MoSync شده است تا بتواند برنامه را برای اجرا بر روی پلتفرم های موبایل ترجمه و تبدیل کند. عقب بندی کامپایلر یک کد میانی[3] تولید میکند که به آن MoSync Intermediate Language میگویند. و بعد این کد میانی از طریق یک ابزار به اسم Pipe-tool به کدهای محلی قابل اجرا بر روی پلتفرم های خاص تبدیل میشود(هر کامپایلر از دو بخش Front-End و Back-End تشکیل شده است. که توضیح دلیل این جداسازی از بحث ما خارج است ولی میتوانید به کتاب های کامپایلر مراجعه نمایید و با دلیل این جداسازی آشنا شوید).

---

[1] Eclipse-Based
[2] Backend
[3] Intermediate Code

GCC[1] هم کامپایلری است که برای ترجمه زبان C ساخته شد.که به آن egcs هم میگویند. Gcc مخفف GNU C/C++ Compiler است. که ما را به یاد یونیکس می اندازد. پس اصل و اساس این SDK زبان C استاندارد است و OpenSource هم میباشد.

Pipe-tool موتور تبدیل کد کیت توسعه نرم افزار MoSync است[2]. این موتور تبدیل بسیار سریع کار میکند.به طوری که در حدود یک ثانیه شش گذر(pass) بر روی ۲۵۰۰۰ خط کد را انجام میدهد. و با سرعت کدهای برنامه را در هر گذر میخواند و کد را بهینه و برای اجرا بر روی پلتفرم خاص آماده میکند.Pipe-tool به صورت خودکار توسط IDE موسینک اجرا میشود و شما برای تبدیل کد نیاز به کار خاصی ندارید و کافی است که گزینه build را از منوی IDE انتخاب نمایید تا این ابزار به صورت خودکار تبدیلات را انجام دهد. در فارسی به کلمه Pipe خط لوله میگویند.شاید دلیل این نام گذاری این است که کدی که ما نوشتیم از یک خط لوله وارد میشود و این خط لوله در انتها چندین انشعاب دارد و از هر انشعاب یک کد نهایی خاص برای یک پلتفرم خاص تولید میشود.

موسینک پلتفرم های مختلف و رایج زیر را پشتیبانی میکند.

---

[1] برای اطلاعات بیشتر به http://gcc.gnu.org/onlinedocs مراجعه شود.

[2] MoSync's code transformation engin

iOS, Android, Windows Phone, Windows Mobile, Symbian, JavaME

برای تست برنامه میتوان از شبیه ساز[1] موسینک که نام آن MoRE[2] است استفاده کرد . این شبیه ساز مانند یک موبایل واقعی عمل میکند و توانایی اجرای بایت کدهای موسینک[3] را دارد. سعی شده که این شبیه ساز به گونه ای طراحی شود که حداقل باگ را داشته باشد تا سرعت تست برنامه ها بالا رود. MoRE دارای بلوتوث مجازی است و ۱۶ مگابایت حافظه دارد و از قلم طراحی[4] هم پشتیبانی میکند. شما میتوانید اندازه صفحه نمایش را به صورت دلخواه تنظیم کنید.و یا برای تنظیم ویژگی های واقعی به او بگویید که تنظیمات و ویژگی هایش شبیه به نوکیا N95 یا سونی اریکسون K700i شود. این انعطاف پذیری کار شما را راحت کرده است.

موسینک دارای یک پایگاه داده از پروفایل ، پلتفرم های مختلف موبایل است. این پایگاه داده شامل اطلاعاتی در مورد صدها دستگاه موبایل است. و به آن Device Profile Database

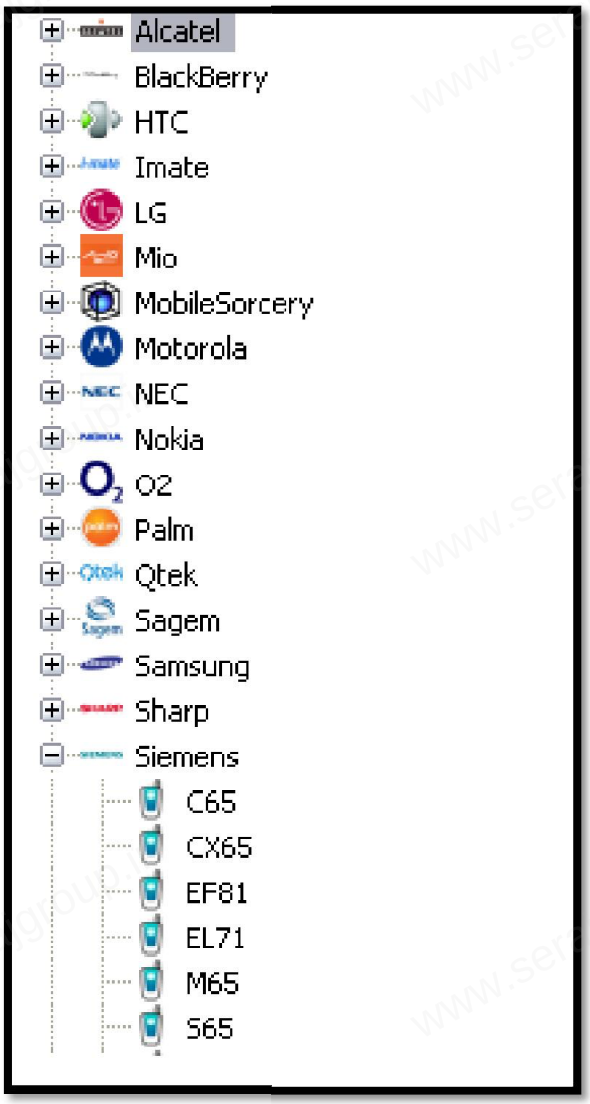
[1] Emulator
[2] MoSync Runtime Environment
[3] MoSync bytecode
[4] stylus

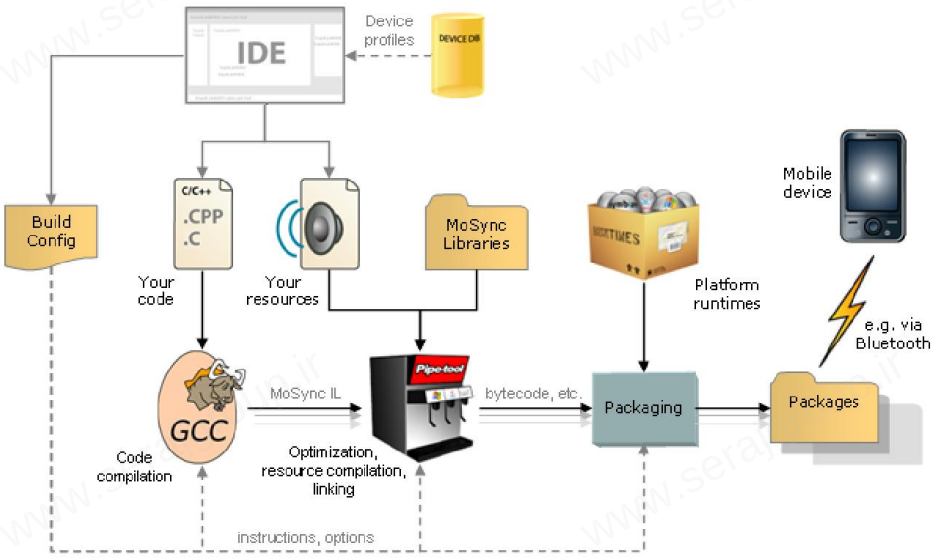
میگویند. در این پایگاه داده برای هر شرکت سازنده[1] یک پوشه[2] وجود دارد و نیز برای هر دستگاه موبایلی که شرکت سازنده تولید کرده ، یک زیر پوشه وجود دارد .

هر دستگاه موبایل یک پروفایل دارد و در این پروفایل ویژگی هایی مثل نام شرکت سازنده، نام دستگاه، اندازه صفحه نمایش و اندازه حافظه و باگ های احتمالی وAPI هایی که در این دستگاه قابل استفاده است(مانند BlueTooth,GPS) و ... ذخیره میشود. اطلاعات مذکور در قالب فایل های سرآمد[3] (h file.) ذخیره میشوند.در شکل( )ساختار درختی مروبط به این پایگاه داده را می بینید.

موسینک دارای یک runtime (اینجا منظور این نیست که برنامه در وضعیت اجرا است بلکه منظور این است که دارای نرم افزاری به نام Runtime است) . این نرم افزار از کتابخانه ها و برنامه هایی تشکیل شده است که قابلیت اجرایی شدن به برنامه های موسینک را میدهد.اما سؤال اینجاست که این نرم افزار چگونه با کتابخانه ها و برنامه های خود باعث میشود تا برنامه نوشته شده با موسینک بر روی دستگاه های مختلف با پلتفرم های مختلف اجرا شود؟

[1] Vendor
[2] Directory
[3] Header file


Alcatel
BlackBerry
HTC
Imate
LG
Mio
MobileSorcery
Motorola
NEC
Nokia
O2
Palm
Qtek
Sagem
Samsung
Sharp
Siemens
C65
CX65
EF81
EL71
M65
S65



همانطور که اشاره شد موسینک یک پروژه متن باز است و شما میتوانید با گروه سازنده آن برای توسعه و بهینه سازی همکاری کنید. این خود یک مزیت مهم این نرم افزار است.

نصب و راه اندازی MoSync بر روی ویندوز

کیت توسعه نرم افزار موسینک را شما میتوانید بر روی ویندوز و اپل مکینتاش نصب نمایید.اما از آنجا که اغلب کاربران ایرانی با سیستم عامل های شرکت مایکروسافت کار میکنند در اینجا به نصب موسینک بر روی ویندوز میپردازیم.برای دانلود این SDK میتوانید به سایت http://www.mosync.com/download مراجعه نمایید. در

زمان نوشتن این کتاب ، آخرین نسخه آن ۲.۷ است. که آن را برای شما در لوح فشرده ای که همراه کتاب است، قرار داده ام.

نکته مهم : با SDK مخصوص ویندوز[1] شما میتوانید بسته های نرم افزاری[2] بسازید که بر روی جاوا ME ، سیمبیان ، اندروید ، ویندوز موبایل و دیگر گوشی های هوشمند[3] و کامپیوتر های جیبی[4] اجرا میشود. اما اگر میخواهیدکه برنامه شما بر روی دستگاه هایی که سیستم عامل iOS دارند، اجرا شود(دستگاه هایی مثل iPhone,iPad,iPod) بایستی کدی که در این SDK نوشته اید به SDK مخصوص اپل مکینتاش منتقل کنید. البته توصیه میشود که اگر شما با سیستم عامل مک (Mac) کار میکنید بهتر است از همان ابتدا SDK مخصوص به این سیستم عامل ، که به آن **MoSync SDK for OS X** میگویند، را نصب نمایید.

نیازمندی های سخت افزاری و نرم افزاری برای موسینک

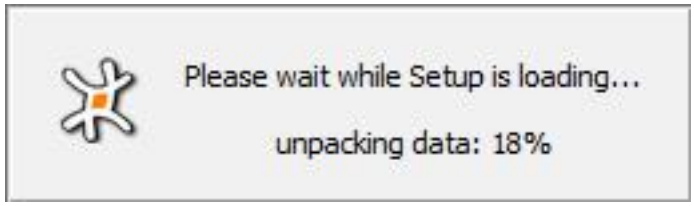
موسینک طراحی شده تا بر روی هر کامپیوتری که نرم افزارهای زیر بر روی آن نصب شده است ، اجرا شود :

---

[1] MoSync SDK for Windows
[2] Application Packages
[3] Smart Phone
[4] Pocket PC

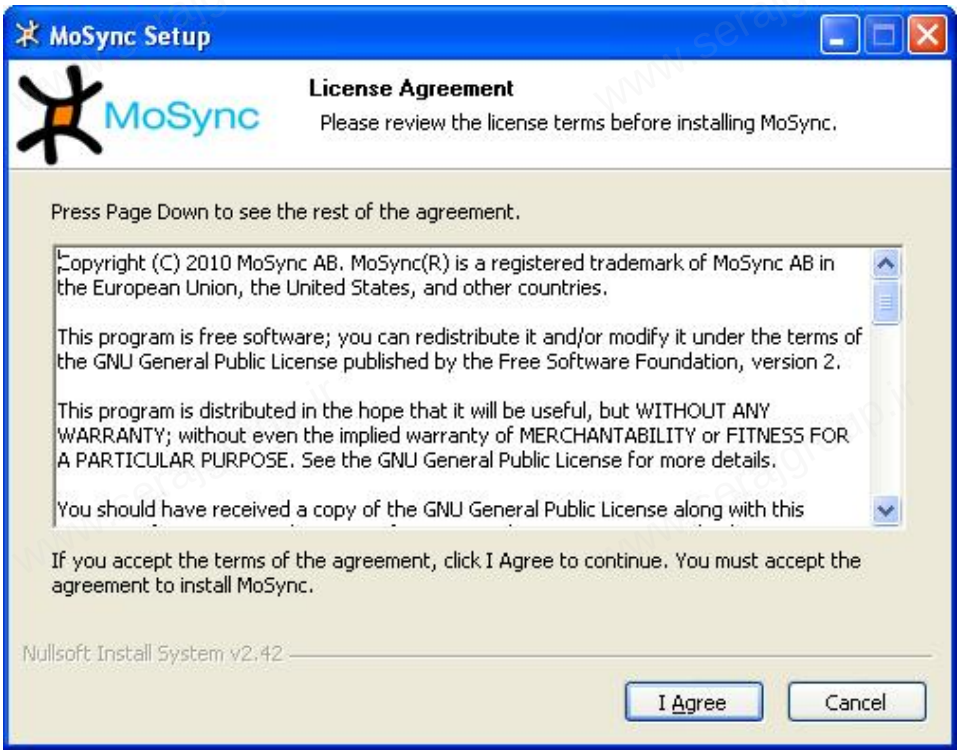
- سیستم عامل ویستا یا XP
- Java SE Runtime Environment (JRE) 6 (آن را از شرکت Oracle/Sun دانلود و نصب نمایید)
- حداقل ۳۰۰ مگابایت فضای خالی از هارد دیسک

