



موسسه آموزش عالی علامه دهخدا

مستندسازی پروژه نهایی برای دریافت درجه کارشناسی

عنوان:

بازی آندروید: تست حافظه

استاد راهنما:

جناب آقای مهندس خزایی

دانشجو:

محمدامین امینی

۹۱۱۱۵۲۰

شهریور ۱۳۹۳



تقدیم به

تمام کسانی که در این راه مرا حمایت کردند ؛ به خصوص استاد گرامی ،

جناب آقای مهندس خزایی.

چکیده:

این مستندسازی برای پروژه ی یک بازی آندرویدی است. این پروژه با زبان جاوا در محیط اکلپس برای سیستم عامل آندروید ساخته شده است.

به همین منظور در این مستند سازی ابتدا درمورد سیستم عامل آندروید و سپس درباره ی زبان برنامه نویسی جاوا و محیط اکلپس توضیحاتی مختصر داده شده است و سپس کدهای ساخت بازی قرار داده شده اند.

حداقل نسخه ی سیستم عامل برای اجرای این بازی آندروید "کندوی عسل" و نسخه هدف بازی "آبنبات ژله ای" میباشد.

در این بازی شما با نه خانه روبرو هستید که با تکیه بر شانس و حافظه باید عکس های مشابه را انتخاب کنید و حواستان به بمب باشد. اگر هر چهار عکس را بتوانید با مشابه های خود انتخاب کنید بازی را برده اید.

کلمات کلیدی : بازی ، سیستم عامل ، آندروید ، زبان برنامه نویسی ، جاوا.

فهرست فصل ها

۱- مقدمه	۱
۲- سیستم عامل آندروید	۴
۱-۲- آندروید چیست؟	۵
۱-۱-۲- معنای آندروید	۵
۲-۱-۲- تعریف آندروید	۵
۳-۱-۲- تاریخچه	۵
۴-۱-۲- کپی رایت	۷
۵-۱-۲- لوگو	۸
۶-۱-۲- ویژگی ها	۸
۷-۱-۲- فرمت فایل های پشتیبانی شده	۹
۲-۲- آرت	۹
۳-۲- محیط برنامه نویسی	۹
۴-۲- نسخه های آندروید	۱۰
۱-۴-۲- نگاهی به نسخه های مختلف آندروید	۱۱
۴-۴-۲- پراکندگی نسخه های مختلف آندروید	۱۵
۵-۲- سندباکس	۱۵
۶-۲- آندروید مارکت	۱۶
۷-۲- گوگل تی وی	۱۸
۸-۲- معماری آندروید	۱۹
۱-۸-۲- تشریح لایه های مختلف معماری	۲۰
۹-۲- رابطه آندروید و جاوا	۲۲
۱۰-۲- ماشین مجازی دالویک	۲۲
۱۱-۲- بازی های آندروید	۲۳
۱۲-۲- رده های سنی بازی آندروید	۲۳
۱۳-۲- ورژن های مختلف بازی های آندروید	۲۳
۱۴-۲- قابلیت های بازی آندروید	۲۴
۱۵-۲- معایب بازی های آندروید	۲۴
۳- جاوا	۲۵
۱-۳- تاریخچه	۲۶
۲-۳- اهداف اولیه	۲۸

۳-۳- نسخه ها	۲۸
۳-۴- برنامه های جاوا و اپلت ها	۲۹
۳-۵- Net. رقیب جاوا؟	۲۹
۳-۶- بین Net. و جاوا کدام رو انتخاب کنیم؟	۳۰
۳-۷- خط مشی جاوا	۳۱
۳-۸- پیاده سازی	۳۲
۳-۹- اداره خودکار حافظه.	۳۳
۳-۱۰- گرامر.	۳۴
۳-۱۱- توزیع های جاوا.	۳۴
۳-۱۲- کتابخانه های جاوا.	۳۵
۳-۱۳- ویرایش ها.	۳۵
۳-۱۴- ایرادات مطرح شده.	۳۶
۳-۱۵- پاسخ به ایرادات.	۳۷
۳-۱۶- یک اشتباه متداول!	۳۸
۳-۱۷- برنامه نویسی برای آندروید.	۳۹
۳-۱۷-۱- نحوه نصب SDK.	۳۹
۳-۱۷-۲- AVD.	۴۱
۳-۱۷-۳- نصب برنامه شبیه ساز در کامپیوتر.	۴۳
۳-۱۸- Eclipse و برنامه نویسی با آن.	۴۵
۳-۱۸-۱- برنامه نویسی آندروید چگونه است؟	۴۵
۳-۱۸-۲- Eclipse چیست؟	۴۵
۳-۱۸-۲-۱- معماری.	۴۵
۳-۱۸-۲-۲- سکوی غنی مشتری.	۴۶
۳-۱۸-۲-۳- تاریخچه.	۴۶

۴۷.....نسخه ها. ۳-۱۸-۲-۴

۴۷.....افزونه ها. ۳-۱۸-۲-۵

۴۸.....شروع کار. ۳-۱۸-۲-۶

۵۱.....نمونه هایی از برنامه های جاوا. ۳-۱۸-۲-۷

۵۳.....کلاس های خاص. ۳-۱۸-۲-۸

۴ - معرفی پروژه ۵۶

۵ - توضیحات فنی و کدها ۶۱

فهرست جدول ها

جدول ۱ - نسخه های آندروید ۱۰

جدول ۲ - نسخه های جاوا ۲۸

جدول ۳- نسخه های Eclipse ۴۷

فهرست شکل ها

شکل ۱-لوگوی آندروید	۸
شکل ۲-معماری سیستم عامل آندروید	۱۹
شکل ۳-لوگوی جاوا	۲۶
شکل ۴-SDK Manager	۴۰
شکل ۵-مراحل نصب شبیه ساز	۴۱
شکل ۶-شبیه ساز آندروید	۴۲
شکل ۷-کدهای نصب شبیه ساز	۴۴
شکل ۸-نرم افزار اکلیپس	۴۸
شکل ۹-محیط برنامه نویسی اکلیپس	۴۹
شکل ۱۰-کادر ورود اطلاعات برای ساختن اپ	۵۰
شکل ۱۱-صفحه ی خانه ی بازی	۵۷
شکل ۱۲-صفحه ی اصلی بازی	۵۸
شکل ۱۳-هدف بازی	۵۹
شکل ۱۴-پیغام پایان بازی	۶۰
شکل ۱۵-محتویات پروژه	۶۲

فصل اول

مقدمه

توسعه ی روزافزون بازار تلفن های همراه و پیشرفت چشمگیر ابزارها و برنامه های کاربردی مختلف در طی دهه های گذشته ، باعث شده است که تلفن همراه به عنوان جزیی جدایی ناپذیر از زندگی افراد تبدیل شود. این مساله تا حدی است که امکانات بعضی از نمونه های تلفن همراه با کامپیوترهای امروزی برابری میکند. از یک طرف، به دلیل قالب جمع و جور تلفن های همراه که امکان حمل و نقل آسان آن را فراهم میکند و از طرف دیگر، پیشرفت چشمگیری که در فشرده سازی هرچه بیشتر مدارهای الکترونیکی حاصل شده است، باعث شده تا بازار تلفن های همراه، هرروزه شاهد ظهور مدل های جدید با قابلیت های بهتر نسبت به مدل های قبلی باشد. این پیشرفت ها فقط محدود به توسعه های سخت افزاری نمی شود و توسعه های نرم افزاری را نیز شامل میشود. به همین دلیل است که شرکت های بزرگی همچون گوگل نیز در برابر وسوسه ورود به این بازار پروتق نمیتوانند مقاومت کنند..

سیستم عامل اندروید که از طرف گوگل عرضه شده و به صورت منبع باز منتشر شده است، راه را برای توسعه هرچه بیشتر این بازار فراهم آورده است. اندروید به سرعت در حال رشد و پیشرفت است و به طور سالانه در حدود ده هزار برنامه کاربردی برای آن معرفی میشود که این مساله به وجود مرجعی برای توسعه و ایجاد برنامه های کاربردی مناسب نیاز دارد. به طور کلی، یادگیری سیستم عامل اندروید بسیار ساده است و گوگل کتابخانه های مختلفی را برای پیاده سازی هرچه بهتر برنامه های کاربردی مختلف عرضه کرده است. تمرکز اصلی این مستند سازی بر قابلیت توسعه ی برنامه های کاربردی اندروید گذاشته شده است.

برای فهم این مستندسازی آشنایی مختصر با محیط برنامه نویسی و زبان های برنامه نویسی لازم است. هرچند که در ادامه به معرفی سیستم عامل اندروید و زبان برنامه نویسی جاوا و محیط توسعه ی Eclipse پرداخته ام ؛ اما بطور کلی ، آشنایی نسبی با ساختارهای برنامه نویسی شی گرا و رویداد محور میتواند در هرچه بهتر فهمیدن این مقاله موثر باشد.

بطور خلاصه این مقاله بصورت زیر ادامه می یابد:

در فصل دوم، به معرفی سیستم عامل اندروید و مروری به جنبه های مختلف آن پرداخته میشود. این فصل حاوی برنامه نویسی نیست. در ادامه این فصل نسخه های مختلف این سیستم عامل و سپس تاریخچه ای کوتاه از بازی های اندروید و حواشی و مزایا و معایب آنها را خواهیم خواند.

در فصل سوم ، معرفی ای از زبان برنامه نویسی جاوا ، چگونگی و چرایی پدید آمدن آن را خواهیم داشت. در ادامه ی این فصل ، محیط توسعه ی Eclipse و کار با آن را توضیح خواهیم داد.

حالا اندکی برای برنامه نویسی و تولید یک برنامه کاربردی آماده ایم.

در فصل چهارم ، توضیحی کوتاه درمورد برنامه ی کاربردی تولیدی خودم ، که یک بازی میباشد خواهیم داد ؛ و در فصل پنجم ، شاهد توضیحات فنی و کدهای برنامه نویسی این برنامه هستیم. در این فصل ، که فصل آخر مستندسازی است، سعی برآن شده تا توضیحات و کدها بصورت آموزشی برای خواننده نوشته شود و امید به آن است که قدمی هرچند کوتاه برای شما خواننده ی عزیز در مسیر بلند برنامه نویسی آندرویدی باشد.

موفق باشید.

سیستم عامل آندروید

۲-۱-آندروید چیست؟

۲-۱-۱-معنای آندروید:

پیش از ورود به اطلاعات مربوط به آندروید، نخست به نام آن می‌پردازیم. بنابر ترجمه دیکشنری کمبریج، آندروید 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 آندروید را آدم آهنی یا آدم ماشینی دانست.

۲-۱-۲-تعریف آندروید:

آندروید^۱ نام یک سیستم عامل موبایل است که توسط شرکت گوگل توسعه داده می‌شود. این سیستم عامل اوپن سورس است و برپایه هسته لینوکس بنا شده است. آندروید بر خلاف سیستم عامل iOS آیفون که فقط پردازنده های ARM را پشتیبانی میکند، بر روی انواع مختلفی از پردازنده ها (ARM, MIPS, Power 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 PC نیز با استفاده از این سیستم عامل به بازار ارائه شده اند.

۲-۱-۳-تاریخچه:

تاسیس آندروید و کار گروهی برای ساخت آن به ماه اکتبر ۲۰۰۳ میلادی بازمی‌گردد؛ زمانی که اندی رابین ، ریچ ماینر ، نیک سیرز و کریس وایت تصمیم به ساخت سیستم عاملی برای گوشی های هوشمند گرفتند که امکانات بیشتری به کاربران خود بدهد.