بعد از این که بسته نصب موسینک را دانلود کردید. بر روی فایل exe. آن دوبار کلیک کنید. شکل( )



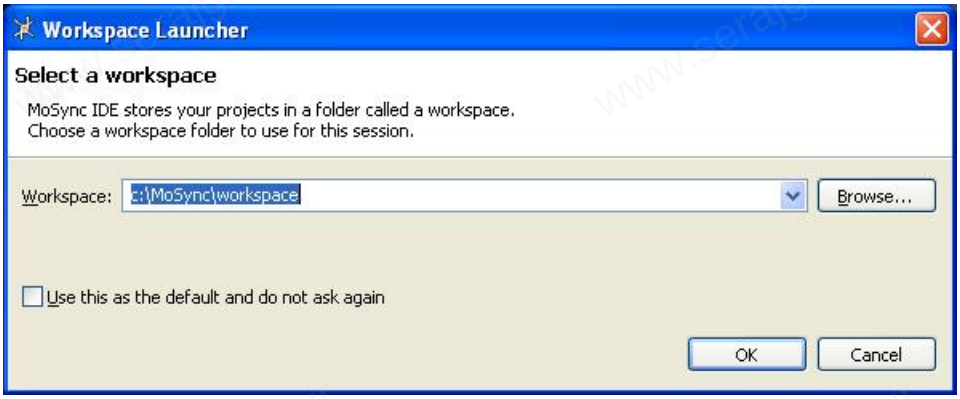
بعد از خواندن اطلاعات فشرده شده و Unpacking شدن آنها صفحه شکل( ) برای شما نمایش داده میشود.

سپس این صفحه محو میشود(Fade) .و بعد صفحه مجوز(License) برای شما نمایش داده میشود.شکل( )



سپس مانند سایر نصب ها کامپوننت های لازم برای نصب را انتخاب میکنید و بعد محل نصب را مشخص میکنید. ودر نهایت برنامه شما نصب میشود.(مراحل نصب پیچیدگی خاصی ندارد و همانند سایر نصب های کلیشه ای است ، به همین خاطر به ادامه آن اشاره نشد).

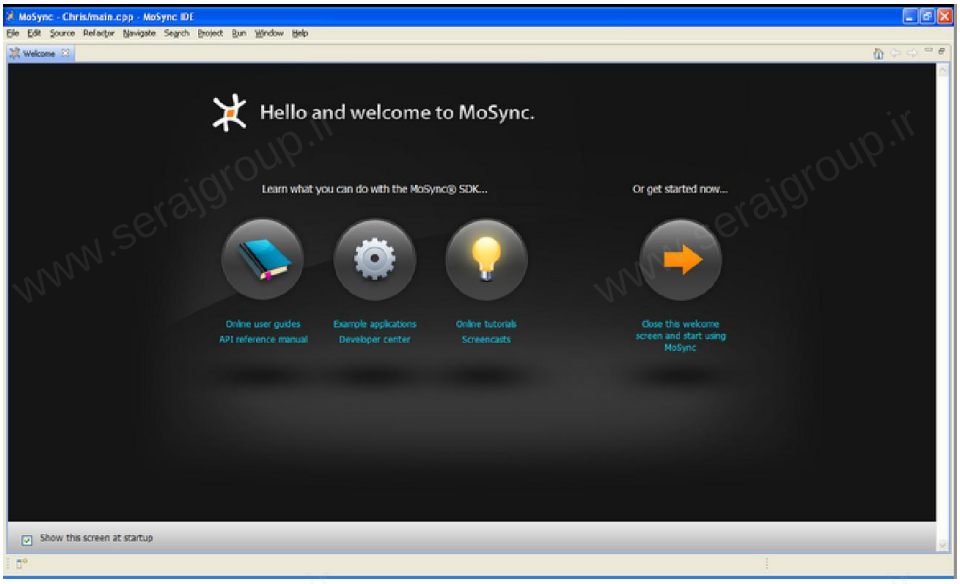
بعد از نصب برنامه موسینک را اجرا کنید. اولین کادر محاوره ای که برای شما نشان میدهد شکل( ) است. و از شما یک آدرس میخواهد تا برنامه هایی که مینویسید را در آنجا ذخیره کند.



سپس اگر شما به اینترنت وصل نباشید به شما پیغام اطلاعاتی را میدهد که شما به اینترنت متصل نیستید.ولی اگر به اینترنت متصل باشید صفحه ای باز میشود که در آن میتوانید به صورت کاملاً رایگان نرم افزار خود را ثبت نمایید.مزایایی که ثبت نرم افزار دارد این است که شما میتوانید با اجرای نرم افزار به صورت خودکار پروفایل های دستگاه های موبایل را بروز رسانی نمایید و نیز دسترسی کامل به وب سایت نرم افزار داشته باشید.مثلاً میتوانید در فروم های مربوط به توسعه دهندگان وارد شوید(www.mosync.com/forum).

راه دیگر برای رجیستر کردن برنامه این است که از منوی Help گزینه Register را انتخاب کنید. سپس یک نام کاربری و آدرس ایمیل وارد کنید (آدرس ایمیل باید واقعی باشد چونکه یک نامه تکمیل ثبت نام ایمیل شما ارسال میشود).

اگر هم قبلاً بواسط سایت ثبت نام کرده اید میتوانید بر روی Already registered? Click Here ، کلیک کنید. بعد از ثبت نرم افزار صفحه خوش آمد گویی را مشاهده میکنید.شکل( ).



در این صفحه چندین لینک مفید وجو دارد که در زیر به آن ها اشاره شده است :

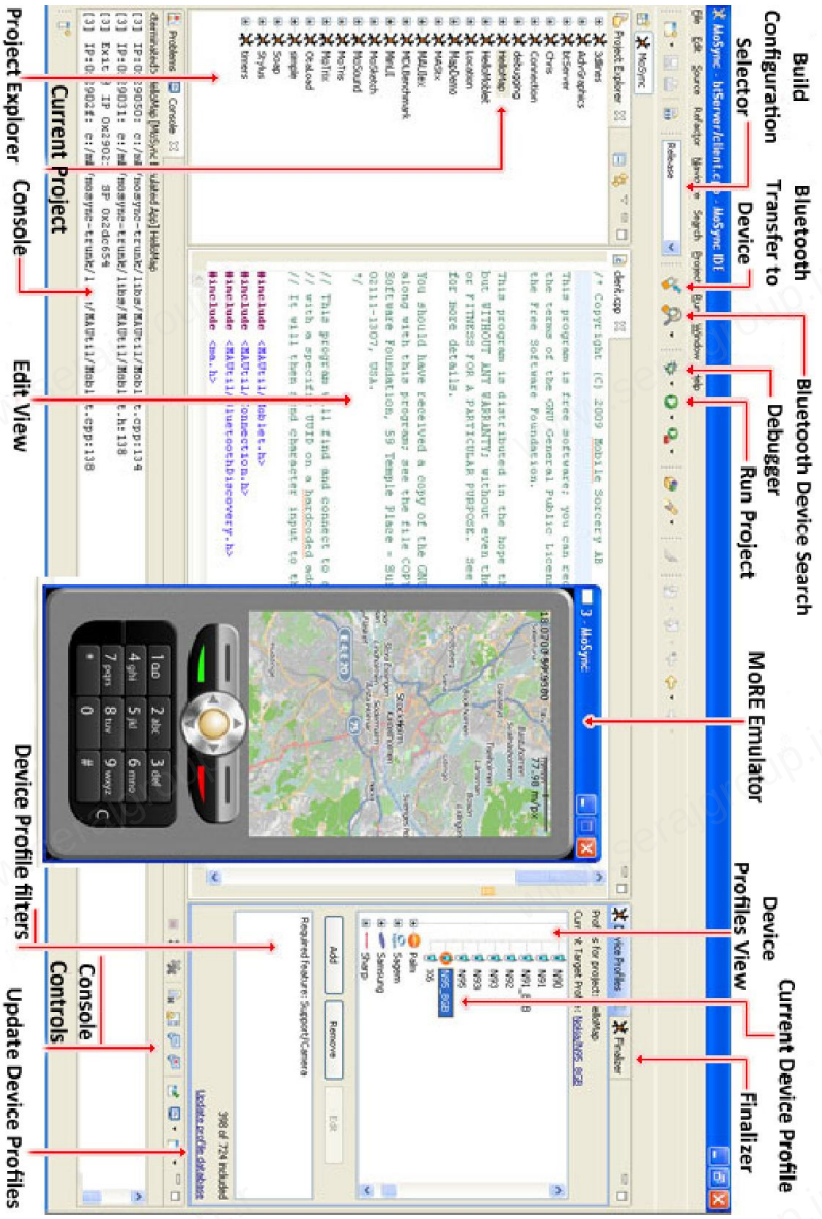
- Online user guides : برای متصل شدن به تمام مستنداتی که بر روی سایت www.mosync.com قرار گرفته شده است . این مستندات به شما کمک میکند تا موسینک را بهتر بشناسید.

- Example applications : با کلیک بر روی این لینک تمام مثال ها(پروژه ها) ای که توسط سازندگان نرم افزار موسینک ایجاد شده اند به پنل Project Explorer اضافه میشود. (این پروژه ها در شاخه MoSync/examples قرار دارند)
- Online tutorials : تمامی آموزش های گام به گام برای برنامه نویسی با این نرم افزار قرار داده شده است.
- API reference manual : مرجعی از API هایی که برای این ورژن از MoSync SDK ساخته شده است.
- Developer center : برای اتصال به فروم برنامه نویسان و خواندن آخرین پست ها.
- Screencasts : آموزش های گام به گام تصویری که به صورت ویدئو هستند و در مورد نحوه استفاده و بکارگیری SDK و IDE هستند.

برای اینکه در اجرا های بعدی نرم افزار پنجره Welcome Screen نمایش داده نشود میتوانید تیک گزینه Show this screen at startup را بردارید. برای اجرا های بعدی این پنجره دیگر نمایش داده نمیشود و برای اجرای دستی آن میتوانید از منوی Help گزینه Welcome را انتخاب نمایید.

شرح محیط کاری موسینک

در شکل( ) IDE موسینک را مشاهده می کنید که بسیار به اکلیپس شبیه است.


Build Configuration Selector
Bluetooth Transfer to Device
Bluetooth Device Search
Debugger
Run Project
MoRE Emulator
Device Profiles View
Current Device Profile
Finalizer
Project Explorer
Console
Current Project
Edit View
Device Profile filters
Console Controls
Update Device Profiles

قسمت های اصلی IDE که در شکل مشخص شده اند به شرح زیر است :

- **Project Explorer View** : این پنل پروژه هایی که شما خود ایجاد کرده اید و یا پروژه هایی که به برنامه وارد کرده اید(import) را نمایش میدهد. و از طریق آن میتوانید فایل های برنامه را مدیریت کنید.در واقع File Manager برنامه شما است.

- **Editor View** : در این بخش میتوانید کدها را ویرایش کنید. این قسمت همانند اکلیپس به صورت خودکار برای متمایز شدن خطوط برنامه کدها را پر رنگ[1] میکند و نیز تورفتگی[2] ها را برای خوانایی کد ایجاد میکند.با نوشتن یک شناسه[3] (مانند یک شی یا متغیر)و گرفتن کلید ترکیبی Ctrl+Spacebar برای شما یک لیست از توابع و ویژگی ها نمایش داده میشود تا راحت تر کد بنویسید.

- **Device Profiles View :** در این لیست دستگاه هایی را مشاهده میکنید که از طریق پروژه جاری(پروژه ای که بر روی آن کار میکنید) میتوانید برای آنها کد اجرایی بنویسید.اگر میخواهید خروجی برنامه را برای یک موبایل خاص مشاهده کنید باید از این لیست موبایل خود را

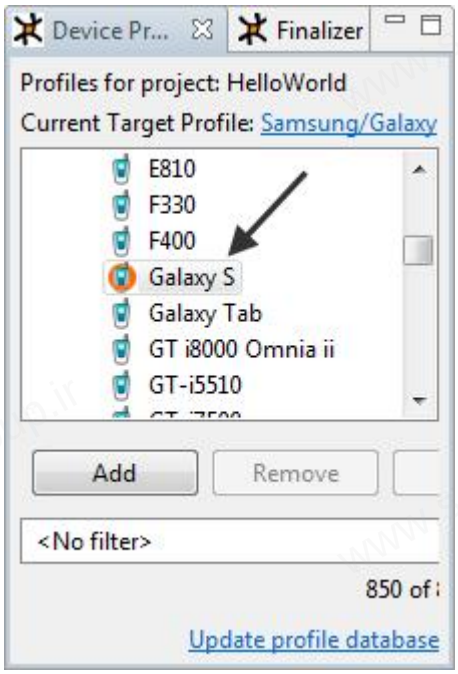
---

[1] highlighting
[2] indenting
[3] identifier

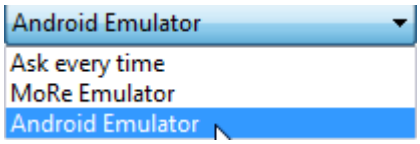
پیدا کنید و بر روی نام آن دوبار کلیک کنید.در این صورت دور نام موبایل شما یک کادر رنگی میکشد و این بدین معنی است که اگر شما برنامه را اجرا کنید خروجی را بر اساس پروفایل موبایل انتخاب شده نمایش داده میشود.