اولین قدم در تولید سیستم عامل آندروید زمانی برداشته شد که گوگل در سال ۲۰۰۵ شرکت ناشر اولیه ی آندروید را در پالو آلتوی کالیفرنیا خریداری کرد ، شرکت کوچک آندروید در زمینه تولید نرم افزار و برنامه های کاربردی برای تلفن های همراه فعالیت می کرد . اندی رابین مدیر ارشد اجرایی این شرکت پس از پیوستن آندروید به گوگل به سمت قائم مقام مدیریت مهندسی این شرکت و مسئول پروژه آندروید در گوگل منصوب شد. در واقع می توان رابین را پایه گذار آندروید دانست ، چرا که وی علاوه بر اینکه ایده تولید آندروید

را در شرکت کوچک خود پرورش داد ، در سمت مدیر این پروژه در شرکت گوگل توانست ایده خود را پیاده سازی کند و سیستم عامل اندروید را با نام شرکت کوچک پیشین خود روانه بازار نماید.

تیم آندروید به رهبری رابین فعالیت خود را برای تولید platform مبتنی بر هسته لینوکس آغاز کردند. درز اخباری از فعالیت های این تیم به خارج از گوگل ، سبب بروز شایعاتی مبتنی بر تمایل گوگل به تولید تلفن های همراه در اواخر سال 2006 گردید . با اعلام زمان کنفرانس خبری شرکت گوگل در نوامبر سال 2007 ، تمامی رسانه ها 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 درمورد ورود صدها تلفن همراه در سال های پیش رو بود که توسط شرکت های مختلف تولید می شد.

گوگل به شرکت های تولیدکننده تلفن همراه ، وعده ی سیستم عامل انعطاف پذیر و قابل به روزرسانی را داده بود به همین منظور گوگل شرکت های بسیاری را با خود برای رسیدن به این هدف همراه کرده بود.

"اتحادیه گوشی باز" یا Open Handset Alliance در روز پنج نوامبر 2007 اعلام موجودیت کرد.

سی و چهار شرکت فعال در زمینه تولید نرم افزار ، تولید گوشی های تلفن همراه ، اپراتور تلفن همراه و تولید کننده نیمه رساناها و پردازنده های تلفن همراه اعضای موسس این اتحادیه بودند . در میان نام های مشهور در بین اعضای موسس ، شرکت هایی چون Samsung ، LG ، Motorola ، HTC ، T-Mobile ، eBay ، intel ، Nvidia و ... و البته Google به چشم می خوردند . اریک اشمیت مدیر ارشد اجرایی گوگل در این مراسم گفت : ((اعلام امروز بسیار جاه طلبانه تر از معرفی تنها "یک" تلفن همراه گوگلی است که در چند هفته اخیر توسط رسانه ها پیش بینی شده بود . از دیدگاه ما پلتفرمی که ما ارائه کرده ایم ، هزاران تلفن همراه گوناگون را به بازار روانه خواهد کرد.))

نخستین گوشی مبتنی بر آندروید 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 بیست و دوم اکتبر 2008 تولید شد ، در بازارهای مختلف به نام های HTC Dream ، T-Mobile G1 ، Era G1 به بازار عرضه گردید . سپس گوگل کدهای مربوط به آندروید را به عنوان یک پروژه ی منبع باز منتشر کرد.

پس از وقایع مذکور گوگل به طور رسمی از Android Open Source Project رونمایی کرد . پروژه ای که به گفته ناشرین آن هدفش ارتقای کیفیت تلفن های همراه و تجربه ای که کاربران از تلفن های همراه خود دارند است . AOSP همچنین با رونمایی خود ، از مفهوم «دستگاه های مبتنی بر آندروید» نیز پرده برداری کرد.

دستگاه هایی که قابلیت پشتیبانی و اجرای نرم افزارهای نوشته شده از سوی شرکت های Third Party را دارند ۹ دسامبر ۲۰۰۸ چهارده عضو جدید از شرکت های صنعت تلفن همراه جهان به اتحادیه گوشی باز پیوستند. در بین این نام ها باید به سونی اریکسون، اریکسون، توشیبا، ایسوز، گارمین، هواوی و آرم هولدینگز اشاره کرد. روند پیوستن شرکت های بزرگ به اتحادیه تا به امروز نیز ادامه داشته است و شرکت هایی چون ایسر، آلکاتل، لنوو، شارپ، فاکس کان، ان ای سی، کیوسرا، ان ایکس پی، اسی-اریکسون، مارول، زد تی ئی و دل نیز از جمله شرکت هایی بوده اند که به جمع پشتیبانی کنندگان اندروید پیوسته اند.

در ۳ سپتامبر ۲۰۱۳ توسعه‌دهندگان اندروید به‌طور رسمی اعلام کردند که با شرکت نستله، که از شرکت‌های مطرح صنعت شکلات‌سازی جهان می‌باشد، همکاری خواهند کرد. در همین راستا نگارش ۴.۴ سیستم‌عامل اندروید، کیت‌کت نام گرفت. کیت‌کت از مارک‌های معروف شکلات است که توسط شرکت نستله تولید می‌شود.

۲-۱-۴- کپی‌رایت و حق امتیاز:

حق امتیاز اندروید به صورت اپن سورس بر اساس حق امتیاز آپاچی^۱ 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. با اینکه سعی می‌شود تا اکثریت قسمت‌های این سیستم عامل بر اساس همین مجوز ارائه گردد، استثناهایی نیز وجود دارد. برای مثال هسته لینوکس موجود در این سیستم عامل با پروانه عمومی همگانی گنو نسخه ۲^۲ منتشر شده است.

1. Apache License

2. GPLv2

۲-۱-۵- لوگوی اندروید:

طراح لوگوی مشهور اندروید “آیرینا بلاک” است. سه سال پیش از آیرینا بلاک و تیم طراحی گوگل تقاضا شد تا لوگویی برای اندروید طراحی کنند که به سرعت با کاربر ارتباط برقرار کند و به آسانی قابل شناسایی باشد؛ همچنین به بلاک گفته شد که این لوگو باید حتماً تصویری از ربات باشد چرا که اندروید به معنای روبات است. آیرینا پس از مطالعه زیاد در مورد اسباب‌بازی‌ها و شخصیت‌های فانتزی و تخیلی در آخر تصمیم گرفت لوگوی اندروید را از یک منبع عجیب الهام بگیرد: دستشویی! هر کسی علامت روی در دستشویی‌ها را دیده و الهام از این علامت‌ها می‌تواند باعث شود در یک نگاه لوگوی اندروید شناخته شود. یک نکته جالب دیگر در مورد لوگوی اندروید این است که گوگل برخلاف دیگر شرکت‌ها که سعی در حفاظت از لوگویشان داشتند تصمیم گرفت تا لوگو را به صورت منبع باز قرار دهد تا هر کس بتواند آن را به دلخواه خودش تغییر دهد. گوگل در این باره می‌گوید: “ما تصمیم گرفتیم این لوگو می‌تواند یک لوگوی اشتراکی و تعاملی باشد که هر کس در دنیا بتواند آن را تغییر دهد. این تصمیم بسیار شجاعانه بود.” شاید با اپلیکیشن Androidify و یا اسباب‌بازی‌ها و Action Figure های کوچک اندرویدی بامزه آشنا باشید؛ همه این‌ها به لطف آزاد بودن لوگوی اندروید امکان‌پذیر شده است.



شکل ۱- لوگوی اندروید

۲-۱-۶- ویژگی‌های اندروید:

اندروید تمامی تکنولوژی‌های اتصال^۱ شامل GSM/EDGE, CDMA, EV-DO, UMTS, ، بلوتوث و وای-فای را پشتیبانی می‌کند.

اندروید از فرمت‌های مختلف فایل‌های مالتی مدیا مثل MPEG, MP3, AAC, AMR, JPEG, PNG, GIF پشتیبانی می‌کند.

اندروید برای ارسال پیغام‌های متنی یا همان اس ام اس از فرم‌های SMS, MMS و XMPP پشتیبانی می‌کند.

مرورگر موجود در اندروید بر اساس فریم ورک اوپن سورس WebKit توسعه یافته‌است.

اندروید برای ذخیره داده‌ها و مدیریت بانک‌های اطلاعاتی سبک از نرم‌افزار SQLite استفاده می‌کند.

تمام برنامه‌های اندروید به زبان جاوا نوشته می‌شوند. برای اجرای برنامه‌های جاوایی روی این سیستم عامل، کدهای جاوا به کدهای Dalvik تبدیل می‌شوند و سپس روی ماشین مجازی جاوایی^۲ اجرا می‌شوند. (درمورد ماشین مجازی دالویک در ادامه توضیح خواهم داد)

1. Connectivity

2. Dalvik virtual machine

ابزارهای مختلف اندروید برای توسعه دهندگان به راحتی در دسترس است و توسط شرکت گوگل پشتیبانی می‌شوند. این ابزارها شامل کتابخانه‌ها، خطایاب، شبیه‌ساز گوشی و یک پلاگین برای اکلیپس است. اندروید از سخت‌افزارهای مختلف همچون جی پی اس و دوربین‌های متنوع پشتیبانی می‌کند. تصاویر و فایل‌های گرافیکی بوسیله OpenGL پردازش می‌شوند که کیفیت بالاتری خواهند داشت. با استفاده از تکنولوژی نسبتاً جدید شرکت Adobe با نام AIR نیز می‌توان به توسعه برنامه‌های کاربردی تحت این سیستم عامل پرداخت.

۲-۱-۷- فرمت فایل‌های پشتیبانی شده:

اندروید در حالت پیش‌فرض فایل‌های mp3, aac, ogg, amr, midi, mpeg4, wav, bmp, gif, png, jpg, را پشتیبانی می‌کند. اندروید Adobe Flash را نیز پخش می‌کند و می‌تواند فایل‌های Gif متحرک را با حرکت پخش کند. برای پخش فایل‌های جریان دار مانند صوت و ویدئو نیز می‌توانید از تگ ویدئو html5 و همچنین تکنولوژی Adobe Flash Streaming استفاده کنید. در نسخه‌های جدید اندروید، موتور جاوااسکریپت مرورگر کروم که سرعت بسیار بالایی در اجرای کدهای جاوا اسکریپت دارد به مرورگر اندروید متصل شده‌است. (در ضمن مرورگر اندروید کدهای HTML5 را پشتیبانی می‌کند)

۲-۲- آرت :

آرت ران تایم جدید اندروید است که جایگزین دالویک شده‌است. شرکت گوگل برای اولین بار در اندروید ۴.۴ آرت را در کنار دالویک قرار داد و کاربران می‌توانستند با مراجعه به تنظیمات آن را فعال کنند.

۲-۳- محیط برنامه‌نویسی اندروید:

مجموعه برنامه‌نویسی اندروید یا Android SDK شامل یک دیباگر، کتابخانه‌های اندروید، شبیه‌ساز سیستم عامل، مستندات اندروید و فایل‌های نمونه و آموزشی است که به کاربر در ایجاد برنامه‌ها کمک می‌کند. هم‌اکنون این SDK بر روی یک سیستم ۳۲ بیتی که لینوکس، ویندوز و یا mac OSX داشته باشد اجرا می‌شود. پیش‌نیازهای نصب این SDK عبارتند از JDK و Apache Ant و python 2.2.