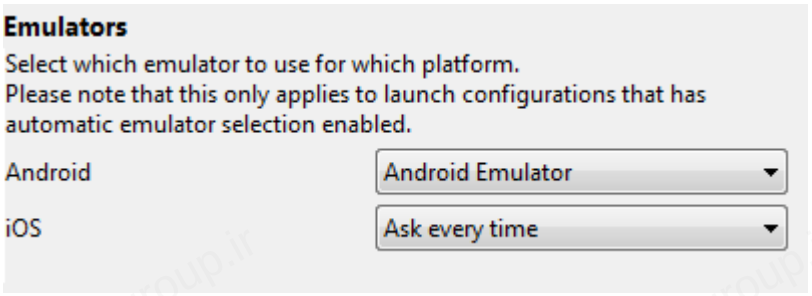
- **Finalizer View** :
- **Problems View :**
- **Console View** :
- **MoRE Emulator :**

اگر میخواهید که خروجی برنامه را از طریق شبیه ساز اندروید مشاهده نمایید میتوانید همانند شکل ( )لیست دستگاه اندرویدی خود را انتخاب نمایید(مثلاً Samsung/Galaxy S) . و در این صورت برنامه به شما پیغام شکل ( ) را میدهد.


Device Pr...
Finalizer
Profiles for project: HelloWorld
Current Target Profile: Samsung/Galaxy
E810
F330
F400
Galaxy S
Galaxy Tab
GT i8000 Omnia ii
GT-i5510
Add
Remove
<No filter>
850 of
Update profile database

البته شبیه ساز مخصوص اندروید، برای تست برنامه های ویژه اندروید ، بهتر از MoRE کار میکند. به همین خاطر توصیه میشود که اگر میخواهید از طریق موسینک برای سیستم عامل اندروید برنامه بنویسید میتوانید SDK آن را از طریق اینترنت تهیه کنید(این SDK توسط گوگل عرضه میشود). و از منوی Window گزینه Preferences را انتخاب کنید.سپس به قسمت MoSync Tools بروید و از زیر بخش Android همانند شکل( ) آدرس SDK خود را که دانلود کرده اید وارد نمایید. سپس به قسمت Emulator بروید و شبیه ساز مورد نظر خود را انتخاب کنید. همانطور که در شکل( ) میبینید بر روی سیستم من شبیه ساز iOS وجود ندارد (به همین خاطر جلوی آن عبارت Ask Every Time نوشته شده است و هر بار که بخواهید برنامه تان طبق پروفایل دستگاه های iPhone اجرا شود از شما میپرسد که اگر شبیه ساز iPhone را دارید آن را انتخاب نمایید) . ولی برای اندروید شبیه ساز داریم . و میتوانیم آن را به جای MoRE انتخاب کنیم. شکل ( )






Preferences
type filter text
General
C/C++
Help
Install/Update
JavaScript
MoSync Tool
Android SDK
Android Signing
Emulators
IPhone Signing
Android SDK
SDK Location
E:\Android SDK\AndroidSDK_2.3
Browse...
برای تعیین شبیه ساز و تست خروجی بر روی شبیه ساز
خاص(مثل اندروید)
Restore Defaults
Apply
OK
Cancel

سایت موسینک و ارتقای نسخه

در حین نوشتن این کتاب بودم که نسخه سوم موسینک با نام MoSync 3.0 عرضه شد. در اینجا مجالب پرداختن به نسخه جدید را ندارم ولی شما میتوانید آن را دانلود و نصب کنید و از آن لذت ببرید.در سایت موسینک (www.MoSync.com) هم اطلاعاتی در مورد نسخه جدید وجود دارد.

یادداشت :

فصل پنجم : ایجاد برنامه های ترکیبی

منظور از برنامه های ترکیبی چیست؟

یکی از مهمترین کارها برای ساخت نرم افزار ایجاد برنامه ای ترکیبی است.منظور از ترکیبی یعنی این که کدهای شما ترکیبی از C++ و HTML5 و جاوا اسکریپت است. اصل واساس کار این است که واسط کاربری را با استفاده از زبان HTML5 ایجاد میکنید و سپس در قسمت کدهای HTML5 با کمک کدهای جاوا اسکریپت و کتابخانه های موسینک که مخصوص جاوا اسکریپت هستند ، کدها و توابع C++ را فراخوانی میکنید و اینگونه یک برنامه قوی که به آن پروژه ترکیبی[1] میگویند ایجاد کرده اید.

هدف اصلی این قسمت : «میخواهیم بین واسط کاربری و کدهای دستوری ، پلی ایجاد نماییم همان کاری که در اکلیپس و برنامه نویسی خاص اندروید برای ارتباط کدهای Xmlو جاوا انجام دادیم».

برای این کار باید از کتابخانه ها و کلاس های زیر کمک بگیرید.

- Wormhole C++ library (WebAppMoblet, MessageStream, MessageStreamJSON)
- JavaScript libraries (included in the file wormhole.js)

[1] Hybrid Project

کتابخانه Wormhole در فایلی که کدهای C++ را مینویسید و عموماً main.cpp است باید include شود و کتابخانه wormhole.js باید در فایل index.html با استفاده از تگ اسکریپت وارد شود.

دو راه برای ایجاد ارتباط وجود داردکه ما در اینجا به هر دو اشاره میکنیم:

1. JSON messages(MessageStreamJSON)
2. string stream messages(MessageStream)

روش اول (JSON messages)

در جاوا اسکریپت با استفاده از تابع توکار[1] **mosync.bridge.sendJSON()** میتوان این پل را ایجاد کرد و یک پیغام برای C++ ارسال کرد.

در زیر تعریف تابع را میبینید :

```
mosync.bridge.sendJSON(message, callbackFunction)
```

پارامتر message میتواند شامل چند مقدار String باشد. پس این پارامتر شامل مجموعه ای از رشته ها است. رشته اول نوع پروتکل را مشخص میکند و باید حتماً تعیین شود ولی رشته های دیگر بسته به نیاز شما دارد و استفاده از آنها اختیاری میباشد.

[1] Built in

پارامتر callbackFunction یک پارامتر اختیاری[1] است.زمانی از این پارامتر استفاده میشود که بخواهید از طریق تابعی که در این قسمت مینویسید مقداری را برای جاوا اسکریپت بفرستید.به بازگشت مقدار به این روش راهکار نامتقارن[2](غیر همزمان) گفته میشود.

روش دوم(string stream messages)

این روش معمولاً از روش JSON messages سریعتر است(در بعضی پلتفرم ها ۲۰ برابر سریعتر است).

```
mosync.bridge.send(stringArray, callbackFunction)
```

با استفاده از تابع بالا میتوان ارتباط بین کدهای جاوا اسکریپت و سی را برقرار کرد.

پارامتر اول که از نامش هم پیداست یک آرایه از رشته ها است. و پارامتر دوم هم مانند روش قبلی است.

نحوه ایجاد پروژه ترکیبی

یادتان باشد که ابتدا باید یک پروژه از نوع Hybrid ایجاد نمایید.

File > New > Project >

---

[1] Optional
[2] Asynchronous Mechanism

MoSync Project > HTML5 > HTML5/JS/C++ Hybrid Project

مثالی که به عنوان Template در هنگام ایجاد پروژه میبیند و ظاهراً توسط یک ایرانی به نام علی صرافی ساخته شده است به شما کمک زیادی میکند تا مفاهیم گفته شده در بالا را درک کنید. در این مثال برای ایجاد ارتباط از هر دو روش JSON message و string stream messages استفاده شده است. در این مثال با ارسال یک پیغام از طریق جاوا اسکریپت به کدهای C++ میگوید که با فراخوانی تابعی به سخت افزار گوشی دستور بده که ویبره[1] دستگاه روشن شود و یا یک بوق به صدا آید و ...

ارسال اطلاعات به جاوا اسکریپت

در بالا توضیح داده شد که چگونه میتوان با استفاده از دستورات جاوا اسکریپت ، توابع C++ را فراخوانی کرد تا کارهای خاص و رویه ای را برای ما انجام دهد.برای مثال به بانک اطلاعاتی وصل شویم تا داده ها را ذخیره کنیم یا یک موسیقی را پخش نماییم و یا ویبره دستگاه را روشن کنیم و یا هر کاری که برای انجام آن نیاز به دستور دادن به سخت افزار داریم و گاهی نیز نیاز به استفاده از کتابخانه ها و کلاس های موسینک که در قسمت کدهای جاوا اسکریپت قابل دستری نیستند. اما برعکس این کار را میخواهیم انجام دهیم.یعنی از درون دستورات C++ توابع جاوا اسکریپت را فراخوانی کنیم.اگر از

[1] Vibrate

درون دستورات C++ بخواهیم ، یک تابع جاوا اسکریپت را فراخوانی کنیم کافی است که دستور زیر را بنویسیم:

callJS("alert('Hello World')");

تابع alert("Message_string") از توابع داخلی جاوا اسکریپت است و بسته به پلتفرم موبایل مقصد یک پیغام را در مقصد نمایش میدهد. این دستور برای پلتفرم های اندروید یک پیغام را در درون کادر مشکی نمایش میدهد.

مانند Toast.makeText(this, "Message_string", 1).show(); عمل میکند.(این دستور مربوط به محیط اکلیپس بود که پیچیده تر است ولی همین کار را انجام میدهد!)

گاهی شما میخواهید یک مقدار را از کاربر بخواهید و آن را در درون پیغام نمایش دهید. فرض کنید که شما سن کاربر را سوال میکنید و سن او در یک متغیر از نوع صحیح ذخیره شده است. برای ارسال متغیر از نوع صحیح تکه کد زیر را بنویسید :

```
int x=getUserAge();
char func[512];
sprintf(func,"alert(%s);",x);
callJS(func);
```

برای ارسال متغیر از نوع رشته ای متاسفانه کد ما کار نمیکند.به احتمال زیاد مشکل از کد ما نمیباشد و این قابلیت توسط طراحین سیستمی موسینک تعریف وطراحی نشده است !

```
char x[]="MiladFashi";
char func[512];
sprintf(func,"alert(%s);",x);
callJS(func);
```

تکه کد بالا را در تست کردم و خروجی نداد.حتی از **char** *x هم استفاده کردم ولی بی نتیجه بود!

اما متخصصین موسینک ریزبین تر از این حرفا هستند. و بعد از بررسی بیشتر به این نتیجه رسیدم که کد من مشکل دارد . اما علت مشکل این بود که در بیشتر زبان ها از جمله سی بهتر است برای جدا کردن یک رشته که در درون رشته دیگر است از تک کوتیشن[1] استفاده کرد، که در کد زیر خطی از برنامه که مشکل از آن بود را اصلاح کردم و نوشتم(قسمت بلد شده اصلاح شده است) :

```
sprintf(func,"alert( ' %s ' );",x);
```

ما همچنین میتوانیم توابع جاوا اسکریپتی را که خودمان تعریف کرده ایم به همین روش صدا بزنیم و برای ارسال هر تعداد پارامتر متوانیم از دستور ، **sprintf** کمک بگیریم.

[1] single quotation

دستور sprintf در اینجا کار Concat (الحاق رشته) را به خوبی برای ما انجام میدهد و از دلایل کاربردی بودن این دستور این است که عملگر + در زبان سی عمل الحاق را انجام نمیدهد.در زبان هایی مثل جاوا و سی شارپ که عملگر بعلاوه هم جمع عددی را انجام میدهد و هم الحاق رشته ها نیازی به همچنین توابعی نمیباشد. برای الحاق رشته ها در سی میتوانید از تابع strcat() هم استفاده کنید ولی استفاده از strcat در این مورد ، ملال آور است. چون تابع strcat() فقط دو پارامتر رشته ای را میگیرد و آنها را الحاق میکند. حالا به این فکر کنید که اگر با strcat بخواهید یک رشته بسازید که شامل نام تابع و پارامترهای تابع است، چقدر به زخمت می افتید .

بنده از تابع بالا برای تست مقادیر برنامه استفاده میکنم.چون که در موسینک اگر شما از پروژه Hybrid استفاده کنید ، توانایی دیباگ کردن و ایجاد Break point را ندارید. یکی از راه های تست برنامه های مخصوص اندروید ، استفاده از همین دستورات است که در بالا به آن اشاره شد.

فراخوانی توابع جاوا اسکریپت

با کمک تابع callJS() شما میتوانید توابع جاوا اسکریپتی که خودتان ساخته اید ، فراخوانی کنید.

برای مثال با استفاده از تکه کد کاربردی زیر اندازه صفحه نمایش گوشی را بدست می آوریم و سپس آن را به سمت کدهای جاوا اسکریپت ارسال میکنیم.به کمک تابع سیستمی **maGetScrSize()** (تابعی از API سیستمی maapi.h) طول و عرض صفحه نمایش را بدست می آوریم و آن را برای تابع جاوا اسکریپت setBoxSize() ارسال میکنیم و این تابع اندازه شی flipbox را بر اساس اندازه اصلی صفحه نمایش دستگاه تنظیم میکند.(یک برنامه موبایل خوب برنامه ای است که قابلیت حمل بالایی داشته باشد ، و یکی از فاکتورهای قابل حمل بودن برنامه ، عدم وابستگی آن به مقیاس صفحه نمایش است و برای مثال یک برنامه خوب برنامه ای است که بر روی یک موبایل ۳ اینچی و یک تبلت ۷ اینچی خروجی یکسان و رابط کاربری یک شکل دارد).