۲-۴- نسخه های آندروید:

گوگل ویرایش های گوناگون آندروید 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 گونه ای که ویرایش های منتشر شده آندروید تا به امروز به این نام ها بوده اند:

جدول ۱- نسخه های آندروید

نسخه	نام لاتین	نام فارسی	تاریخ انتشار
1.0	Alpha	-	۲۳ دسامبر ۲۰۰۸
1.1	Beta	-	۹ فوریه ۲۰۰۹
1.5	Cupcake	کیک فنجانی	۳۰ آوریل ۲۰۰۹
1.6	Donut	پیراشکی	۱۵ سپتامبر ۲۰۰۹
2.0 و 2.1	Éclair	نان خامه ای	۲۶ اکتبر ۲۰۰۹
2.2	Froyo	ماست یخ زده	۲۰ می ۲۰۱۰
2.3	Gingerbread	نان زنجبیلی	۶ دسامبر ۲۰۱۰
3.0، 3.1 و 3.2	Honeycomb	کندوی عسل	۲۲ فوریه ۲۰۱۱
4.0	Ice Cream Sandwich	بستنی حصیری (ساندویچ بستنی)	نوامبر ۲۰۱۱
4.1	Jelly Bean	آبنبات ژله ای	ژوئیه ۲۰۱۲
4.2	Jelly Bean	آبنبات ژله ای	اکتبر ۲۰۱۲
4.3	Jelly Bean	آبنبات ژله ای	ژوئیه ۲۰۱۳
4.4	KitKat	کیت کت	اکتبر ۲۰۱۳
5.0	Lollipop	آبنبات چوبی	۵ جولای ۲۰۱۴

۲-۴-۱- نگاهی به نسخه های مختلف آندروید:

نسخه ی 1.5 یا Cupcake:

نسخه ی 1.5 آندروید در 30 آوریل 2009 منتشر شد. این نسخه مبتنی بر هسته لینوکس 2.6.27 بود. از جمله قابلیت هایی که در این ویرایش گنجانده شده بود شامل :

- امکان ضبط فیلم از طریق دوربین فیلمبرداری
 - فرستادن فیلم به سایت یوتیوب و عکس به سایت پیکاسا به صورت مستقیم از روی گوشی
 - صفحه کلید مجازی با قابلیت پیش بینی کلمات وارد شده
 - پشتیبانی از پخش استریم موسیقی از طریق بلوتوث و کنترل پخش موسیقی یا ویدیو از طریق بلوتوث
 - قابلیت اتصال اتوماتیک به دستگاه های بلوتوث
 - امکان شخصی سازی صفحه اصلی با استفاده از ویجت ها و یا پرونده های شخصی
- جابجایی انیمیشنی تصاویر به هنگام عوض شدن صفحات

اندروید نسخه 1.6 یا Donut:

در سپتامبر 2009 اندروید نسخه 1.6 را منتشر کرد. این نسخه مبتنی بر هسته لینوکس 2.6.29 بود و قابلیت های زیر را به اندروید افزود:

- بهبود در سرویس Android Market
- رابط کاربری یکپارچه برای دوربین عکسبرداری ، دوربین فیلمبرداری و گالری تصاویر
- امکان انتخاب چند عکس برای پاک کردن در منوی گالری
- به روزرسانی ویژگی جست و جوی صوتی
- به روزرسانی ویژگی جست و جوی با قابلیت جست و جوی در موارد نشانه گذاری شده، تاریخچه، اسامی و وب از صفحه اصلی
- پشتیبانی از تکنولوژی به روز شده CDMA/EVDO ، X802.1 ، VPN و موتور Text To Speech
- پشتیبانی از رزولشن WVGA برای صفحه نمایش

اندروید نسخه 2 و 2.1 یا Eclair:

هر دو نسخه 2 و 2.1 اندروید مانند نسخه 1.6 مبتنی بر هسته لینوکس 2.6.29 طراحی شده بودند. اهم امکانات اضافه شده در این نسخه به شرح زیر است :

- سرعت سخت افزاری بهبود یافته
- ویژگی چندلمسی
- پشتیبانی از رزولشن های بیشتر برای صفحه نمایش
- رابط کاربری به روز رسانی شده
- مرورگر اینترنتی با قابلیت پشتیبانی از HTML5
- دفترچه تلفن به روز رسانی شده
- Google Map نسخه 3.1.2
- پشتیبانی از Microsoft Exchange
- افزوده شدن امکان فلش داخلی برای دوربین
- افزوده شدن زوم دیجیتال دوربین
- به روزرسانی صحنه کلید مجازی
- پشتیبانی از بلوتوث نسخه 1.2
- اضافه شدن قابلیت Live Wallpaper
- اضافه شدن امکان ارسال فایل با استفاده از بلوتوث

نسخه 2.2 یا Froyo:

اندروید نسخه 2.2 را در می 2010 معرفی شد. این ویرایش اندروید مبتنی بر هسته لینوکس نسخه 2.6.32 است و قابلیت های زیر به آن اضافه شده است :

- افزایش سرعت سیستم عامل ، حافظه و عملکرد سیستم بین 2 تا 5 برابر نسخه 2
- افزایش سرعت اجرای برنامه های کاربردی با استفاده از تکنیک های JIT
- اضافه شدن موتور نرم افزار مرورگر اینترنتی مبتنی بر موتور Webkit که با موتور Chrome JavaScript همراه شدن است که سرعت بسیار بالایی در اجرای کدهای JavaScript دارد.
- افزایش پشتیبانی از Microsoft Exchange با قابلیت هایی چون سیاست حریم شخصی به روز شده ، همسان سازی تقویم و ...

- Android Market به روز شده با قابلیت به روزرسانی خودکار برنامه های کاربردی
- شماره گیری صوتی و انتقال دفترچه تلفن از طریق بلوتوث
- امکان نصب برنامه های کاربردی روی حافظه های جانبی
- پشتیبانی از فلش نسخه 10.1
- بهبود عملکرد دوربین در حالت های عکس و فیلمبرداری

اندروید نسخه 2.3 یا Gingerbread:

اندروید نسخه 2.3 یا نان زنجبیلی در دسامبر 2010 معرفی شد. امکانات اضافه شده در این نسخه به شرح زیر هستند:

- بهینه سازی و تغییر در طراحی رابط کاربری به همراه ساده تر و سریعتر کردن آن
- پشتیبانی از صفحه های بزرگتر و رزولشن های بیشتر WXGA و بیشتر
- پشتیبانی از تماس اینترنتی یا تماس های VoIP
- افزودن قابلیت NFC یا همان ارتباط حوزه نزدیک
- بهبود در نمای گرافیکی اندروید و صفحه ی کیبورد لمسی و قابلیت های دیگر

اندروید نسخه 3 و 3.1 و 3.2 یا Honeycomb که مخصوص Tablet ها می باشد:

- بهبود در بسیاری از ویژگی های گرافیکی اندروید
- قابلیت پشتیبانی از پردازنده های چند هسته ای
- پشتیبانی از USB Host Mode برای انتقال مستقیم اطلاعات از دوربین و دیگر دستگاه ها
- اضافه شدن برنامه های Google Books و Google Movies
- بهینه سازی آن برای صفحات نمایش عریض

اندروید نسخه 4.0 یا Ice Cream Sandwich:

اندروید نسخه 4.0 یا بستنی حصیری در اکتبر 2011 معرفی شد . در این نسخه تغییرات عمده ای در سیستم عامل نسبت به نسخه های پیشین به وجود آمد. رابط کاربری به طور کلی بازنویسی و دگرگون شد و همه چیز از نو بهینه سازی شده است.

نسخه 4.0 اندروید را میتوان متحد کننده ی دو نسخه ی 2.3.x و نسخه ی 3.x این سیستم عامل دانست که بدین ترتیب از این پس نسخه های سیستم عامل اندروید ویژه گوشی های هوشمند و تبلت ها یکسان خواهد بود.

ضمن اینکه ، امکانات جدید زیادی به اندروید چهارم اضافه شد.قابلیت های تازه و تغییرات شامل :

- امکان استفاده از دکمه های مجازی در رابط های کاربری به جای استفاده از دکمه های فیزیکی در پایین گوشی.

- قرار دادن ویجت ها در Tab هایی مشابه با لیست برنامه ها

- ساخته شدن پوشه ها به آسانی و با کشیدن و انداختن

- نرم افزار جدید برای تلفن

- امکان زوم در تقویم

- اضافه کردن امکان جستجوی آفلاین در میان رایانامه ها در Gmail و خطی کردن پیش نمایش رایانامه ها

- امکان گرفتن Screenshot با نگه داشتن دکمه ی پاور و دکمه ی صدا

- بهبود سیستم غلط یابی صفحه کلید

- دسترسی مستقیم از صفحه اصلی گوشی به برنامه ها

- بهبود Copy و Paste کردن

- بهبود سیستم تشخیص صدا

- خارج کردن سیستم از حالت قفل با سیستم تشخیص چهره

- پشتیبانی مرورگر جدید از Tab که میتواند تا 16 تب همزمان را پشتیبانی کند.

- سیستم Font جدید که گوگل آن را " روباتو " نامیده است که برای صفحه نمایش با رزولشن بالا بهینه سازی شده است.

- امکانات نظارت و مدیریت بر میزان مصرف دیتا و مشخص کردن سقف مصرف دیتا

- امکان توقف برنامه هایی که در پس زمینه از اینترنت استفاده کردند.

- بهبود برنامه دوربین ، رساندن تاخیر Shooter به صفر و امکان Zoom هنگام فیلمبرداری

- برنامه ویرایش عکس

- گالری مدیریت تصاویری جدید بر اساس موقعیت و افراد

- برنامه شبکه های اجتماعی به نام People ادغام شده با Google+

- اندروید Beam امکانی است که با کمک NFC اجازه میدهد اطلاعاتی مانند وب سایت ها ، دفترچه تماس ، آدرس ، فیلم و ... را سریعاً به فرد دیگری انتقال داد.

1. Drag & Drop

- موتور قدرتمند ورود صوتی متن (با استفاده از این موتور کاربران حتی میتوانند هنگام صحبت مکث نموده و دوباره صحبت خود را از سر گیرند)
- ارائه صفحه کلید فارسی

۲-۴-۲- پراکندگی نسخه‌های اندروید:

طبق آماری که شرکت گوگل در آپریل ۲۰۱۴ منتشر کرده، میزان پراکندگی نسخ اندروید بدین صورت می‌باشد:

فروبو (نسخه ۲،۲): ۰،۷ درصد
 جینجری برد (نسخه ۲،۳): ۱۳،۶ درصد
 ایس کریم سندویچ (نسخه ۴،۰): ۱۰،۶ درصد
 جلی بین (نسخه ۴،۱ و ۴،۲ و ۴،۳): ۵۴،۲ درصد
 کیت کت (نسخه ۴،۴): ۲۰،۹ درصد

۲-۵- Sandbox :

کلیه ی نرم افزارهای اندروید در فضای ایزوله ای از سیستم عامل به نام Sandbox اجرا میشوند، فضایی که به دیگر قسمت‌ها و منابع سیستم عامل دسترسی ندارد. هنگامی که کاربر قصد دریافت یک نرم افزار از Android Market را دارد کلیه بخش هایی که آن نرم افزار قصد دسترسی به آن را دارد برای کاربر به نمایش در می‌آید. به عنوان مثال یک بازی ممکن است بخواهد قابلیت لرزش گوشی را فعال کند اما نباید به پیغام‌های ذخیره شده بر روی گوشی دسترسی پیدا کند.

علی رغم توضیحات داده شده ، باز هم نرم افزارهای جاسوسی و دزدی اطلاعات بر روی Android Market دیده شده است. به گزارش Kaspersky Lab در اگوست سال ۲۰۱۰، بر روی Android Market ، یک تروجان برای دزدی SMS ها منتشر شد که تعدادی از دستگاه‌ها را نیز آلوده ی خود کرد. گوگل خیلی سریع به این موضوع واکنش نشان داد و مسئولیت غیر فعال کردن این گونه App ها را از Android Market ها را بر عهد گرفت. با این وجود نرم افزارهای آنتی ویروس بسیاری نظیر AVG Kaspersky ، F-Secure و McAfee برای اندروید منتشر شده اند تا امنیت اطلاعات کاربران را تامین کنند.

۶-۲- آندروید مارکت:

آندروید مارکت سرویس فروش نرم‌افزارهای کاربردی برای گوشی‌های آندروید است. یک برنامه کاربردی ویژه آندروید مارکت به صورت از پیش بارگذاری شده بر روی گوشی‌های آندروید نصب گردیده و به کاربران امکان می‌دهد نرم‌افزارهای مورد نیاز خود را خریداری و دانلود کنند. البته تمامی نرم‌افزارهای موجود در آندروید مارکت فروشی نیستند بلکه بیش از نیمی از نرم‌افزارهای موجود در آندروید مارکت به صورت رایگان عرضه می‌شوند و از این نظر آندروید 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 آندروید 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 آندروید از سراسر جهان می‌توانند برنامه‌های کاربردی رایگان خود را از طریق آندروید 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، سوئیس، 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 آندروید مارکت را دارند! این در حالی است که گوشی‌های مجهز به آندروید 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 آندروید 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 آندروید 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 آندروید را به شدت محدود ساخته و تمامی دسترسی کاربران را از طریق نرم‌افزار آندروید 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 آندروید از وب سایت رسمی آندروید 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 آندروید 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 آندروید مارکت از طریق وب سایت‌های خود نیز در اختیار کاربران قرار می‌دهند.

علاوه بر این برخی وب‌سایت‌ها اقدام به جمع‌آوری و در اختیار قرار دادن برنامه‌های کاربردی پرتعداد آندروید 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 آندروید به ویژه برنامه‌های اصلی آن که توسط خود گوگل طراحی شده‌اند، مانند نقشه‌های گوگل یا برنامه گگلز تنها از طریق آندروید مارکت قابل دسترسی هستند. پس اگر می‌خواهید به برنامه‌های کاربردی اصلی آندروید دسترسی داشته باشید، باید از آندروید مارکت نصب شده بر روی گوشی خود استفاده کنید.

۷-۲- آندروید و گوگل تی وی:

خب اگر فکر کرده‌اید که کار شما با آندروید تمام شده است سخت در اشتباه هستید. اگر کار شما هم با آندروید تمام شده باشد، کار آندروید 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 آندروید 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 آندروید خبر دادند.

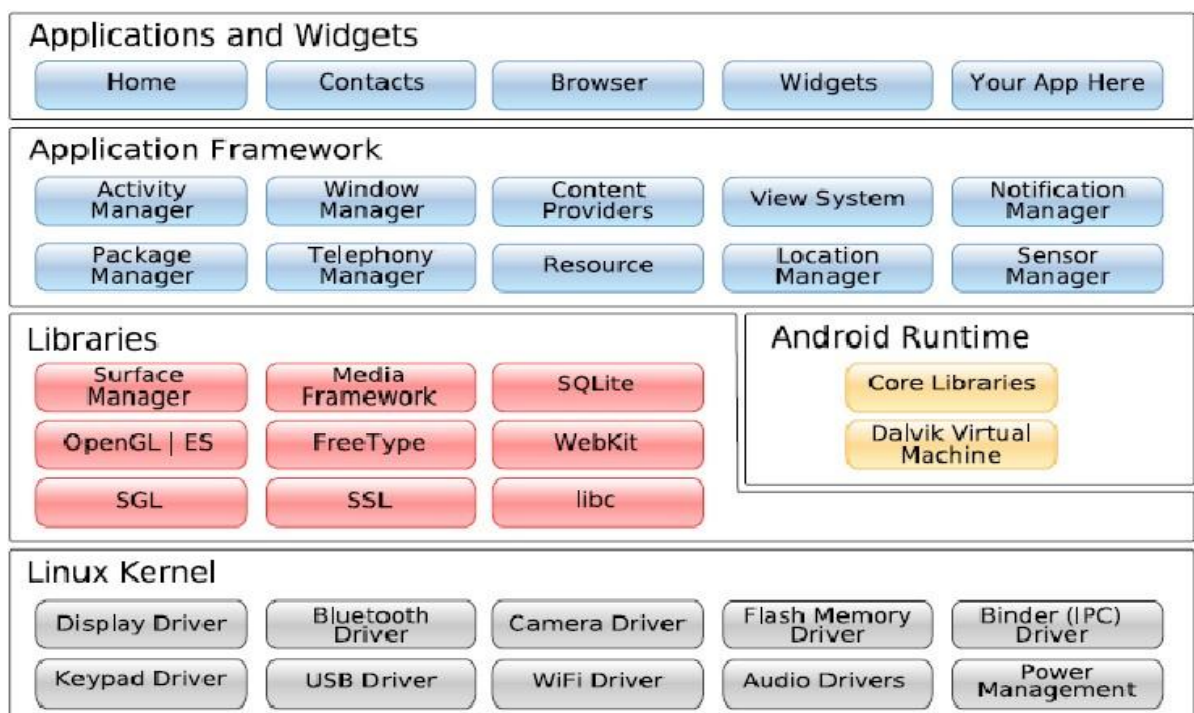
تلویزیون‌هایی که به طور بی‌سیم به اینترنت متصل می‌شوند و علاوه بر اینکه امکان اتصال به شبکه‌های آنلاین پخش فیلم را دارند، از برنامه‌های کاربردی که برای نصب بر روی این تلویزیون‌ها تهیه می‌شوند نیز بهره خواهند برد.

۸-۲- معماری آندروید :

این سیستم عامل براساس هسته سیستم عامل لینوکس توسعه یافته است. درواقع مدیریت و بهینه سازی حافظه،ارتباط با سخت افزار و سرویس های سیستم و مدیریت منابع دستگاه موبایل را هسته لینوکس انجام میدهد و آندروید فقط لایه ایست که ارتباط میان کاربر و سیستم عامل را برقرار میکند.این لایه نرم افزاری به وسیله جاوا پیاده سازی شده است. سیستم عامل آندروید از نظر معماری یک پشته نرم افزاری به حساب می آید ، به این معنی که مجموعه ای از برنامه های کوچک متصل به هم است که همگی به صورت یک سیستم عامل واحد کار میکنند.

نمودار معماری اندروید:

تصویر زیر لایه ها و بخش های اصلی سیستم عامل آندروید را نشان میدهد:



شکل ۲-معماری سیستم عامل اندروید

۱-۸-۲- تشریح لایه های مختلف معماری:

Linux Layer

در پایین ترین بخش معماری اندروید ، هسته^۱ این سیستم عامل قرار دارد. Kernel هسته مرکزی سیستم عامل و ابتدایی ترین بخش آن را تشکیل میدهد. گوگل از نسخه 2.6 لینوکس برای طراحی هسته اندروید استفاده کرد که برای انجام سرویس های اصلی شامل برنامه های مدیریت حافظه ، شبکه ، ایمنی ، پشته و Driver های سیستم میشود. این هسته همچنین مانند یک لایه انتزاعی^۲ مابین سخت افزار و سایر نرم افزارها عمل میکند.

Libraries

بخش دیگر پشته اندروید ، کتابخانه های سیستم عامل است. این بخش شامل دستورالعمل های مختلفی است که به دستگاه دستور میدهد با داده های مختلف چطور رفتار کند. برای مثال کتابخانه Media Framework شامل اطلاعات اجرای انواع فرمت های فایل های عکس ، موزیک و فیلم است.

Android Runtime

در سطح Libraries از پشته اندروید ، Android Runtime قرار دارد که شامل کتابخانه های جاوا است که در ساخت برنامه های اندروید به کار میرود و برای اجرای آن ها کاملاً حیاتی میباشد. اندروید شامل مجموعه ای از کتابخانه های اصلی است که اکثر عملکردهای قابل دسترس را با استفاده از زبان جاوا ممکن میسازد. هر برنامه کاربردی اندروید در فرایند مخصوص به خودش اجرا میشود و دسترسی مخصوص به خود در ارتباط با ماشین مجازی دالویک دارد.

بخش دیگر پشته Runtime ، Davlik Virtual Machine است. یکی از مزایای استفاده از Virtual Machine در سیستم عامل اندروید این است که هیچ برنامه ای به منابع دیگر وابسته نیست و اگر یکی از برنامه ها از کار بیفتد ، کارکرد برنامه های دیگر تحت تاثیر قرار نمیگیرد. این امر مدیریت حافظه نیز بسیار ساده میسازد.

1. Kernel

2. Abstarction Layer

Application Framework:

Application Framework امکان استفاده مجدد و جایگزینی اجزاها را فراهم میکند. با فراهم آوردن Open Development Platform ، اندروید برنامه سازان را قادر کرده است تا برنامه های کاربردی خلاقانه و غنی بسازند. معماری برنامه های کاربردی به منظور ساده سازی استفاده مجدد از اجزاها طراحی شده است. توسعه دهندگان آزادی کامل دارند تا از ویژگی هایی مانند دسترسی به سخت افزار ، دسترسی به اطلاعات محلی (موقعیت جغرافیایی) ، اجرای سرویس های پس زمینه ، تنظیم زنگ ساعت ، اضافه کردن اطلاعیه ها به نوار وضعیت و بسیاری دیگر در برنامه هایی که میسازند ، استفاده کنند. هر برنامه ای میتواند 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 هم بستگی دارد)

برنامه های زیربنایی ، مجموعه ای از سرویس ها و سیستم های زیرند:

- مجموعه View System ها که برای ساخت برنامه های کاربردی استفاده میشوند ، مانند متن ، کادر متنی ، کلیدها و ...
- Content Providers که برنامه ها را قادر میسازند 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.
- Notification Manager که از این طریق برنامه ها را قادر میکند تا هشدارهای خود را نوار وضعیت نشان دهند.
- Resource Manager اجازه دسترسی به منابعی که کد-برنامه نیستند را فراهم میکند مانند دسترسی به رشته های محلی (Localized Strings) ، تصاویر و فایل های مربوط به طرح برنامه
- Activity Manager که مدیریت چرخه زندگی (Lifecycle) برنامه ها را در دست دارد و به نحوه اجرای برنامه ها نظارت میکند.

Applications and Widgets:

در بالاترین سطح پشته اندروید ، چارچوب نرم افزارها قرار دارد. این لایه شامل برنامه های کاربردی سیستم عامل که به همراه سیستم عامل ارائه شده نظیر برقراری تماس ، استفاده از دوربین و ذخیره شماره تلفن ، نقشه ، ایمیل و ... میباشد. تمام این برنامه ها با استفاده از زبان برنامه نویسی Java نوشته شده اند.

۲-۹-رابطه اندروید و Java :

نرم افزارهای جانبی اندرویدی با استفاده از زبان جاوا نوشته میشود 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. اما این سیستم عامل ، ماشین مجازی جاوا^۱ ندارد. برای اجرای برنامه های جاوایی روی این سیستم عامل ، کدهای جاوا به کدهای دالویک تبدیل میشوند و سپس روی Dalvik Virtual Machine اجرا میشوند.