تابع فراخوان :

ابتدا یک تابع جاوا اسکریپتی مینویسیم که هنگام بارگذاری صفحه (html) یک پیغام از نوع string stream messages برای کدهای سی میفرستد :

```
window.onload=function()
{
        mosync.bridge.send(["Custom", "screenSize"]);
}
```

سپس برای پاسخگویی به این پیغام در کدهای سی تکه کد زیر را مینویسیم :

```
void handleMessageStream(WebView* webView, MAHandle data)
```

```
{
        // Create a message stream object. This parses the message data.
        // The message object contains one or more strings.
        MessageStream stream(webView, data);
        // Pointer to a string in the message stream.
        const char* p;
        while (p = stream.getNext()){
         if (0 == strcmp(p, "Custom")){
        const char* command = stream.getNext();
        if (NULL != command && (0 == strcmp(command, "screenSize"))){
        MAExtent scrSize = maGetScrSize();
        int width = EXTENT_X(scrSize);
        int height = EXTENT_Y(scrSize);
        char buf[512];
        sprintf(buf,"setBoxSize(%d,%d)",width,height);
        callJS(buf);
        }
}
}
}
```

سپس با استفاده از تابع جاوا اسکریپت setBoxSize(x,y) ، شی flipbox را که پس زمینه برنامه است را مقدار دهی میکنیم. با اینکار برنامه شما بر روی تمام گوشی های اندروید با اندازه صفحه نمایش متفاوت خروجی یکسان دارد.

```
function setBoxSize(x,y)
{
	$("#flipbox").animate({width:'+='+x},"slow");
	$("#flipbox").animate({height:'+='+y},"slow");
}
```

برای فهم تابع بالا شما نیاز به آشنایی با jquery دارید.برای آشنایی با jQuery میتوانید به فصل jQuery از همین کتاب رجوع کنید و برای کار بیشتر با jQuery میتوانید به سایت jQuery.com مراجعه نمایید.در این سایت مثال های متنوعی قرار داده شده است.

البته برای این کار راه ساده تری هم وجود دارد. برای اینکه اندازه صفحه نمایش را بدست بیاوریم میتوانیم از خصوصیات زبان جاوا اسکریپت استفاده کنیم. مثل خصوصیات زیر :

Screen.width

Screen.height

Screen.availWidth

Screen.availHeight

شی flipbox هم با استفاده از CSS به صورت زیر تعریف شده است :

```
#flipbox {
  width: 10px; //مقادیر اولیه برای شی
  height:10px;
  line-height: 200px;
  background-color: #ff9000;
  font-family: 'ChunkFive Regular', Tahoma, Helvetica;
  font-size: 2.2em;
  color: #ffffff;
  text-align: center;
}
```

برای تمرین و فهم بهتر مفاهیم بالا یک پروژه ترکیبی ایجاد کنید و دست به کار شوید.

یادداشت :

فصل ششم : موسینک و پایگاه داده SQLite

اهمیت کار با پایگاه داده ها

همان‌طور که می‌دانید بخش عمده‌ای از برنامه‌های کاربردی نوشته شده، به نوعی با داده‌های ورودی از سوی کاربر در تعامل است، که بعضی وقت‌ها روند اصلی برنامه را تشکیل می‌دهد.برای مثال در یک برنامه حسابداری، داده‌ها روند اصلی کار برنامه را مشخص می‌کند. داده‌های مالی یک شرکت، اسناد مالی، خرید و فروش و... خروجی مورد نظر کاربر هم با توجه به همین داده‌هاست.اگر شما هم از این مدل برنامه‌ها نوشته باشید می‌دانید که نحوه ذخیره‌سازی آنها مهم‌ترین بخش این برنامه‌هاست. اما این که داده‌های برنامه کجا و چگونه ذخیره شوند، بستگی به نیاز مشتری دارد. اگر قرار باشد داده‌ها به طور متمرکز در یک سرور باشد و بقیه برنامه‌ها ـ که اصطلاحا به آنها خدمت‌گیرنده (Client) می‌گویند ـ باید به آنها دسترسی داشته باشند (تصور این که این Clientها همان برنامه‌های رومیزی (Desktop) هستند، اشتباه است)، یا این که داده‌های هر برنامه مختص خود است و همیشه یک خدمت گیرنده دارد و در یک محیط بسته اجرا می‌شود، در هر دوی این حالت‌ها شما به یک مدل پایگاه داده نیاز دارید. در حالت اول شما پایگاه داده‌ای می‌خواهید که بتواند نیازهای به اشتراک‌گذاری داده‌ها و دسترسی به آنها را براحتی فراهم کند. در مورد دوم نیز می‌توان از همان پایگاه داده‌ای

که در حالت اول ذکر شد استفاده کرد. اما آیا یک رایانه قادر است از تمام توانایی‌های پایگاه‌های داده استفاده کند؟

قطعا جواب خیر است، اما چه باید کرد؟ در این حالت شما با توجه به شرایط کاربری که قرار است با برنامه شما کار کند و بر اساس نیازمندی‌های آن، باید تصمیم بگیرید.

در حال حاضر تعدادی پایگاه داده توسعه داده شده‌اند که قابلیت‌های یک پایگاه داده رابطه‌ای را دارند، و همین طور برای دسترسی به داده‌ها نیازی به نصب هیچ گونه نرم‌افزار اضافی ندارند و آنها را به صورت یک فایل ذخیره می‌کنند. شما می‌توانید فایل‌های داده‌ای خود را همراه خود ببرید و همیشه به آنها دسترسی داشته باشید. این پایگاه داده‌ها مزیت‌هایی دارند و معایبی؛ از مزیت آنها همان‌هایی بود که در بالا ذکر شد، به علاوه سبک بودن و قابلیت انتقال داده‌ها بین هر سیستم‌عامل. در واقع به خاطر ساختار فایلی، آنها مستقل از سیستم‌عامل هستند و همیشه و همه جا می‌توان از آنها استفاده کرد. از معایب آنها به حجم محدودشان می‌توان اشاره کرد و این که بسیاری از قابلیت‌های به اشتراک‌گذاری را مانند پایگاه داده‌های دیگر مانند SQL Server و MySql ندارند و...

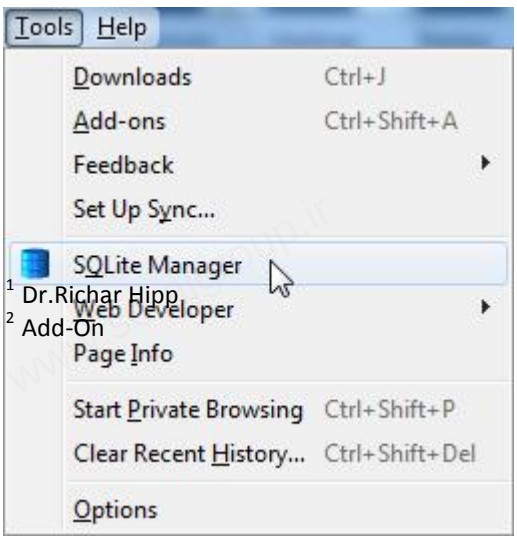
یکی از این پایگاه داده‌ها SQLite است.

تاریخچه SQLite

این پایگاه داده سال ۲۰۰۰ توسط دکتر. ریچارد هیپ[1] زمانی که در نیروی دریایی آمریکا کار می‌کرد توسعه داده شد. . SQLite برنامه‌ای برای کار با داده‌ها ندارد، بلکه فقط یک dll است کهAPI ای برای دسترسی به داده‌ها ارائه می‌کند. این پایگاه داده مانند بقیه پایگاه داده‌های رابطه‌ای امکاناتی مانند Table و تعریف کلید اصلی و کلید خارجی و ارتباط بین آنها را دارد و از این بابت شما نگران هیچ کمبودی نباشید.

چگونه کار با SQLite را شروع کنیم؟

همان‌طور که گفته شد این پایگاه داده، برنامه‌ای خاص برای دسترسی به داده‌ها ندارد و فقط یک کتابخانه برای آن ارائه می‌کند. البته محیط Command(دستوری) هم برای کار با این پایگاه داده وجود دارد ولی کار با آن مشکل و وقت گیر است. همین موضوع باعث شده تا برنامه‌نویسان یک‌سری برنامه با استفاده از API ارائه شده توسط SQLite بنویسند که امکان دسترسی به داده‌ها را فراهم می‌کند.



یکی از آنها یک افزونه[2] نوشته شده برای Firefox است. خوشبختانه Firefoxروی تمام سیستم‌های عامل

---

[1] Dr.Richar Hipp
[2] Add-On

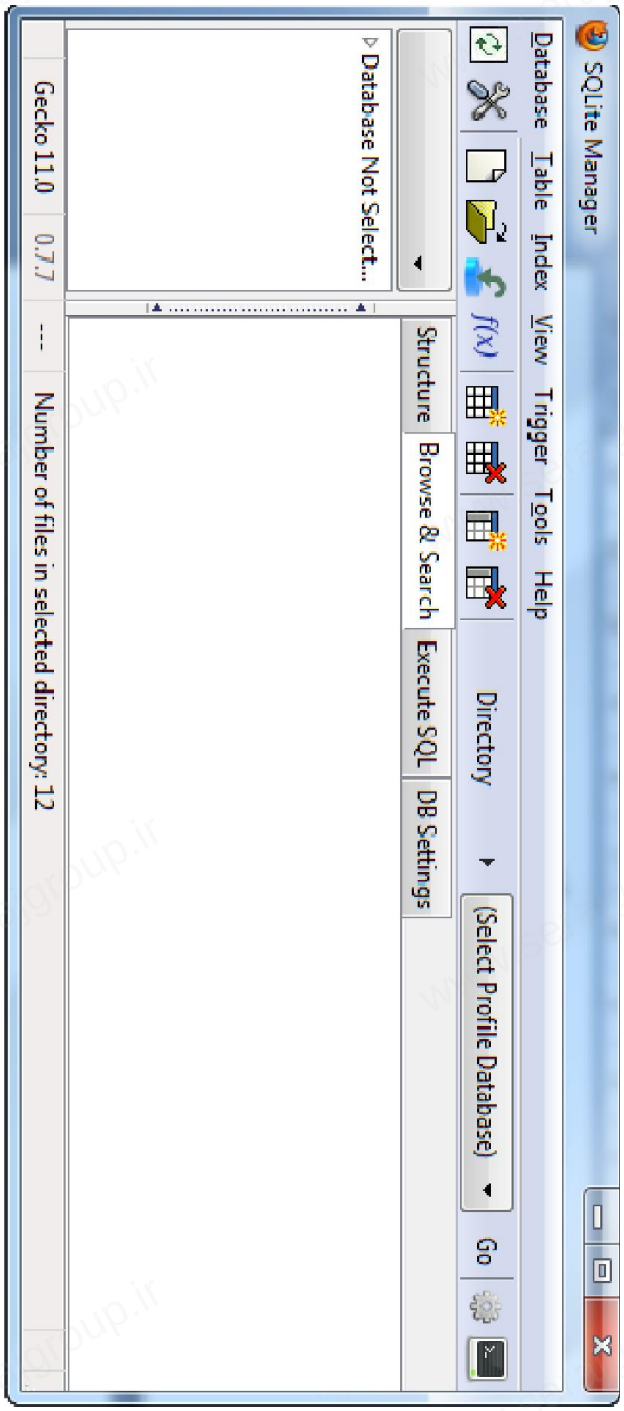
اجرا می‌شود و می‌توانید فایل‌های پایگاه داده خود را در سیستم‌عامل‌های دیگر نیز مشاهده کنید.(شاید دلیل انتخاب فایر فاکس رایگان بودن آن است و در لینوکس و ویندوز بدون مشکل کار میکند).

برای دانلود کردن این افزونه می‌توانید از لینک زیر استفاده کنید:

https://addons.mozilla.org/zh-cn/firefox/addon/sqlite-manager

بعد از نصب این افزونه ، مرورگر Fire Fox را باز کنید .مطابق شکل( ) از منوی Toolsگزینه SQLite Manager را انتخاب کنید. سپس پنجره همانند شکل( ) برای شما باز می‌شود که در آن شما ابتدا با استفاده از منوی Database و گزینه New Database فایل پایگاه داده خود را می‌سازید و سپس با توجه به نوار ابزار می‌توانید جداول خود را که در واقع ساختار پایگاه داده شما را مشخص می‌کند، طراحی کنید.کار با این محیط بسیار ساده است و اگر شما قبلاً تجربه کار با نرم افزار های مدیریت پایگاه داده های رابطه ای[1] را داشته باشید، با ابزارهای این افزونه کاملاً آشنایی دارید.

[1] RDBMS


SQLite Manager
Database Table Index View Trigger Tools Help
Directory
(Select Profile Database)
Go
Structure
Browse & Search
Execute SQL
DB Settings
Database Not Select...
Gecko 11.0
0.7.7

Number of files in selected directory: 12

از منوی Database و گزینه Connect Database به داده‌های پایگاه داده‌ای که قبلا ایجاد کردید نیز دسترسی داشته باشید.

اما شما به عنوان یک برنامه‌نویس قرار است چگونه با این دیتابیس کار کنید؟

فراهم‌کننده[1] های زیادی برای دسترسی به این نوع داده‌ها نوشته شده‌اند. برای هر زبان با توجه به امکانات آن زبان یک Provider مخصوص نوشته شده است، این نکته را هم باید در نظر گرفت که API ارائه شده از طرفSQLite ، به زبان C++ است، و شما در C++براحتی می‌توانید با آن کار کنید.

چند نمونه از Provider های توسعه داده شده برایSQLite

1ـ :Php این دیتابیس به صورت محلی در php پشتیبانی می‌شود و نیازی به استفاده از Providerنیست.

2ـ :Java برای دسترسی به این پایگاه داده می‌توان از SQLiteJDBC استفاده کرد، برای دانلود آن می‌توانید از نشانی زیر استفاده کنید:

http://www.zentus.com/sqlitejdbc

3ـ :.NET. برای دات نت یک Provider متن باز توسط شرکت phx software توسعه داده شده است، که برای دانلود آن می‌توانید به این نشانی بروید:

---

[1] Provider

http://sqlite.phxsoftware.com

مطالب فوق از ضمیمه کلیک روزنامه جام جم آورده شده است.

در زیر گزیده ای از مطالب کتاب انگلیسی زبان زیر را ترجمه کرده ام ،تا بتوانم شما را بیشتر و بهتر وارد کنه ماجرا کنم.

The Definitive Guide to SQLite

Author: Michael Owens

Publisher : Apress

SQLite را بهتر و بیشتر بشناسید

SQLite یک پایگاه داده ادغام شده [۱] است و از خانواده پایگاه داده های رابطه ای[۲] است. که در سال ۲۰۰۰ منتشر شد. ویژگی های مهم این پایگاه داده که آن را مشهور کرده است قابلیت حمل بالا[۳] ، سادگی استفاده از آن[۴]، کم حجمی و فشردگی[۵] ، کارایی[۶] خوب و قابلیت اطمینان[۷] است.

---

[1] Embedded Database
[2] Relational DB
[3] Portability
[4] Easy to Use
[5] Compact
[6] Efficient
[7] Reliability

منظور از فشردگی و کم حجمی این است که در این پایگاه داده API ها و کتابخانه های C و کلاینت و سرور و به طور کلی همه چیز در یک فایل ذخیره میشود و مزیت آن این است که شما نیاز به پیکربندی شبکه[1] و Administration(مدیریت) ندارید.وقتی که سرور و کلاینت هر دو بر روی یک فایل باشند در نتیجه سربار[2] مربوط به پردازشِ فراخوانی های شبکه[3] کاهش میابد ، و مدیریت پایگاه داده ساده تر میشود. از سوی دیگر باعث میشود که یک پایگاه داده سبک وزن[4] داشته باشیم که بر روی یک فلاپی هم جا میشود.پس براحتی میتوان آن را در هر نرم افزار و سخت افزاری تعبیه کرد.

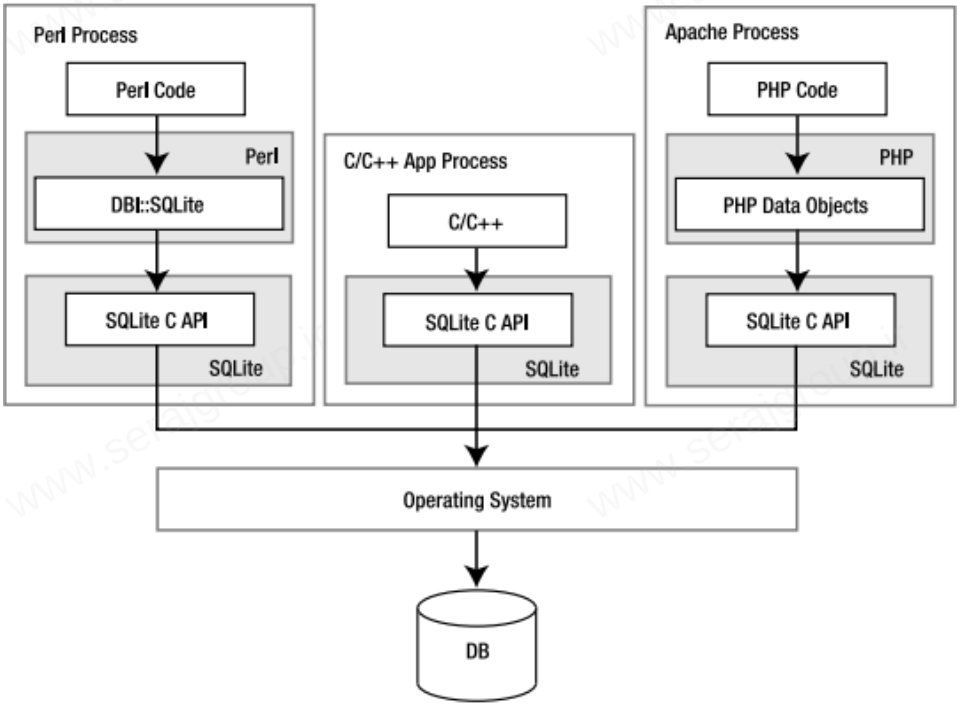
انواع مختلف نرم افزارهای پایگاه داده های رابطه ای تولید شده اند.خصوصاً پایگاه داده های رابطه ای توکار (ادغام شده).محصولاتی مانند Sybase SQL any where,Microsoft 'S Jet Engine,... از نوع پایگاه داده های رابطه ای توکار هستند.

بعضی از کمپانی ها پایگاه داده های بزرگ(Large-Scale) خود را تغییر داده اند و نسخه توکار(Embedded) آن را تولید کرده اند! برای مثال IBM 'S DB2 EveryPlace, Oracle 's log.

---

[1] Network Configuration
[2] Overhead
[3] Network Call
[4] LightWeight

یکی از دلایل اصلی که باعث محبوبیت SQLite در بین این همه نرم افزار مختلف پایگاه داده شد . متن باز بودن آن بود.از دلایل دیگر طراحی خوب و ماجولار آن بود.که باعث **توسعه** آن شد.**متن باز بودن و ماجولار بودن** آن باعث شد که Provider های مختلفی توسط کمپانی ها و انجمن های طراح و پیاده ساز زبان های برنامه نویسی ساخته شود.و خیلی سریع SQLite توسعه داده شد . در شکل( )برخی از این زبان های میزبان را میبینید که برای کار با این پایگاه داده یک رابط کدنویسی شده ایجاد کرده اند.



همان طور که در شکل میبیند زبان C نیاز به رابط ندارد.چون که SQLite با زبان C طراحی شده است.

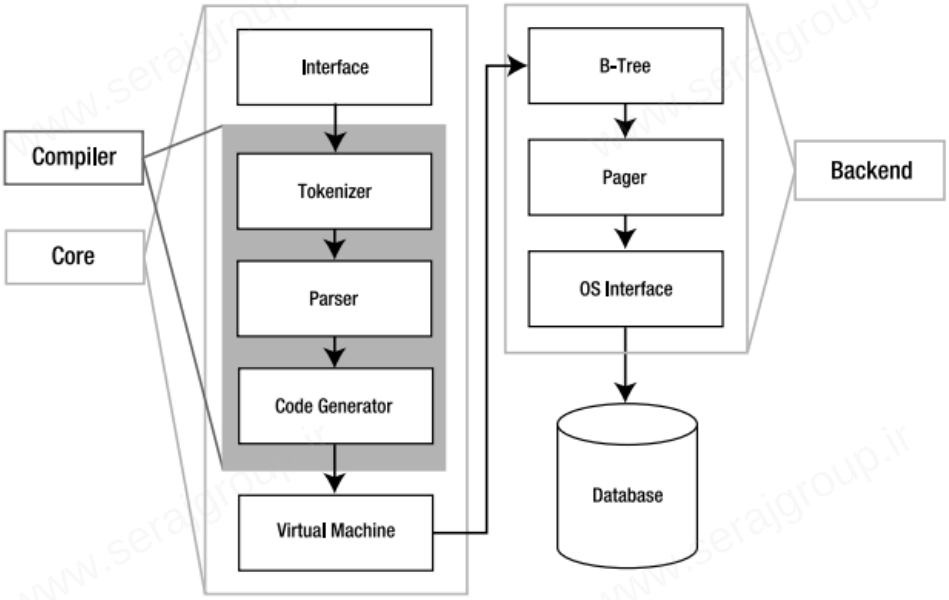
از SQLite در نرم افزارها و سخت افزارهای مختلفی به صورت توکار[1] استفاده شده است.برخی از سخت افزار ها و نرم افزارها و سیستم عامل هایی که از آن استفاده کرده اند در زیر به آن اشاره شده است :

- Apple’s Mac OS X
- Safari web browser, Mail.app email program, RSS manager, Apple’s Aperture photography software.
- Solaris operating environment
- Mozilla Project’s mozStorage
- C++/JavaScript API layer
- Firefox, Thunderbird, Sunbird.
- SQLite has been added as part of the PHP 5 standard library.
- Linux-based Palm OS
- Symbian OS platform.Andriod OS platform.And most smart phone OS and cell phone applications .
- D-Link Media Lounge.
- Music players.

[1] Embedded

- Complete New Yorker DVD set-a digital library of every issue of the New Yorker magazine-.

همانطور که گفته شد ، SQLite یک طراحی خوب و ماجولار دارد. معماری پیمانه ای[1] خوب و زیبای این پایگاه داده را در شکل( ) مشاهده کنید.



Interface که در بالاترین لایه قرار دارد ، شامل SQLite C API میشود. منظور از Tokenizer فاز Lexical Analyser کامپایلر است.(همان Scanner است که به آن اتوماتا هم میگویند).

[1] Modular Architecture

پارسر هم که فاز Syntax Analyser است که به آن گرامر هم میگویند. اما سوالی که میتواند بحث برانگیز باشد این است که چرا در فاز های کامپایل فاز Semantic Analyser دیده نمیشود؟

در اینجا کار کامپایلر گرفتن دستورات زبان سطح بالای SQL[1] است.این لایه بعد از بررسی صحت نحوی دستورات [2] ، آنها را به داده ساختارهایی[3] تبدیل میکند که توسط لایه های پایین تر قابل فهم باشد.

لایه Virtual Machine که به آن virtual database engine (VDBE) نیز میگویند. این ماشین مجازی همانند ماشین مجازی جاوا بر روی بایت کدها کار میکند. بایت کدها [4] برای این ماشین مجازی حکم یک زبان را دارد. پس به آن میتوانیم زبان مخصوص ماشین مجازی بگوییم.این زبان ۱۲۸ آپ کد (opcode) برای عملیات های پایگاه داده دارد.هر دستوری که در زبان SQL وجود دارد مثل Select,update,delete,... به زبان مخصوص ماشین مجازی تبدیل میشود.

مثال :

دستور SQL:

---

[1] **S**tructured **Q**uery **L**anguage
[2] Syntax Analyase and Validation
[3] Data Structure
[4] VDBE's byte code

```
SELECT name FROM episodes LIMIT 10;
```

معادل بایت کد(مخصوص ماشین مجازی)

```
0  Integer       10   0
1  MustBeInt     0   0
2  Negative      0   0
3  MemStore      0   1
4  Goto          0  15
5  Integer       0   0
6  OpenRead      0   2
7  SetNumColumns     0   4
8  Rewind        0  13
9  MemIncr       0  13
10 Column        0   3
11 Callback      1   0
12 Next          0   9
13 Close         0   0
14 Halt          0   0
15 Transaction           0   0
16 VerifyCookie          0 190
17 Goto          0   5
18 Noop          0   0
```

صحبت در مورد این لایه ها از بحث ما خارج است اما اگر خواننده با مفاهیم دروسی مانند پایگاه داده ها ، کامپایلر ، ساختمان داده ها و ذخیره و بازیابی اطلاعات آشنایی داشته باشد. براحتی مفاهیم فوق را درک میکند و معماری زیبای این پایگاه داده را بهتر درک میکند.برای اطلاعات بیشتر میتوانید به کتاب فوق الذکر مراجعه نمایید و یا از سایت www.SQLite.org استفاده نمایید.

مروری بر ویژگی های SQLite

به طور خلاصه برخی از ویژگی های مهم آن را بررسی میکنیم :

- Portability(قابل حمل بودن) : بر روی پلتفرم های ۱۶و۳۲و۶۴ بیتی کار میکند. و هر دو فرمت Big , Little Endian Byte Order را پشتیبانی میکند.و از Encoding های UTF-8,UTF-16 پشتیبانی میکند.
- Compactness (فشرده و کم حجم) : SQLite به گونه ای طراحی شده است که سبک وزن باشد. هدر فایل ها ، کتابخانه ها ، کلاینت و سرور و ماشین مجازی همگی در یک چهارم مگابایت(۲۵۶ کیلوبایت) جا میگیرند!
- Simplicity (سادگی) : استفاده از API های آن بسیار ساده است.

- Flexibility (انعطاف پذیری) : چندین فاکتور دست به دست هم داده اند و باعث شده اند که SQLite انعطاف پذیر باشد . یکی از این فاکتور ها معماری پیمانه ای آن است که در مورد آن بحث شد.

محدودیت های استفاده از SQLite

- اگر شما پایگاه داده بزرگی دارید و پرس و جوهای (Query) پیچیده ای نیاز دارید با افزایش زمان پاسخ مواجه میشوید.در این موارد توصیه میشود که از پایگاه داده های غیر توکار که برای کار با اطلاعات زیاد و پرس و جو های پیچیده ساخته شده اند استفاده شود(مثل SQL Server).یادتان باشد که SQLite یک پایگاه داده توکار است که برای برنامه های کوچک تا مقیاس متوسط(Medium-Scale) ساخته شده است.خیلی از تازه کار ها فکر میکنند که برای برنامه های بزرگشان میتوانند از پایگاه داده SQLite استفاده کنند ولی در نهایت میبینند که سیستم طراحی شده از کارایی و سرعت پایینی برخوردار است.

- اگر اندازه دیتابیس شما بیشتر از ۲ ترابایت است نمیتوانید از SQLite استفاده کنید.چونکه این دیتابیس حداکثر میتواند 2Terabyte داده را ذخیره کند.