۲-۱۰- ماشین مجازی Dalvik^۲:

Dalvik یک ماشین مجازی جاوایی است که برای سیستم عامل اندروید بهینه شده است تا هم RAM و هم CPU و هم باتری کمتری مصرف کند. ماشین مجازی دالویک برای اجرای قابلیت های اساسی مانند مدیریت حافظه کم ، متکی بر هسته لینوکس است و وظیفه بهینه سازی کدها را برای اجرا بر روی موبایل را دارد. این ماشین به گونه ای ساخته شده است که هر دستگاهی میتواند چندین ماشین مجازی را به طور همزمان اجرا کند. هر ماشین مجازی دالویک فایل ها را به فرمت (.dex) اجرا میکند که این کار باعث بهینه سازی در دستگاه هایی که حافظه پایینی دارند ، میشود . این ماشین مجازی مبتنی بر رجیسترها و کلاس هایی که توسط کامپایلر Java ساخته شده است را اجرا میکند.البته همانطور که قبلا نیز گفته شد هم اکنون ران تایم جدید اندروید به نام "آرت" برای اندرویدهای 4.4 یا همان نسخه ی "کیت کت" به بعد قابل دسترسی است.

1. Java Virtual Machine

2. Dalvik Virtual Machine

۱۱-۲ بازی های آندروید :

آندروید یک سیستم عامل متن باز است که در حال حاضر بیش از ۳۰ شرکت مختلف به آن پیوسته اند. همانطور که می دانید بازی هایی که برای سیستم عامل آندروید عرضه می شوند یکی از زیباترین و گرافیکی ترین بازی های جهان می باشند. این بازی ها همچنین دارای گیم پلی روان و بسیار جذابی نیز هستند که می توانند کاربر را به سمت خود سوق دهند. بازی های اندروید با فرمت apk منتشر می شوند. برخی از بازی های آندروید دارای فایل دیتا می باشند. فایل های دیتا در اصل اطلاعات کامل کننده فایل apk می باشند که بعد از نصب کردن apk شما باید فایل دیتا را نیز در مسیری که معمولا سایت منتشر کننده آن بازی می گوید کپی کنید تا بتوانید آن بازی آندروید را شروع کنید. بازی های آندروید دارای حجم های مختلفی می باشند. برخی از این بازی ها از یک مگابایت و حتی کمتر و برخی بازی های اندروید دارای دیتای چند گیگابایتی می باشند. طبعاً هرچه دیتای یک بازی بیشتر باشد، گرافیک آن نیز بیشتر خواهد بود و کاربر گیم پلی جذاب تری تجربه خواهد کرد. امروزه یکی از بزرگترین کمپانی هایی که در زمینه تولید بازی های آندروید فعالیت دارد، کمپانی گیم لافت می باشد.

۱۲-۲ رده های سنی بازی آندروید:

بازی های آندروید مانند بازی سایر سیستم عامل ها و کنسول ها امروزه برای رده بندی های سنی مختلفی ارائه میشوند. اما نکته ای که در مورد بازی آندروید جالب است بدانید این است بازی های آندروید معمولاً به گونه ای ساخته می شوند که اکثر رده بندی های سنی بتوانند به سادگی از آن استفاده کنند و از آن لذت ببرند. اما در کل بازی های آندروید را می توان برای سه سطح کودک، نوجوان و بزرگسال در نظر گرفت. بازی های کودکان و نوجوانان به طبع دارای حجم کمتری نسبت به بازی هایی که برای بزرگسالان عرضه میشوند، می باشند. معمولاً کسانی که منتشر کننده بازی های آندروید هستند در کنار عرضه آن باید رده بندی آن را نیز نمایش دهند. در سایت های مختلف ارائه دهنده ی اپلیکیشن های آندرویدی دسته بندی های اختصاصی برای بازی های کودکان را میشود پیدا کرد، که کودکان می توانند بازی های مناسب خود را از طریق این دسته بندی انتخاب کنند و از آنها لذت ببرند.

۱۳-۲ ورژن های مختلف بازی های آندرویدی :

امروز شاهد انتشار نسخه هایی مختلف سیستم عامل آندروید هستیم. سیستم عامل آندرویدی که امروزه قدیمی

ترین آنها شناخته می شود، ورژن ۱۰۵ و بالاترین سیستم عامل آندروید فعلی ۵ می باشد. بازی هایی که برای سیستم عامل بالاتر عرضه می شوند دارای گرافیک و گیم پلی جذاب تری نسبت به بازی هایی که هستند که برای سیستم های پایین تر عرضه میشوند. اما نمی توان از این نکته غافل بود که گاهی ممکن است بازی برای سیستم عامل آندروید ورژن پایین عرضه شود که دارای گرافیک و گیم پلی فوق العاده ای باشد. شما در سایت های دانلود برنامه های آندرویدی دسته بندی های گوناگونی را میبینید که از بین آن ها میتوانید به سادگی تنها بازی ها و برنامه هایی را دانلود کنید که مناسب ورژن آندروید آن گوشی شما باشد. این کار سبب می شود زمان کمتری برای جستجوی نرم افزار و بازی آندروید مورد علاقه خود صرف کنید.

۱۴-۲ قابلیت های بازی های آندرویدی:

یکی از قابلیت های بارز بازی های آندروید که آن را نسبت به بازی سایر سیستم عامل ها متمایز می کند داشتن گرافیکی بسیار قوی و خیره کننده می باشد. البته نمی توان از گیم پلی عالی بازی های آندروید نیز چشم پوشی کرد. اگر بخواهیم در بین سیستم عامل های مختلف یک بررسی کامل داشته باشیم می توان گفت که بازی های عرضه شده برای سیستم عامل آندروید دارای بالاترین قدرت از لحاظ گرافیک و گیم پلی می باشد. گرافیک خیره کننده بازی های آندروید یکی از مواردی است که توانسته امروزه میلیون ها کاربر مختلف را به سمت خود بکشد و آنها را به خود معتاد کند.

۱۵-۲ معایب بازی های آندرویدی:

آن قدر دنیای آندروید و بازی های آندروید بزرگ و قدرتمند است که نمی توان عیبی برای آن برشمرد. اما همانطور که می دانید هیچ چیز بدون مشکلی وجود ندارد. یکی از معایبی که در بازی های آندروید به چشم می خورد این است که برخی از بازی های آندروید دارای حجم بالایی هستند و این می تواند برای کاربرانی که حافظه گوشی آنها کم می باشد، دردسر ساز باشد. همچنین بالا بودن حجم بازی های آندروید برای آن دسته از کاربرانی که بازی های خود را از طریق اینترنت دانلود می کنند نیز می تواند مشکل باشد، چرا که دانلود کردن بازی که دارای چندین گیگ حجم می باشد که می تواند ساعت ها زمان ببرد.

جاوا

جاوا^۱ یک زبان برنامه‌نویسی شی‌گراست که برای اولین بار توسط جیمز گاسلینگ در شرکت سان مایکروسیستمز ایجاد شد و در سال ۱۹۹۵ به عنوان بخشی از سکوی جاوا منتشر شد. زبان جاوا شبیه به C++ است اما مدل شی‌گرایی آسان‌تری دارد و از قابلیت‌های سطح پایین کمتری پشتیبانی می‌کند. یکی از قابلیت‌های اصلی جاوا این است که مدیریت حافظه را بطور خودکار انجام می‌دهد. ضریب اطمینان عملکرد برنامه‌های نوشته‌شده به این زبان بالا است و وابسته به سیستم‌عامل خاصی نیست، به عبارت دیگر می‌توان آن را روی هر رایانه با هر نوع سیستم‌عاملی اجرا کرد. برنامه‌های جاوا به صورت کدهای بیتی همگردانی (کامپایل) می‌شوند. که مانند کد ماشین هستند و به ویژه وابسته به سیستم‌عامل خاصی نیستند.

نام جاوا از Just Another Vague Acronym گرفته شده است و دلیل آن این بود که تیم برنامه نویسان جاوا به قهوه خیلی علاقه داشتند. این موضوع در لوگوی جاوا هم مشهود است !!



شکل ۳- لوگوی جاوا

۳-۱- تاریخچه :

در مقایسه با زبان‌های دیگر، همچون C++ یا بیسیک یا فورترن، جاوا زبان نسبتاً جدیدتری است. شرکت سان مایکروسیستم^۲ در سال ۱۹۹۱ یک پروژه تحقیقاتی به نام گرین^۳ را آغاز کرد. هدف این پروژه ایجاد زبانی جدید شبیه به C++ بود که نویسنده اصلی آن، جیمز گاسلینگ، آن را بلوط نامید. اما بعدها به دلیل برخی مشکلات حقوقی از میان لیستی از کلمات تصادفی نام آن به جاوا تغییر کرد.

1. Java

2. Sun Microsystems

3. Green

پروژه گرین به دلیل مشکلات بازاریابی در شرف لغو شدن بود تا اینکه گسترش وب در سال ۱۹۹۳ باعث نمایش توانایی‌های وافر جاوا در این عرصه گشت. اینگونه بود که شرکت سان میکروسیستمز در مه ۱۹۹۵ جاوا را رسماً به بازار عرضه کرد.

جاوا یک زبان برنامه‌نویسی است که در آغاز توسط شرکت سان میکروسیستمز ایجاد شده‌است و در سال ۱۹۹۵ به عنوان بخش اصلی سکوی جاوا منتشر شد. این زبان قسمت‌های بسیاری از گرامر خود را از C و C++ گرفته اما دارای مدل شی‌گرایی ساده‌ای است و امکانات سطح پایین کمی دارد. کاربرد جاوا در کامپایل به صورت بایت کد است که صرف نظر از معماری و خصوصیات آن کامپیوتر، قابلیت اجرا روی تمامی ماشین‌های شبیه‌سازی جاوا را داشته باشد. اجزای اصلی کامپایلرهای جاوا، ماشین‌های پیاده‌سازی و کتابخانه‌های آن توسط این شرکت از سال ۱۹۹۵ منتشر شد. در مه ۱۹۹۷ این شرکت، نرم‌افزار رایگان این زبان را فراهم کرد. دیگران هم کاربردهای دیگری از این زبان را منتشر کردند مثل کامپایلر GNU برای جاوا.

مرورگرهای اصلی وب، به هم پیوستند تا به طور مطمئن جاوا اپلت را بدون صفحات وب اجرا کنند و به این صورت جاوا خیلی زود معروف و محبوب شد. با پیدایش java2، نسخه ی جدید توانست ترکیب‌های جدیدی را برای نوع‌های مختلف پلت فرم‌ها ایجاد کند. به عنوان مثال J2EE، باهدف کاربرد برای تشکیلات اقتصادی، و نسخه ی سکوی جاوا، نسخه میکرو برای تلفن همراه منتشر شد. در سال ۱۹۹۶ با هدف بازاریابی، این شرکت نسخه ی جدید J2 را با نام‌های سکوی جاوا، نسخه سازمانی، سکوی جاوا، نسخه میکرو و سکوی جاوا، نسخه استاندارد منتشر کرد. در سال ۱۹۹۷ شرکت سان میکروسیستمز، ISO/IEC JTC1 standards body و Ecma International را به فرمول جاوا تغییر داد. شرکت Sun بسیاری از کاربردهای جاوایش را بدون هیچ هزینه‌ای فراهم آورد. شرکت Sun با فروش مجوز برای بعضی از کاربردهای خاص مثل Java Enterprise System درآمدی را بدست آورد. در ۱۳ نوامبر ۱۹۹۶ شرکت Sun نرم‌افزار جاوا را به صورت رایگان و با مجوز عمومی برای همه منتشر کرد.

۳-۲- اهداف اولیه:

- این زبان باید ساده، شی گرا و مشهور باشد.
- مطمئن و بدون خطا باشد.
- وابسته به معماری کامپیوتر نبوده و قابل انتقال باشد.
- باید با کارایی بالا اجرا شود.
- باید به صورت پویا و نخ کشی شده باشد

۳-۳- نسخه ها :

نسخه های جاوا و تاریخ انتشارشان به شرح زیر است :

جدول ۲-نسخه های جاوا

نسخه	تاریخ انتشار
JDK 1.0	۲۱ ژانویه ۱۹۹۶
JDK 1.1	۱۹ فوریه ۱۹۹۷
J2SE 1.2	۸ دسامبر ۱۹۹۸
J2SE 1.3	۸ می ۲۰۰۰
J2SE 1.4	۶ فوریه ۲۰۰۲
J2SE 5.0	۳۰ سپتامبر ۲۰۰۴
Java SE 6	۱۱ دسامبر ۲۰۰۶
Java SE 7	۲۸ ژوئیه ۲۰۱۱
Java SE 8	۱۸ مارچ ۲۰۱۴

۳-۴- برنامه های جاوا و اپلت ها :

جاوا برای نوشتن انواع برنامه های کاربردی مناسب است. با جاوا می توان انواع برنامه های زیر را نوشت:

- برنامه های تحت وب
- برنامه نویسی سیستم های کوچک مانند تلفن های همراه و تبلت ها
- برنامه های کاربردی بزرگ^۱
- برنامه های رومیزی^۲

قابلیت خاصی در جاوا وجود دارد بنام اپلت. اپلت ها امکانات فراوانی برای نوشتن برنامه های تحت وب در اختیار برنامه نویسان قرار می دهند که دیگر زبان های برنامه نویسی فاقد آن هستند. البته وجود ماشین مجازی جاوا برای اجرای اپلت لازم است. اپلت ها نظیر فناوری اکتیو ایکس شرکت مایکروسافت هستند که برنامه نویسان را قادر می سازد تا امکاناتی را به مرورگر کاربر بیفزایند. البته تفاوت این دو در امنیت می باشد به گونه ای که اپلت ها بدلیل اینکه در محیطی به نام sandbox اجرا می شوند امن هستند ولی اکتیو ایکس ها فاقد چنین امنیتی هستند.

۳-۵- .Net. رقیب Java ؟

.NET فرمی است که به وسیله مایکروسافت^۳ برای ساخت سایت ها با قابلیت های زیاد و متفاوت ایجاد شده است به طوری که قسمتی از یک پروژه را می توان با ++C و بخش دیگر را با برنامه ای دیگر نوشت و در نهایت کل آن توسط یک واسطه به نام MSIL ترجمه شده و در محیط .NET قابل اجرا است .

J2EE به وسیله شرکت SUN و با همکاری شرکت های IBM و HP تهیه شده است. در این قالب تنها زبانی که مورد استفاده قرار می گیرد جاوا است، و بر خلاف .NET. که فقط روی محیط های سازگار با ویندوز عمل می کند، مستقل از سخت افزار است .

-
1. Enterprise
 2. Desktop
 3. MicroSoft

۳-۶- بین .NET و JAVA کدام را انتخاب کنیم؟

-قابلیت انتقال برنامه ها روی سیستم های مختلف و شعار اصلی جاوا یعنی: "یک بار بنویس و هر جا استفاده کن (Writeonce-Run anywhere)" موضوعی بسیار حایض اهمیت است که بسته به نیاز شما و نوع برنامه شما می تواند خیلی مهم باشد .

-در محیط هایی که چند نوع سیستم سخت افزاری وجود دارد قطعا برنامه های تحت جاوا مناسب تر است ولی اگر در محیط مورد نظر شما از سیستم عامل ویندوز استفاده می شود استفاده از .NET. هزینه کمتری در برخواهد داشت .

.NET ویژگی هایی دارد که با آن می توان سایت های جذاب و زیباتری ساخت اما در سایت های پیچیده و بزرگ و جایی که کارایی مهمتر از ظاهر است بهتر است از جاوا استفاده کنیم .

-تصور کنید نیاز به برنامه ای دارید که در آن چیزهایی به صورت دینامیک ایجاد می شوند و پس از طی یک سری عملیات بر روی آنها از بین می روند، در این نوع برنامه ها تعقیب اشیایی که در برنامه ساخته می شوند، تخصیص و مدیریت حافظه تماما بر عهده برنامه نویس است. اما جاوا دارای یک سیستم خودکار مدیریت و پاکسازی حافظه است که بسیاری از مشکلات را آسان کرده است .

هر تکنولوژی که ایجاد می شود؛ هر برنامه ای که مورد توجه عموم قرار می گیرد قطعا مورد توجه نفوذگران هم واقع می شود، هر سیستمی که بخواهد بماند و پیشرفت کند نیاز به ایمنی دارد و اینجاست که سدهای حفاظتی جاوا و مدل های امنیتی آن که حاصل تلاش زبردست ترین برنامه نویسان دنیاست مورد توجه قرار می گیرد. مدل چهار لایه امنیتی جاوا جلوی خیلی از اقدامات خرابکارانه و نفوذی را می گیرد، ولی هیچ سدی صددرصد غیر قابل نفوذ نیست !!

در هر حال انتخاب بین این دو نیاز به بررسی دقیق، آینده نگری و مشخص شدن اهداف دراز مدت و میزان سرمایه گذاری شما دارد زیرا هزینه هایی که یک برنامه جاوا دربر دارد در اکثر موارد چندین برابر یک برنامه .NET خواهد بود.

بطور خلاصه بدلائل زیر جاوا مورد استقبال تر است:

سیستم عامل: هر چقدر زبانهای net. قوی باشند تنها بر روی پلت فرم ویندوز اجرا می شوند و برخی ویندوز را سیستم عامل غیر قابل اعتمادی در برنامه نویسی Enterprise می دانند. ولی جاوا از این نظر انتخاب خوبی است

قابلیت حمل: جاوا بر روی سکوهاى رایانش گوناگونی قابل اجرا است، از ATM و ماشین رختشویی گرفته تا سرورهای سولاریس با قابلیت پشتیبانی از cpu 1024 برای پردازش.

جاوا بیشتر از یک زبان است: جاوا فقط یک زبان نیست و انجمن هایی متشکل از بزرگان صنایع و برنامه نویسان زیادی مشغول به توسعه و ایجاد استانداردهای جدید و به روز هستند.

۳-۷- خط مشی جاوا:

یکی از ویژگی های جاوا قابل حمل بودن آن است. یعنی برنامه ی نوشته شده به زبان جاوا باید به طور مشابهی در کامپیوترهای مختلف با سخت افزارهای متفاوت اجرا شود. و باید این توانایی را داشته باشد که برنامه یک بار نوشته شود، یک بار کامپایل شود و در همه کامپیوترها اجرا گردد. به این صورت که کد کامپایل شده ی جاوا را ذخیره می کند، اما نه به صورت کد ماشین بلکه به صورت بایت کد جاوا. دستورالعمل ها شبیه کد ماشین هستند، اما با ماشین های مجازی که به طور خاص برای سخت افزارهای مختلف نوشته شده اند، اجرا می شوند. در نهایت کاربر از سکوی جاوا نصب شده روی ماشین خود یا مرورگر وب استفاده می کند. کتابخانه های استاندارد یک راه عمومی برای دسترسی به ویژگی های خاص فراهم می کنند. مانند گرافیک، نخ کشی و شبکه. در بعضی از نسخه های ماشین مجازی جاوا، بایت کدها می توانند قبل و در زمان اجرای برنامه به کدهای محلی کامپایل شوند. فایده ی اصلی استفاده از بایت کد، قسمت کردن است. اما ترجمه ی کلی یعنی برنامه های ترجمه شده تقریباً همیشه کندتر از برنامه های کامپایل شده ی محلی اجرا می شوند. این شکاف می تواند با چند تکنیک خوش بینانه که در کاربردهای JVM قبلی معرفی شد، کم شود. یکی از این تکنیک ها JIT است که بایت کد جاوا را به کد محلی ترجمه کرده و سپس آن را پنهان می کند. در نتیجه برنامه خیلی سریع تر نسبت به کدهای ترجمه شده ی خالص شروع و اجرا می شود. بیشتر VM های پیشرفته، به صورت کامپایل مجدد پویا، در آنالیز VM، رفتار برنامه ی اجرا شده و کامپایل مجدد انتخاب شده و بهینه سازی قسمت های برنامه، استفاده می شوند. کامپایل مجدد پویا می تواند کامپایل ایستا را بهینه سازی کند. زیرا می تواند قسمت hot spot برنامه و گاهی حلقه های

داخلی که ممکن است زمان اجرای برنامه را افزایش دهند را تشخیص دهد. کامپایل JIT و کامپایل مجدد پویا به برنامه‌های جاوا اجازه می‌دهد که سرعت اجرای کدهای محلی بدون از دست دادن قابلیت انتقال افزایش پیدا کند.

تکنیک بعدی به عنوان کامپایل ایستا شناخته شده است. که کامپایل مستقیم به کدهای محلی است مانند بسیاری از کامپایلرهای قدیمی. کامپایلر ایستای جاوا، بایت‌کدها را به کدهای شی محلی ترجمه می‌کند.

کارایی جاوا نسبت به نسخه‌های اولیه بیشتر شد. در تعدادی از تست‌ها نشان داده شد که کارایی کامپایلر JIT کاملاً مشابه کامپایلر محلی شد. عملکرد کامپایلرها لزوماً کارایی کدهای کامپایل شده را نشان نمی‌دهند. یکی از پیشرفت‌های بی نظیر در در زمان اجرای ماشین این بود که خطاها ماشین را دچار اشکال نمی‌کردند. علاوه بر این در زمان اجرای ماشینی مانند جاوا وسایلی وجود دارد که به زمان اجرای ماشین متصل شده و هر زمانی که یک استثنا رخ می‌دهد، اطلاعات اشکال زدایی که در حافظه وجود دارد، ثبت می‌کنند.

۳-۸- پیاده سازی:

شرکت سان میکروسیستم مجوز رسمی برای پلت فرم استاندارد جاوا را به مایکروسافت ویندوز، لینوکس، و سولاریس (سیستم عامل). داده است. همچنین محیط‌های دیگری برای دیگر پلت فرم‌ها فراهم آورده است. علامت تجاری مجوز شرکت سان میکروسیستم طوری بود که با همه ی پیاده‌سازی‌ها سازگار باشد. به علت اختلاف قانونی که با مایکروسافت پیدا کرد، زمانی که شرکت سان ادعا کرد که پیاده‌سازی مایکروسافت از RMI یا JNI پشتیبانی نکرده و ویژگی‌های خاصی را برای خودش اضافه کرده است. شرکت سان در سال ۱۹۹۷ پیگیری قانونی کرد و در سال ۲۰۰۱ در توافقی ۲۰ میلیون دلاری برنده شد. در نتیجه کمی بعد مایکروسافت جاوا را به ویندوز فرستاد. در نسخه ی اخیر ویندوز، مرورگر اینترنت نمی‌تواند از جاوا پلت فرم پشتیبانی کند. شرکت سان و دیگران یک سیستم اجرای جاوای رایگان برای آنها و نسخه‌های دیگر ویندوز فراهم آوردند.