- در دستگاه های قابل حمل مثل موبایل ها ، چونکه حافظه اصلی(RAM) کوچکی دارند شما باید همواره اندازه دیتابیس را محاسبه کنید تا کارایی برنامه پایین نیاید.چونکه برای هر مگابایت از اطلاعاتی که در دیتابیس قرار میگیرد این نرم افزار برای آن ۲۵۶ بایت حافظه RAM احتیاج دارد(برای تخصیص dirty page ها که یکی از کارهای سیستمی این نرم افزار است و به طور کلی به آن Bitmap Allocation میگویند). برای مثال یک پایگاه داده ۱۰۰ گیگابایتی به ۲۵ مگابایت از حافظه RAM احتیاج دارد.
- اگر میخواهید از تراکنش های تودرتو[1] استفاده کنید . SQLite از این قابلیت پشتیبانی نمیکند. به صورت کامل از Trigger ها هم پشتیبانی نمیکند. و نیز View ها در SQLite فقط خواندنی هستند.

محدودیت های دیگری هم وجود دارند ولی محدودیت های بالا بیشتر به چشم می آیند.

کار با پایگاه داده SQLite در موسینک

موسینک برای کار با پایگاه داده ، دارای API کار با پایگاه داده است که از توابع سطح پایین(low-level) زیر برای کار با پایگاه داده SQLite استفاده میکند :

[1] Nested Transaction

- **maDBOpen** : دیتابیس را باز میکند و اگر وجود نداشت آن را ایجاد میکند.
- **maDBClose** : برای بستن دیتابیس.
- **maDBExecSQL** : یک دستور SQL را اجرا میکند.اگر این دستور دارای مقدار بازگشتی باشد یک cursor handle را برمیگرداند.
- **maDBExecSQLParams** : یک دستور SQL که حاوی پارامتر است را اجرا میکند(در حال حاظر فقط در MoRE و iOS کار میکند).
- **maDBCursorDestroy** : این تابع cursor ای که در اجرای توابع **maDBExecSQL** , **maDBExecSQLParams** بازگشت داده شده است را نابود میکند و حافظه آن را پس میگیرد.
- **maDBCursorNext** : مقدار کرسر را یک واحد افزایش میدهد.یعنی کرسر را به سطر بعدی از مجموعه جواب(Result Set) میبرد.**شما بایستی قبل از خواندن اطلاعات این تابع را فراخوانی کنید ، چونکه مقدار اولیه کرسر یک واحد قبل از اولین سطر است.**
- **maDBCursorGetColumnData** : در سطر جاری (سطری که کرسر به آن اشاره میکند) مقدار داده موجود در ستونی که توسط ایندکس مشخص شده است را به صورت data handle بر میگرداند. (منظور از ایندکس یا شاخص

عددی است که ستون(فیلد) مورد نظر از سطر یا (رکورد) جاری را مشخص میکند و در اینجا این عدد از صفر شروع میشود).

- **maDBCursorGetColumnText** : همانند تابع قبلی است ، با این تفاوت که مقدار بازگشتی آن از نوع رشته است.
- **maDBCursorGetColumnInt** : همانند توابع قبلی است ، با این تفاوت که مقدار بازگشتی این تابع int است.
- **maDBCursorGetColumnDouble** : : همانند توابع قبلی است ، با این تفاوت که مقدار بازگشتی این تابع double (اعشاری با دقت مضاعف) است.

استفاده معمول و رایج از پایگاه داده

معمولاً شما برای وصل شدن به هر نوع پایگاه داده ای باید یک سری روال را به صورت قدم به قدم انجام دهید. در اینجا هم برای وصل شدن و خواندن اطلاعات از دیتابیس بایستی به ترتیب کارهای زیر را به کمک Database API انجام دهید:

۱. بانک اطلاعاتی را با استفاده از تابع maDBOpen() باز کنید.آدرس کامل[1] فایل دیتابیس را به این تابع میدهید.این تابع این آدرس را بررسی میکند و اگر

[1] Absolute path

دیتابیسی که شما آدرسش را نوشته اید پیدا نکند ، در همان آدرس یک دیتابیس جدید ایجاد میکند.

۲. با استفاده از تابع maDBExecSQL میتوانید دستورات insert,update,delete,... را بنویسید. و این تابع این دستورات که به آنها Command هم میگویند بر روی بانک اجرا میکند و اگر دستور مانند select دارای خروجی باشد، Cursor Handle برمیگرداند.تا شما از طریق کرسر بتوانید به داده ها دسترسی داشته باشید.

۳. با استفاده از کرسری که در مرحله ی قبل بدست آورید میتوانید به کمک تابع maDBCursorNext جلو ببرید و رکورد به رکورد داده ها را پیمایش کنید.این کرسر از نظر طرز کار مانند اشاره گر آرایه است که خانه های آرایه را پیمایش میکرد. اما در اینجا اشاره گر به رکورد است و رکورد های یک جدول را پیمایش میکند.

۴. در آخر هم بایستی حافظه ای که این کرسر گرفته است را بازپس بگیرید.برای این کار از تابع maDBCursorDestroy استفاده کنید.

۵. در نهایت هم باید برای این که ارتباط شما با دیتابیس قطع شود بایستی با استفاده از تابع maDBClose() آن را ببندید.

توابعی که گفته شد.در واقع توابع کتابخانه بزرگ maapi هستند.در این کتابخانه توابع سطح پایین [1] زیادی برای دسترسی به ویژگی های اصلی دستگاه (موبایل) وجود دارد. از آنجا که موسینک Open Source است ، شما هم میتوانید با شرکت موسینک از طریق ایمیل در ارتباط باشید و توابع سطح پایینی که خودتان نوشته اید و یک کار خاص را انجام میدهد به این کتابخانه اضافه کنید.هدف موسینک این است که شما خودتان قادر به شخصی سازی آن باشید و حتی توابع سیستمی برای آن بنویسید.

مثال :

ابتدا دیتابیسی به نام Vocabulary.sqlite ایجاد کنید.برای این کار میتوانید از افزونه[2] SQLite Manager استفاده کنبد. (در مورد این افزونه و نحوه نصب آن توضیح داده شد).

سپس جدولی به نام Words ایجاد کنید. برای این جدول دو ستون از نوع VARCHAR بسازید.ستون اول(english) مربوط به لغات انگلیسی است و ستون دوم(persian) معنی لغت به زبان فارسی است.هر دو ستون هم نمیتوانند تهی باشند.برای این کار میتوانید از

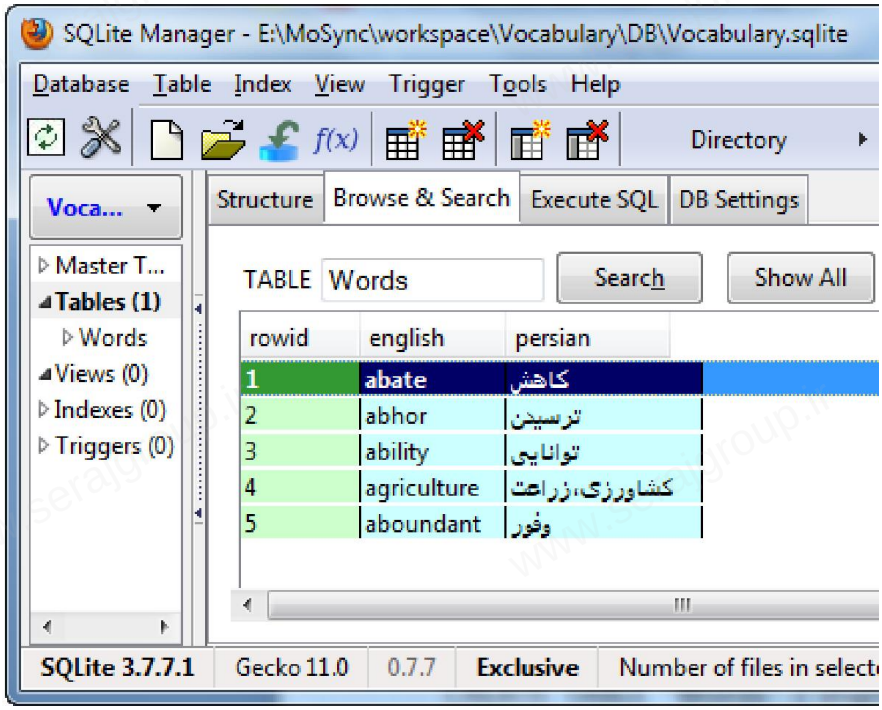
[1] low-level functions
[2] Add-on

رابط کاربری نرم افزار استفاده کنید و یا دستور آن را بنویسید.دستور SQL آن به شکل زیر است :

```
CREATE TABLE "Words" ("english" VARCHAR NOT NULL , "persian" VARCHAR NOT NULL )
```

این جدول را با مقادیری پر کنید. شکل ( )

در بخش بعدی میخواهیم از طریق موسینک اطلاعات موجود در این دیتابیس را بخوانیم.این اطلاعات میتواند اطلاعات یک برنامه آموزش ۵۰۴ یا لغت های ضروی برای تافل باشد. که برای موبایل ساخته شده است تا همیشه همراهتان باشد. بانک های شما میتواند پیچیده تر از این باشد و دارای کلید اصلی و خارجی باشد. ولی برای شروع همین دیتابیس ساده مناسب است.



اگر چه برای کار با دیتابیس شما بایستی از توابع و کلاس های هدرفایل زیر استفاده کنید ولی **نیاز نیست** که هدر فایل زیر را به برنامه خود اضافه کنید!

```
#include "maapi_defs.h"
```

برای باز کردن دیتابیس بایستی از تابع maDBOpen() استفاده نمایید.

تعریف این تابع به شکل زیر است :

```
MAHandle maDBOpen(const char* path)
```

Path ثابت رشته ای است که آدرس مطلق دیتابیس در آن نگهداری میشود. نحوه آدرس دهی در اندروید به آدرس دهی لینوکس شباهت دارد. مثلاً اسم دیتابیس ما Vocabulary.sqlite است و در مموری کارت در آدرس زیر قرار دارد :

"/mnt/sdcard/Vocabulary.sqlite"

متغیری به شکل زیر تعریف کنید(در زبان C ، رشته با طول متغیر با استفاده از اشاره گر کارکتری ایجاد میشود) :

```
const char *path="/mnt/sdcard/Vocabulary.sqlite";
```

تابعی که از این دیتابیس اطلاعات را میخواند به شکل زیر است :

```
void openDataBase(){
        const char *path="/mnt/sdcard/Vocabulary.sqlite";
        MAHandle MAHDB = maDBOpen(path);
        MAHandle cursor;
        cursor = maDBExecSQL(MAHDB,"Select * from Words");
        char func[512];
        char eword[50];
        maDBCursorNext(cursor);
        maDBCursorGetColumnText(cursor, 0, &eword,50);
        sprintf(func,"alert('%s')",eword);
        callJS(func);
        maDBClose(MAHDB);
```

}

تعریف تابع maDBExecSQL() به شکل زیر است :

```
MAHandle maDBExecSQL(MAHandle databaseHandle,
                     const char * sql )
```

تعریف تابع maDBCursorNext() به صورت زیر است :

```
int maDBCursorNext(MAHandle cursorHandle)
```

این تابع اگر با موفقیت اجرا شود مقدار صفر را برمیگرداند. در صورتی که تابع با خطا مواجه شود مقدار 2- را برمیگرداند. که معادل این ثوابت هستند :

```
#define MA_DB_OK  0
#define MA_DB_ERROR  -2
```

تابع maDBCursorGetColumnText() به شکل زیر است :

```
int maDBCursorGetColumnText(MAHandle cursorHandle,int colum
nIndex,void* buffer,int bufferSize)
```

cursorHandle اشاره گری انتزاعی برای دسترسی به رکوردهای دیتابیس.

columnIndex برای دسترسی به ستونها (صفات خاصه) استفاده میشود و اولین ستون دارای مقدار صفر است.

Buffer	اشاره گر به بافری برای نگه دارای داده های خوانده شده از دیتابیس.
bufferSize	برای تعیین اندازه بافر.

فصل هفتم : کار با jQuery و jQuery Mobile

متاسفانه فرصت نکردم این فصل رو بنویسم.

فایل رو باز گذاشتم هر کسی دوست داشت ادامه این فصل رو بنویسه.

طراحی محیط کاربری پروژه

هدف ما از طراحی محیط کاربری ، علاوه بر زیبایی محیط بایستی به انعطاف پذیر بودن هم توجه کنیم.طوری که برنامه ما در گوشی های مختلف که دارای رزولوشن متفاوت هستند ظاهر یکسانی داشته باشد.

در این فصل میخواهیم محیط کاربری زیبا و UserFriendly برای نرم افزار انگلیسی در سفر ایجاد کنیم که بر روی گوشی ها و تبلت ها اجرا شود و ظاهر یکسانی داشته باشد.

برای مثال یک برنامه خوب که دارای محیطی زیباست ، دارای تم های مختلف است. تم اول از ترکیب رنگ های زرد و سفید درست شده است و تم دوم از ترکیب رنگ های مشکی و آبی ساخته شده است و ... . بنابراین بسته به تم ها به آیکون های با سایز های متفاوت برای تامیین رنگ های زرد و آبی نیاز داریم.