۳-۹- اداره ی خودکار حافظه :

جاوا از حافظه ی باز یافتی خودکار برای اداره ی حافظه در چرخه ی زندگی یک شی استفاده می کند. برنامه نویس زمانی که اشیا به وجود می آیند، این حافظه را تعیین می کند. و در زمان اجرا نیز، زمانی که این اشیا در استفاده ی زیاد طولانی نباشند، برنامه نویس مسئول بازگرداندن این حافظه است. زمانی که مرجعی برای شی های باقی مانده نیست، شی های غیر قابل دسترس برای آزاد شدن به صورت خودکار توسط بازیافت حافظه، انتخاب می شوند. اگر برنامه نویس مقداری از حافظه را برای شی هایی که زیاد طولانی نیستند، نگه دارد، چیزهایی شبیه سوراخ حافظه اتفاق می افتند.

یکی از عقایدی که پشت سر مدل اداره ی حافظه ی خودکار جاوا وجود دارد، این است که برنامه نویس هزینه ی اجرای اداره ی دستی حافظه را نادیده می گیرد. در بعضی از زبان ها حافظه لازم برای ایجاد یک شی، به صورت ضمنی و بدون شرط، به پشته تخصیص داده می شود. و یا به طور صریح اختصاص داده شده و از heap بازگردانده می شود. در هر کدام از این راه ها، مسئولیت اداره ی اقامت حافظه با برنامه نویس است. اگر برنامه شی را برنگرداند، سوراخ حافظه اتفاق می افتد. اگر برنامه تلاش کند به حافظه ای را که هم اکنون بازگردانده شده، دستیابی پیدا کند یا برگرداند، نتیجه تعریف شده نیست و ممکن است برنامه بی ثبات شده و یا تخریب شود. این ممکن است با استفاده از اشاره گر مدتی باقی بماند، اما سرباری و پیچیدگی برنامه زیاد می شود. بازیافت حافظه اجازه دارد در هر زمانی اتفاق بیفتد. به طوری که این زمانی اتفاق می افتد که برنامه بی کار باشد. اگر حافظه ی خالی کافی برای تخصیص شی جدید در هیپ وجود نداشته باشد، ممکن است برنامه برای چند دقیقه متوقف شود. در جایی که زمان پاسخ یا اجرا مهم باشد، اداره ی حافظه و منابع اشیا استفاده می شوند.

جاوا از نوع اشاره گر ریاضی C و C++ پشتیبانی نمی کند. در جایی که آدرس اشیا و اعداد صحیح می توانند به جای هم استفاده شوند. همانند C++ و بعضی زبان های شی گرای دیگر، متغیرهای نوع های اولیه ی جاوا شی گرا نبودند. مقدار نوع های اولیه، مستقیماً در فیلدها ذخیره می شوند. در فیلدها (برای اشیا) و در پشته (برای توابع)، بیشتر از هیپ استفاده می شود. این یک تصمیم هوشیارانه توسط طراح جاوا برای اجرا است. به همین دلیل جاوا یک زبان شی گرای خالص به حساب نمی آید.

۳-۱۰- گرامر :

گرامر جاوا وسیع تر از C++ است و برخلاف C++ که ترکیبی است از ساختارها و شی گرایی، زبان جاوا یک زبان شی گرای خالص می باشد. فقط نوع داده ی اصلی از این قاعده مستثنی است. جاوا بسیاری از ویژگی ها را پشتیبانی می کند و از کلاس ها برای ساده تر کردن برنامه نویسی و کاهش خطا استفاده می کند.

۳-۱۱- توزیع های جاوا :

منظور از توزیع جاوا پیاده سازی های مختلفی است که برای کامپایلر جاوا و همچنین مجموعه کتابخانه های استاندارد زبان جاوا (JDK) وجود دارد. در حال حاضر چهار توزیع کننده ی عمده جاوا وجود دارند:

- **سان میکروسیستمز:** توزیع کننده اصلی جاوا و مبدع آن می باشد. در اکثر موارد هنگامی که گفته می شود جاوا منظور توزیع سان می باشد.
- **GNU Classpath:** این توزیع از سوی موسسه نرم افزارهای آزاد منتشر شده و تقریباً تمامی کتابخانه استاندارد زبان جاوا در آن بدون بهره گیری از توزیع شرکت سان از اول پیاده سازی شده است. یک کامپایلر به نام GNU Compiler for Java نیز برای کامپایل کردن کدهای جاوا توسط این موسسه ایجاد شده است. فلسفه انتخاب نام Classpath برای این پروژه رها کردن تکنولوژی جاوا از وابستگی به علامت تجاری جاوا است بطوریکه هیچ وابستگی یا محدودیتی برای استفاده آن از لحاظ قوانین حقوقی ایجاد نشود و از طرفی به خاطر وجود متغیر محیطی classpath در تمامی محیط های اجرایی برنامه های جاوا، این نام به نوعی تکنولوژی جاوا را برای خواننده القا می کند. کامپایلر GNU توانایی ایجاد کد اجرایی (در مقابل بایت کد توزیع سان) را داراست. لازم به ذکر است که در حال حاضر شرکت سان تقریباً تمامی کدهای JDK را تحت مجوز نرم افزارهای آزاد به صورت متن باز منتشر کرده است و قول انتشار قسمت بسیار کوچکی از این مجموعه را که به دلیل استفاده از کدهای شرکت های ثانویه نتوانسته به صورت متن باز منتشر نماید در آینده نزدیک با بازنویسی این کدها داده است.
- **مایکروسافت J#:** این در حقیقت یک توزیع جاوا نیست. بلکه زبانی مشابه می باشد که توسط مایکروسافت و در چارچوب .net ارائه شده است. انتظار اینکه در سیستم عاملی غیر از ویندوز هم اجرا شود را نداشته باشید

۳-۱۲- کتابخانه های جاوا :

کتابخانه های جاوا که به صورت بایت کد از کد اصلی کامپایل شده اند، برای پشتیبانی از کاربردهای جاوا، توسط JRE منتشر شده است. مثال هایی از این کتابخانه ها عبارتند از:

- کتابخانه های مرکزی
- کتابخانه هایی که برای ساختار داده کاربرد دارند. مثل لیست ها، درخت ها، مجموعه ها، مترجم ها.
- کتابخانه ی پرداز XML (تجزیه، تغییر شکل، اعتبار)
- کتابخانه های موضعی و بین المللی
- کتابخانه های انتگرال گیری که امکان تایپ کردن توسط سیستم های بیرونی را می دهند.
- JDBC برای دستیابی به داده ها
- JNDI برای مراجعه و کشف کردن
- CORBA & RMI برای توسعه ی کاربرد توزیع کردن
- کتابخانه های واسط کاربر
- AWT (توابع پنجره ای مجرد) که قسمتهایی از GUI را فراهم می کنند.
- کتابخانه های swing که در AWT ساخته شده اند اما کاربردهایی از AWT widgetry را فراهم می کنند.
- APL ها برای ضبط صدا، پردازش و بازنواختی
- کاربردهای وابسته ی پلت فرم ماشین های مجازی جاوا
- Plugins که توانایی اجرا شدن در مرورگرهای وب را به اپلت می دهد.
- java web start
- دادن مجوز و مستندسازی

۳-۱۳- ویرایش ها :

شرکت سان میکروسیستم، ۴ نوع ویرایش از کاربردهای مختلف جاوا را ارائه داده است:

1. Java card for smartcard

2. JavaME

3. JavaSE

4. JavaEE

۳-۱۴- ایرادات مطرح شده :

مهم‌ترین ایرادی که برنامه نویسان سایر زبان‌ها به زبان جاوا می‌گیرند سرعت اجرایی پایین جاوا در مقایسه با زبان‌ها سطح پایین‌تر مانند C++ و اسمبلی است. یک برنامه جاوا به صورت بایت کد می‌باشد و باید در ماشین مجازی جاوا اجرا گردد. به همین دلیل سرعت اجرای پایینی را در مقابل زبان‌هایی همچون C++ دارد. به صورت دیگر یک برنامه C به طور متوسط تا ۱۰ برابر سریعتر از برنامه مشابه جاوا اجرا می‌گردد.

جاوا علی‌رغم شیء‌گرا بودن در بخشی از قسمت‌ها برخی از اصول شیء‌گرایی را نادیده گرفته‌است. از جمله این قسمت‌ها قابلیت بازتابش^۱ می‌باشد. هدف اصلی بازتابش بررسی (مشاهده) و ایجاد تغییر در برنامه در حال اجرا است ولی این مهم با زیر پا گذاشتن بعضی اصول ممکن شده‌است. برای نمونه با استفاده از بازتابش (و در صورت داشتن مجوز لازم ضمن اجرای برنامه) می‌توان به متدهای خصوصی دیگر کلاس‌ها دسترسی داشت.

زبان جاوا در مقابل زبانی مثل C++ ساده‌تر و یادگیری آن آسانتر است. این آسانتر بودن با حذف بسیاری از موارد که باعث قدرتمندتر بودن زبان C++ بوده‌اند ایجاد شده‌است. مهم‌ترین این موارد اشاره گر‌ها و وراثت چندگانه بوده‌اند که در زبان جاوا یافت نمی‌شوند.

از آنجایی که جاوا زبانی با عدم وابستگی به بستر می‌باشد پس استفاده از توابع سیستم‌عامل در برنامه را به طور مستقیم نمی‌پذیرد. به همین صورت نمی‌توان به طور مستقیم از واسط‌های برنامه نویسی غیر از جاوا در آن استفاده نمود.

1.Reflection

۳-۱۵- پاسخ به ایرادات :

سرعت پایین برنامه‌های جاوا در محیطی که اجرا می‌شوند ملاک کارایی نبوده زیرا در محیط وب مسئله‌ای که سرعت را کند می‌سازد، شبکه بوده و ابتدا باید سربر شبکه را از روی برنامه‌ها برداشت. از طرف دیگر در برنامه‌های رومیزی هم در JDK 5.0 و ۶۰۰، بهینه‌سازی بسیاری بوجود آمده که این مسئله باعث شده که در آخرین تست کارایی که انجام شده یک برنامه جاوا در محدوده ۰.۸ تا ۱.۳ همان برنامه در ++C کارایی داشته باشد که ۱.۳ آن مربوط به بخش واسط کاربری و سرعت ۰.۸ آن مربوط به بسته تخلیه حافظه می‌شده که هیچ الگوریتمی نتوانست از الگوریتم Garbage Collector جاوا پیشی بگیرد. همچنین سال ۱۹۹۹ در مقاله‌ای آقای Lutz Prechelt به این مسئله را ثابت کردند که تجربه برنامه‌نویسی که برنامه‌ای را می‌نویسد از انتخاب زبانی که برنامه بر روی آن نوشته می‌شود در کارایی تأثیر بیشتری دارد و این بدان معناست که کارایی یک برنامه را برنامه‌نویس مشخص می‌کند و نه زبان برنامه‌نویسی (ایشان در همان مقاله از زبان جاوا استفاده نمودند تا ذهنیت بد را از بین ببرند)

همچنین در صنعت نرم‌افزار هزینه اصلی مربوط به ساخت نرم‌افزار است و نه تهیه سخت‌افزار برای دستیابی به سرعت بیشتر.

حذف اشاره‌گرها در جاوا به دلیل مشکلاتی بوده که آنها در طول تاریخشان بوجود آورده‌اند، اگرچه این موارد در برنامه‌های سیستمی لازم به نظر می‌رسد ولی در محیط‌های تحت وب که بستر اصلی جاوا هستند می‌توانند اثراتی به مراتب شدیدتر نسبت به آنچه در برنامه‌های سیستمی دارند داشته باشند و باعث می‌شود که توجه برنامه‌نویسان از مسائلی چون کارایی، قابلیت اطمینان و مقیاس‌پذیری برنامه به تنظیم اشاره‌گرها معطوف گردد.