برای انعطاف پذیر بودن برنامه برای هر آیکون نیاز به سه نسخه از آن داریم :

۱- HDPI=high dot per inch برای تبلت ها و بعضی از گوشی های با رزولوشن بالا

۲- MDPI=medium dot per inch معمولاً برای گوشی های با کیفیت متوسط

۳- LDPI=low dot per inch برای گوشی های با رزولوشن پایین

اما گوشی ها با توجه به رزولوشن و وضوحی که دارند در کدام یک از این دسته ها جای میگیرند؟

در جدول ( ) انواع مختلف گوشی ها و تبلت ها را بسته به اندازه و سایزشان مشاهده میکنید:

Device Name	Screen Size	Screen Resolution	DPI
Motorola Atrix	4	540 x 960	275
Google Nexus 1	3.7	480 x 800	254
iPhone 3GS	3.5	320 x 480	163
iPod Retina Display	3.5	640 x 960	326
iPad	9.7	1024 x 768	132
PlayBook	7	1024 x 600	168
Galaxy Tab 7'	7	1024 x 600	168
Motorola Xoom	10.1	1280 x 800	150
Nook Color	7	1024 x 600	168

تنوع مدل ها خیلی بیشتر از جدول بالاست. حتی در iPad 5(iOS5) که اخیراً وارد بازار شده است رزولوشون تصویر ۱۵۳۶*۲۰۴۸ است که عدد خیلی بزرگی است و اندازه فیزیکی آن هم ۹ اینچ است.برای مثال حداکثر سایز میتواند iPad باشد و حداقل سایز

گوشی Galaxy mini که تقریباً صفحه ۳ اینچی دارد و رزولوشن آن ۳۲۰*۲۴۰ است.برای ساخت برنامه ها باید حداکثر سایز و حداقل سایز را مشخص کنید.سپس با استفاده از یک سری تکنیکها میتوان برای تمام دستگاه هایی که اندازه صفحاتشان بین این دو بازه است برنامه بنویسید. یک برنامه نویس موبایل همیشه باید سعی کند برنامه ای بنویسد که مثلاً بر روی دستگاه های بزرگتر از ۳ اینچ و کوچکتر از ۱۰ اینچ به خوبی کار کند.برای مثال یک فایل APK داریم که این فایل بر روی یک دیوایس ۳ اینچی و یک دیوایس ۱۰ اینچی خروجی قابل قبول و بدون مشکلی را دارد.

برای این کار تکنیک های زیادی وجود دارد. ابزارهایی که در اینجا به ما کمک میکنند که چنیین برنامه جذابی بسازیم CSS3(Media Query) و جاوا اسکریپت (screen.width,screen.height) هستند.

اما متدهای متفاوتی برای استفاده از این ابزارها وجود دارد. طراحی فایل های CSS جداگانه برای هر دیوایس و یا ایجاد یک فایل CSS و .... برای آشنایی بیشتر با Media Query میتوانید به فصل Adapting Pages for Mobile with Media Queries که در کتاب Adobe Dreamweaver CS5.5 Studio Techniques Designing and Developing for Mobile with jQuery, HTML5, and CSS3 که نویسنده آن آقای DAVID POWERS است مراجعه کنید.

مراجعه به آدرس زیر هم میتواند بسیار سودمند باشد :

http://developer.android.com/guide/practices/screens_support.html

لغات و مفاهیم

Screen Size :معمولاً با اینچ داده میشود و اندازه قطر صفحه نمایش به اینچ است.

Screen density: نشان دهنده کمیت[1] پیکسل ها در ابعاد فیزیکی صفحه است که با نقطه در اینچ و یا پیکسل در اینچ بیان میشود.

تذکر : نقطه در اینچ(DPI=dot per inch) با پیکسل در اینچ(PPI=pixel per inch) برابر است. گاهی از dpi استفاده میشود و گاهی نیز از ppi . ولی معمولاً برای پرینتر ها و اسکنر ها بیشتر از dpi استفاده میشود و برای صفحه نمایش (خصوصاً موبایل و تبلت) از ppi استفاده میشود. ولی در هر صورت هر دو واحد یکی هستند و نیاز به تبدیل ندارد.

Orientation : جهت صفحه نمایش است. که دو حالت افقی(landscape) و عمودی(portrait) دارد و برای تشخیص اینکه موبایل در چه وضعیتی است تا طبق آن واسط کاربری هم بچرخد بسیار کاربرد دارد.و با چرخاندن صفحه توسط کاربر aspect ratio تغییر میکند و یک برنامه خوب برنامه ای است که طبق aspect ratio جدید تنظیم شود.

[1] Quantity

Resolution: تعداد پیکسل های موجود در سطر و ستون گوشی را رزولوشن گویند.

نکته مهم :

برای ساخت یک برنامه که قرار است بر روی دستگاه های با سایزهای مختف اجرا شود، شما بایستی علاوه بر رزولوشن به dpi گوشی هم توجه کنید. و گرنه به هدف خود که ساخت یک برنامه که دارای ویژگی Multiple Screen Support است نمیرسید! پس هم رزولوشن و هم چگالی(density) در تعیین ابعاد باید دخالت داشته باشد. به همین خاطر نبایستی از واحد پیکسل در برنامه خود استفاده کنیم.حال این سوال پیش می آید چرا از پیکسل استفاده نکنیم؟ و به جای آن از چه واحدی استفاده شود؟

واحدی که گوگل پیشنهاد کرده است واحد *Density-independent pixel (dp)* است.

Density-independent pixel (dp) : این واحد یک پیکسل مجازی است که بایستی در طراحی واسط گرافیکی کاربر از این واحد استفاده کنید و گرنه با مشکلات عدم تطبیق با سایز های مختلف مواجه میشوید. این واحد به صورت زیر محاسبه میشود :

dp = px / (dpi / 160)

dp: Density-independent pixel

px: Pixel

dpi: dot per inch

بدیهی است که برای رسیدن از *dp* به *px* بایستی این کار را کرد :

px = dp * (dpi / 160)

عدد ۱۶۰ که در بالا آمده را گوگل تعیین کرده است. گوگل چگالی ۱۶۰ پیکسل در هر اینچ را به طور ثابت برای همه ی دستگاه های اندرویدی در نظر گرفته است تا با مشکل عدم تطبیق مواجه نشوید و نیز با واحدی مستقل از اندازه فیزیکی و صفحه نمایش کار کنید و آن واحد همان *Density-independent pixel* است .

برای مثال گوشی اندرویدی سامسونگ گالاکسی مینی(s5570) دارای رزولوشن پایینی است. رزولوشن آن ۳۲۰*۲۴۰ است. و چگالی آن dpi ۱۲۰ است. گوشی Sony Ericsson Xperia X10 رزولوشن آن ۸۵۴*۴۸۰ است و تقریباً ۲۴۵ عدد dpi آن است. برای اینکه یک چگالی مشترک داشته باشیم ، چگالی هر دو گوشی را ۱۶۰ در نظر میگیریم تا با واحد مستقل از دستگاهی مثل *Density-independent pixel* کار کنیم.

در زیر برای دو گوشی dp را محاسبه کرده ایم.

❖ برای گوشی گالاکسی مینی :

اندازه عرض تمام صفحه موبایل dp(width) = 240 / (120 / 160) = 320

به این معنی است که اگر در هر اینچ از صفحه ۱۲۰ پیکسل قرار داده شود به ۲۴۰ پیکسل لازم است تا عرض صفحه ی این موبایل پر شود. و اگر در هر اینچ ۱۶۰ پیکسل قرار داده شود به ۳۲۰ پیکسل لازم داریم تا تمام عرض صفحه موبایل را در بر گیرد.(موبایل گالاکسی مینی سامسونگ عرض ۲ اینچی دارد. و بدیهی است که وقتی چگالی ۱۲۰ پیکسل در اینچ است به ۲۴۰ پیکسل احتیاج است پس اندازه عرض صفحه بایستی ۲ اینچ باشد).

dp(Height) = 320 / (120 / 160) = 426.67

اهمیت عرض صفحه از طول آن بیشتر است. چون که کمبود های طول صفحه به نوعی با اسکرول عمودی جبران میشود !

❖ برای گوشی Xperia X10 :

dp (width) = 480 / (240 / 160) = 320

به این معنی است که اگر در هر اینچ از صفحه ۲۴۰ پیکسل قرار داده شود به ۴۸۰ پیکسل لازم است تا عرض صفحه ی این موبایل پر شود. و اگر در هر اینچ ۱۶۰ پیکسل قرار داده شود به ۳۲۰ پیکسل لازم داریم تا تمام عرض صفحه موبایل را در بر گیرد.

dp (Height) = 854 / (240 / 160) = 569.3

سوالی که پیش می آید این است که این تبدیلات بر روی کیفیت صفحه تاثیر منفی نمیگذارد؟

همان طور که قبلاً اشاره شد برای گرفتن اندازه صفحه نمایش هم میتوانید از جاوا اسکریپت استفاده کنید و هم از تابع سیستمی موسینک(**maGetScrSize**).

توصیه بنده استفاده از جاوا اسکریپت است. زیرا اولاً نیاز به محاسبه ابعاد در C++ نیستید و در همان فایل HTML این کار را انجام میدهید در نتیجه رد و بدل اطلاعات بین C++ و جاوا اسکریپت صورت نمیگیرد و همین در سرعت اجرای برنامه شما تاثیر گذار است (اگر چه تاثیر آن بر سرعت ناچیز است) . دوماً دستورات screen.width و screen.height هر دو چگالی را ۱۶۰ در نظر میگیرند و مستقیماً به ما -*Density independent pixel* را میدهند! و اینکه جاوا اسکریپت با این دقت بالا کار میکند نشان دهنده ی قدرت این زبان است و این خود جای تحسین دارد. که یک زبان که سال ها پیش ساخته شده است با این دقت برای دستگاه های موبایل و تبلت هم اعداد درست و دقیق را به ما میدهد.

مطمئن باشید که اعدادی را که جاوا اسکریپت برای عرض صفحه میدهد دقیقاً برابر اعدادی هستند که از فرمول بالا بدست می آید. ولی اعدادی را که برای طول صفحه میدهد کوچکتر است و آن هم برای شما مشکلی ایجاد نمیکند. چون که جاوا اسکریپت برای طول صفحه تمام طول صفحه را نمیدهد و طولی از صفحه را به شما میدهد که قابل

استفاده است و برای مثال اگر نواری که در بالای صفحه ی گوشی شما است ۲۵ پیکسل ارتفاع دارد این عدد را از ارتفاع کم میکند مثلا جاوا اسکریپت برای این دستگاه عدد۵۴۴ را برای ارتفاع به ما میدهد. برای اینکه مطمئن شویم جاوا اسکریپت در محاسبات اشتباه نکرده است کاری میکنیم که برنامه تمام صفحه شود و نوار بالایی هم نشان داده نشود.در این صورت با تستی که بنده انجام دادم عدد ۵۶۹ به عنوان طول صفحه برگردانده شد.ولی در هر صورت مشکلی برای برنامه شما پیش نمی آید.(منظور از نوار بالای صفحه نواری است که میزان شارژ باطری و آنتن دهی را نشان میدهد).

کد تمام صفحه شدن برنامه :

```
maScreenSetFullscreen(1);
```

دسته بندی که گوگل برای اندازه صفحات گوشی انجام داده در جدول ( ) آورده شده است. از آنجا که خرید دستگاه های مختلف بسیار پر هزینه است شما میتوانید با استفاده از AVD میتوانید Emulator های مختلف بسازید که در ابعاد صفحه نمایش متفاوتند.برای تعیین ابعاد شبیه ساز میتوانید از مشخصات صفحه نمایش دستگاه واقعی استفاده کنید. و هنگام اجرا در موسینک میتوانید از منوی Run گزینه ی Run configurations را انتخاب نمایید. تا تعیین کنید برنامه شما در کدام دستگاه اجرا شود. و همزمان میتوانید برنامه خود را بر روی چند دستگاه مجازی اجرا و تست کنید.

	Low density	Medium	High density	Extra high

	(120), *ldpi*	density (160), *mdpi*	(240), *hdpi*	density (320), *xhdpi*
Small **screen**	**QVGA (240x320)**		480x640	
Normal **screen**	**WQVGA400 (240x400)** **WQVGA432 (240x432)**	**HVGA (320x480)**	**WVGA800 (480x800)** **WVGA854 (480x854)** 600x1024	640x960
Large **screen**	**WVGA800** (480x800)** **WVGA854** (480x854)**	**WVGA800* (480x800)** **WVGA854* (480x854)** 600x1024		
Extra Large **screen**	1024x600	**WXGA (1280x800)**† 1024x768 1280x768	1536x1152 1920x1152 1920x1200	2048x1536 2560x1536 2560x1600

*** To emulate this configuration, specify a custom density of 160 when creating an AVD that uses a WVGA800 or WVGA854 skin.**

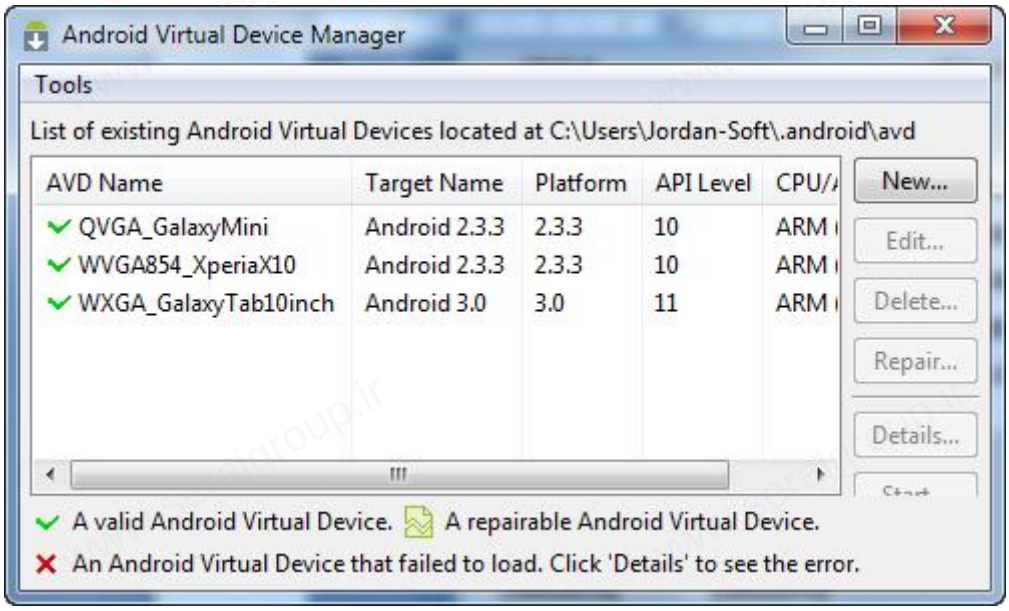
**** To emulate this configuration, specify a custom density of 120 when creating an AVD that uses a WVGA800 or WVGA854 skin.**

† This skin is available with the Android 3.0 platform

ما برای تست برنامه خود چند دستگاه مجازی با سایز و اندازه های مختلف را در نظر گرفتیم.

- سامسونگ گالاکسی مینی دستگاهی با سایز کوچک(۳ اینچ) که حداقل سایز را تعیین میکند. این دستگاه در دسته QVGA قرار میگیرد و برای ساخت آن بایستی در قسمت skin مقدار Built-in را QVGA تعیین کنید.
- XperiaX10 : دستگاهی با سایز متوسط(۴ اینچ) است.این دستگاه در دسته WVGA854 قرار میگیرد و برای ساخت آن بایستی در قسمت skin مقدار Built-in را WVGA854 تعیین کنید.
- Samsung Galaxy Tab 2 (10.1) : دستگاهی با سایز خیلی بزرگ(۱۰.۱ اینچ) است.این دستگاه در دسته WXGA قرار میگیرد و برای ساخت آن بایستی در قسمت skin مقدار Built-in را WXGA تعیین کنید. و فراموش نکنید که نوع سیستم عامل هم بایستی Android 3.0 (API level 11) باشد. چونکه به گفته گوگل این شبیه ساز فقط بر روی این نسخه از اندروید کار میکند.

در شکل ( ) این دستگاه های مجازی را مشاهده میکنید.



گالاکسی تب دستگاهی است که رزولوشن واقعی آن با رزولوشن مجازی(رزولوشنی که با استفاده از فرمول بدست آوردیم) برابر است . و دلیل آن این است که چگالی واقعی این دستگاه ۱۶۰ پیکسل در اینچ است.

برای فهم بهتر میتوانید تکه کد زیر را با پسوند html. ذخیره کنید و آن را در شبیه ساز های مختلف اجرا کنید تا اعداد و ارقام را طبق فرمول مذکور به شما بدهد.

```
<script>
        document.write(screen.width);
        document.write(",");
```

```
document.write(screen.height);
</script>
```

نکته مهمی را که در اینجا باید بدانید این است که رزولوشن واقعی گالاکسی تب ۱۰ اینچی شرکت سامسونگ 1280 x 800 px و از روی این عدد براحتی میتوان فهمید که چون عدد اول بزرگتر است(1280>800) بنابراین این دستگاه توسط کمپانی سامسونگ به صورت پیش فرض برای حالت افقی(landscape) ساخته شده است و شما بایستی توانایی تشخیص این مدل دستگاه ها را هم داشته باشید. برای تشخیص میتوانید از موسینک استفاده کنید.چونکه با تستی که من انجام دادم متوجه شدم که جاوا اسکریپت عدد 1280*800 را در برنامه فوق در خروجی مرورگر نشان داد! و از روی این عدد نمیتوان فهمید که گوشی در چه وضعیتی از نظر Orientation قرار دارد.به همین دلیل برای تشخیص Orientation دستگاه بهتر است از موسینک استفاده کنید. و خود موسینک در برنامه ای که در آدرس MoSync\examples\cpp\ScreenOrientation قرار دارد این کار را انجام داده است و شما میتوانید از این برنامه کمک بگیرید. خروجی برنامه را در شکل های ( ) و ( ) مشاهده میکنید.


Portrait 240 320
1 ∞
2 abc
3 def
4 ghi
5 jkl
6 mno
7 pqrs
8 tuv
9 wxyz
C
*
0
#

برای چرخش شبیه ساز اندروید میتوانید از دکمه ترکیبی CTRL+F11 استفاده کنید.(از کنترل سمت چپ استفاده کنید).

برای چرخش شبیه ساز موسینک (MoRE) میتوانید از دکمه F4 استفاده کنید.(از کنترل سمت چپ استفاده کنید).

در برنامه ScreenOrientation اولین کاری که انجام شده است این است که در سازنده کلاسی که از MAUtil::Moblet مشتق شده است نوع برنامه را از نظر Orientation داینامیک تعریف کرده است و یعنی برنامه برای هر دو وضعیت افقی و عمودی کار میکند.برای این کار از تابع سیستمی maScreenSetOrientation() استفاده کرده است.

int maScreenSetOrientation (int *orientation*)

این تابع تا حال حاظر فقط برای اندروید پیاده سازی شده است و بر روی گوشی های اندرویدی کار میکند.اگر این تابع کارش را با موفقیت انجام ندهد مقدار بازگشتی آن منفی است.

یک پارامتر از نوع عدد صحیح دارد که شما میتوانید هم از عدد استفاده کنید و هم از ثوابتی که در زیر میبینید استفاده کنید :

```
#define SCREEN_ORIENTATION_LANDSCAPE  1
#define SCREEN_ORIENTATION_PORTRAIT  2
#define SCREEN_ORIENTATION_DYNAMIC  3
```

عدد ۳ معادل نوشتن مقدار SCREEN_ORIENTATION_DYNAMIC است و باعث میشود برنامه به سنسور گوشی به عنوان یک رویداد گوش بدهد و با تغییر

Orientation گوشی ، جهت برنامه هم تغییر وضعیت بدهد و در دو مد افقی و عمودی کار کند.

عدد ۱و۲ برنامه ما را به طور مطلق افقی یا عمودی میکنند.یعنی برنامه در یک مد کار میکند. و انگار که برنامه قفل شده باشد. حالت پیشفرض ۲ است و برنامه فقط در وضعیت عمودی کار میکند.

maScreenSetOrientation(SCREEN_ORIENTATION_DYNAMIC);

سپس بایستی ابعاد صفحه نمایش را گرفت :

```
//Get the screen size.
MAExtent extent = maGetScrSize();
```

MAExtent **maGetScrSize** (void)

مقدار بازگشتی تابع maGetScrSize() ، MAExtent است. MAExtent از نوع typedef int MAExtent است و به نوعی تغییر نوع داده شده است که دو مقدار عددی را برای عرض و طول صفحه در خود ذخیره میکند. و به طور کلی اندازه صفحه نمایش را نگهداری میکند.

سپس برای تشخیص رویداد « تغییر Orientation »از تابع زیر استفاده شده است.

void customEvent(**const** MAEvent& event)

این تابع رویداد های خاصی که مربوط به گوشی هستند مثل تغییر Orientation را شناسایی و کنترل میکند.ما با جزییات تابع کاری نداریم و فقط با واسط کاربری آن کار داریم تا عملیات مورد نظر خود را انجام دهیم و با جزییات پیاده سازی و نحوه عملکرد آن کاری نداریم.

سپس با شرط زیر بررسی کردیم که رویداد مورد نظر انجام شده است تا رابط کاربری طبق چرخش تنظیم شود(برای مثال تابع rotateUI فراخوانی شود) :

```
if (event.type == EVENT_TYPE_SCREEN_CHANGED)
{
        rotateUI();
        …
}
```

به جای عبارت EVENT_TYPE_SCREEN_CHANGED میتوان از عدد ۲۱ استفاده کرد. چون این عبارت یک ثابت عددی است.

تا اینجا ابعاد صفحه نمایش را در متغیری به نام extent که از نوع MAEXTENT است ذخیره کرده ایم . و هنگامی که رویداد تغییر Orientation صورت میگیرد برنامه ما متوجه میشود. اما برای تشخیص حالت افقی یا عمودی بودن باید این کار را انجام بدهیم

که اگر پهنا بیشتر از طول بود در وضعیت افقی قرار داریم و اگر طول بیشتر از پهنا بود در وضعیت عمودی و طبق این وضعیت ها میتوان یک عملیات خاص را انجام داد.

```
if (EXTENT_X(extent) > EXTENT_Y(extent)) { // Landscape
        ...
}
else {  // Portrait
        ...
}
```

مشکلی که این روش دارد این است که از این روش نمیتوانید برای تعیین Orientation برنامه هایی که با HTML5 و جاوا اسکریپت نوشته اید استفاده کنید!

برنامه های hybrid یا همان برنامه هایی که UI آنها با استفاده از HTML5 ساخته شده است ، رویداد های متفاوتی نسبت به برنامه هایی دارند که توسط C++ ایجاد شده اند. برنامه شما در اولین اجرا میتواند Orientation خود را مشخص کند.چون که در اولین اجرا سازنده کلاسی که از WebAppMoblet ارث بری دارد و شروع کننده برنامه است میتواند رویداد ها را کنترل کند ، ولی بعد از لود شدن برنامه رویداد ها را HTML5 و جاوا اسکریپت مدیریت (Handling) میکنند . به همین خاطر شما در HTML5 باید رویداد تغییرOrientation را بنویسید و نه در فایل main.cpp. تنها کاری که شما در

فایل main.cpp انجام میدهید این است که کد زیر را در سازنده کلاس فوق الذکر مینویسید تا به برنامه مجوز داینامیک بودن را بدهید :

```
maScreenSetOrientation(SCREEN_ORIENTATION_DYNAMIC);
```

در فایل HTML میتوانید از jQuery Mobile استفاده کنید. این کتابخانه جاوا اسکریپتی رویداد های ویژه ای برای موبایل های لمسی (مثل رویداد های tap, taphold,swipe, orientationchange, scrollstart, scrollstop) دارد. ولی ما اینجا ترجیح میدهیم از این کتابخانه استفاده نکنیم و از HTML5 برای این کار استفاده نمیکنیم(اگر چه تفاوتی هم نمیکند و شما با هر کدام که راحت تر هستید میتوانید کار کنید).

مثال زیر به صورت اتوماتیک طبق چرخش دستگاه برنامه را از نظر ابعاد تنظیم میکند.

توضیح :

```
<body onorientationchange="rotateUI()">
```

کد بالا باعث فراخوانی تابع جاوا اسکریپت *rotateUI()* میشود.

در این تابع میتوان مقدار window.orientation را بررسی کرد و این ویژگی دارای ۴ مقدار زیر است :

- ۹۰ : به منی چرخش دستگاه به سمت چپ و قرارگیری آن در حالت افقی.*(landscape)*
- ۰ : حالت صفر ، حالت اولیه / پیشفرض دستگاه است.(در موبایل ها حالت عمودی portrait ، حالت صفر است).
- 90- : به معنی چرخش دستگاه به سمت راست و قرارگیری آن در وضعیت افقی است.
- ۱۸۰ : معادل برعکس شدن(سر و ته شدن) دستگاه است.

```
<!--Create by MILAD FASHI_1391/02/18-->
```

```
<!DOCTYPE html>
<html>
<head>
	<script src=" js/jquery-latest.js"></script>
	<script src=" js/jquery.mobile-1.1.0-rc.1.js"></script>
	<script src=" js/wormhole.js"></script>
	<link rel="stylesheet" href=" js/jquery.mobile-1.1.0-rc.1.min.css"
	/>
	<title>Orientation in HTML5</title>
	<style type="text/css">
		div.background
		{
			background-color:#f9c456;
			margin:0;
		}
		img.openPanel
		{
			opacity:0.4;
			position:absolute;
			filter:alpha(opacity=60); /* For IE8 and earlier */
		}
	</style>
	<script type="text/javascript">
	//global variables
```

```
        var X;
        var Y;
        var maxX=1280;//maximum width of screen(Samsung Galaxy
        Tab2 10.1 inch width for base of calculation)
        var maxY=800;//maximum height of screen(Samsung Galaxy Tab2
        10.1 inch height for base of calculation)
                window.onload=function()
                {
                        setSize();
                }
function setSize()
{
        X=screen.width;//width of native device
        Y=screen.height;//height of native device
        //set size for multiple screen
        support(tablet,handset(smart phone),...)
        //setting background :
        $("#bg").css("width",X);
        $("#bg").css("height",Y);

        //setting openPanelButton:
        var openPanelWidth = parseInt($("#openPanel").css("width"),10);
        var widthCalc=(X*openPanelWidth)/maxX;
        $("#openPanel").css("width",widthCalc);
}
```

```
function rotateUI()
{
	if(window.orientation!=180)/*window.orientation==90 ||
	window.orientation==-90 || window.orientation==0*/
	{
		//Exchange X and Y dimension
		var temp=X;
		X=Y;
		Y=temp;
	}

			//Rearrange background
			$("#bg").css("width",0);
			$("#bg").css("height",0);
			$("#bg").animate({width:'+='+X},"fast");
			$("#bg").animate({height:'+='+Y},"fast");

		}
	</script>
</head>
<body onorientationchange="rotateUI()">
	<div id="bg" class="background">
<img id="openPanel" src="pic/openPanel.png"
class="openPanel"></img>
```

```
</body>
</html>
```

ببخشید که عکس ها شماره نداشت.فرصت نکردم شماره بزنم.به خاطر کاستی ها عذر خواهی میکنم.

خواهشمندم این حقیر را از دعای خیر خویش بی بهره نسازید و اشتباهات بنده را و نظرات و پیشنهادات خود را در مورد این متن به آدرس پست الکترونیکی milad.fashi@gmail.com ارسال فرمایید.