وجود وراثت چندگانه در زبانی مانند ++C، باعث ایجاد مشکلات اساسی می‌گردد که اکثر برنامه‌نویسان ++C از آن دوری می‌کرده و هنوز هم می‌کنند. ولی قابلیت چندریخته شدن یک کلاس از لحاظ شی گرای بسیار مهم بوده و بنابراین توجیهی برای وجود وراثت چندگانه را فراهم می‌نمود.

در جاوا با وارد شدن مفهومی به نام واسط برنامه‌سازی^۱، دیگر نیازی به وجود وراثت چندگانه احساس نشد و بنابراین از زبان جاوا حذف گردید. در حال حاضر اکثر طراحان برنامه‌ها حتی به این نتیجه رسیده‌اند که وراثت تکی هم باعث ایجاد مشکل بوده و تا آنجایی که می‌شود باید از Composition استفاده نمود و در تمامی کتاب‌های طراحی که از سال ۲۰۰۰ به این طرف چاپ شده به آن اشاره نموده‌اند.

از ابتدای بوجود آمدن جاوا، کتابخانه JNI - Java Native Interface در آن وجود داشته که قابلیت فراخوانی و دستکاری برنامه‌هایی در C++ و... را می‌داده که از نمونه‌های آن می‌توان به Jtwin که یک بسته‌ایست که از کتابخانه‌های ویندوز برای اسکن عکس استفاده می‌کند، یا SWT که یک بسته نرم‌افزاریست که از کتابخانه‌های ویندوز و لینوکس (برحسب سیستم‌عامل) برای ساخت واسط کاربری^۲ استفاده می‌کند، نام برد.

بسیاری از موارد یاد شده به عنوان ایرادات به جاوا به عنوان ایرادات به طراحی زبان‌های سطح بالا هستند و نه جاوا.

۳-۱۶- یک اشتباه متداول !

برخی مردم به علت شباهت اسمی، جاوا و جاوااسکریپت را با هم اشتباه می‌گیرند. در حالیکه این دو زبان گرچه در ظاهر و کلمات شبیه‌ند ولی بطور ساختاری با یکدیگر متفاوتند. جاوا اسکریپت محصول شرکت نت اسکایپ است. جاوا برای اجرا باید به زبان ماشین مجازی ترجمه شود اما جاوااسکریپت زبانی است که معمولاً در صفحات وب نوشته می‌شود و توسط مرورگر تفسیر می‌گردد. در جاوا متغیرها همگی بر اساس نوعشان معرفی می‌شوند اما در جاوااسکریپت نوع متغیرها به صورت ضمنی مشخص می‌شود.

1. Interface

2. UI : User Interface

۳-۱۷- برنامه نویسی برای آندروید :

برای شروع کار با آندروید به منظور برنامه نویسی برای آن ابتدا چند برنامه مهم از جمله یک IDE (که بهترین پیشنهاد Eclipse است ، اما از موارد دیگر مانند Netbeans هم میتوانید استفاده کنید) را نصب کنید. دیگر مواردی که باید داشته باشید :

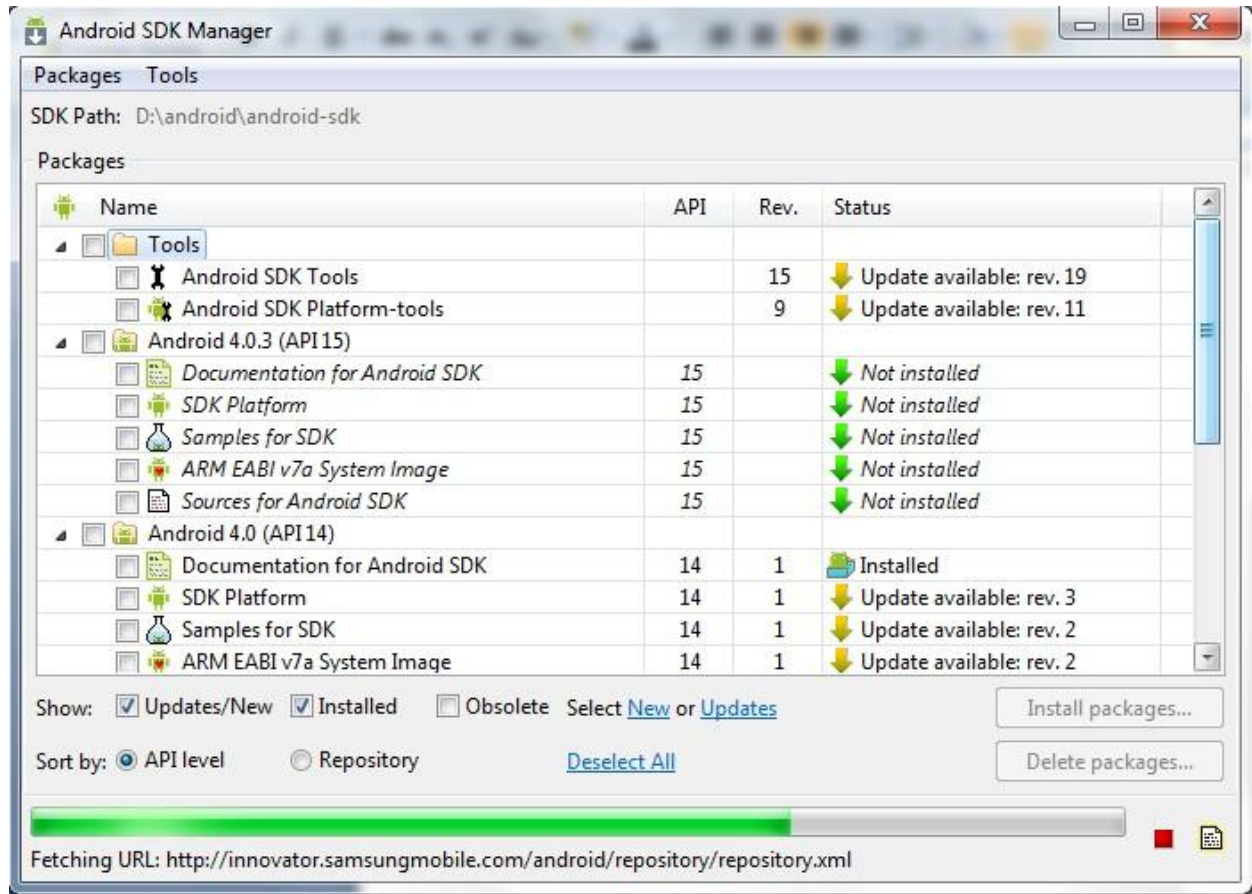
- Java : رابط برنامه نویسی است.
- ADT : یک پلاگین برای Eclipse است که برای شناساندن برنامه به اندروید به کار میرود.
- USB Drive : زمانی به کار میرود که بخواهید برنامه نوشته شده را به گوشی انتقال دهید.
- API : مخفف Application Programming Interface میباشد که به معنای رابط برنامه سازی کاربردی است و توابعی هستند که در برنامه نویسی به کار میروند.
- AVD : مخفف Android Virtual Device است که شبیه ساز سیستم عامل اندروید است.

۳-۱۷-۱- نحوه ی نصب SDK :

SDK مخفف Software Development Kit است که درواقع مجموعه ای از زبان برنامه نویسی ، کتابخانه های اندروید ، شبیه ساز سیستم عامل ، مستندات آندروید و IDE (که پیشنهاد گوگل همان Eclipse است) و فایل های نمونه و آموزشی که تمام کلاس ها و رابط های مورد استفاده در برنامه نویسی آندروید را کاملاً توضیح میدهد و API های لازم برای شروع برنامه نویسی بر روی پلتفرم اندروید با زبان برنامه نویسی جاوا را مهیا کرده است.

برای شروع کار با SDK ابتدا باید برنامه Java نصب شود ، چون Java رابط با برنامه نویسی است و ابزارهای توسعه اندروید به آن نیاز دارند. بعد از آن Google Installer را نصب کنید تا بتوانید SDK را دانلود کنید.

در صفحه ی بعد میتوانید تصویر SDK Manager را مشاهده نمایید.

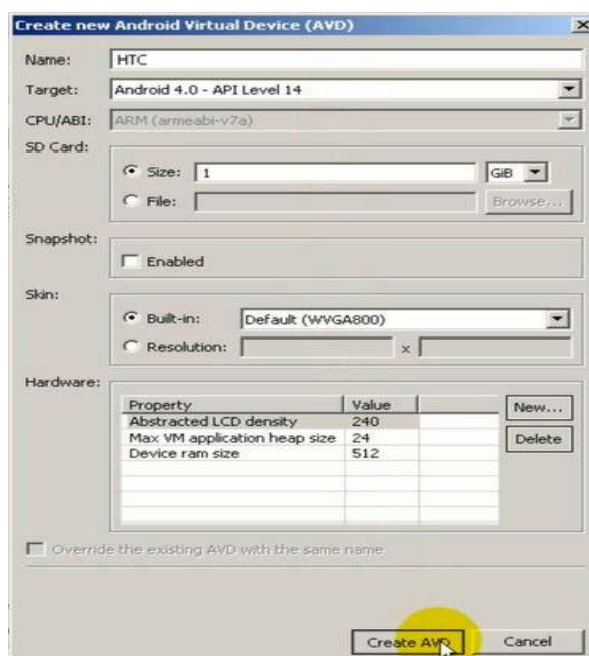


شکل ۴- SDK Manager

۳-۱۷-۲-AVD:

AVD شبیه ساز سیستم عامل اندروید است که یکی از کاربردهای آن تست نرم افزار است.

قبل از اینکه بتوانید برنامه ای را که توسط Eclipse نوشته اید بر روی شبیه ساز مشاهده کنید باید شبیه ساز را تنظیم کنید مثلاً بگویید ایت شبیه ساز قرار است کدام نسخه از اندروید را برای شما شبیه سازی کند یا کدام سطح API را میخواهید پشتیبانی کنید. یکی از مزایای ماشین مجازی اندروید نسبت به سایر شبیه سازها مانند نوکیا یا اپل این است که به تعداد نامحدودی میتوانید شبیه ساز داشته باشید. ابتدا به محلی که SDK را نصب کردید بروید و AVD را اجرا کنید. سپس پنجره ی اصلی Android Virtual Device Manager را مشاهده میکنید. در قسمت راست بر روی New کلیک کنید تا پنجره ی Create new Android Virtual Device – AVD باز شود. اطلاعات موردنظر را وارد فیلدهای مربوطه کنید. در آخر بر روی دکمه Create AVD کلیک کنید.



شکل ۵-مراحل نصب شبیه ساز

Name: به دلخواه خود یک نام انتخاب کنید. از آنجا ممکن است به چند ماشین مجازی نیاز داشته باشید، نامی را انتخاب کنید که بتوانید تشخیص دهید چه ماشینی است و کدام سطح از API را قرار است پشتیبانی کند.

Target: در این قسمت مشخص میکنید که ماشین کدام ورژن از اندروید ، با چه سطح API ای را شبیه سازی کند. دقت کنید اگر برنامه ای برای نسخه 2.2 نوشته شود قابل اجرا روی نسخه های پایینتر یا قبلی نیست.

SD Card: این فضا برای اجرا و نصب یک برنامه از فضای کامپیوتر شما کسر میشود.

Skin: اندازه صفحه نمایش را مشخص میکند.

Hardware: اگر نیاز به قابلیت خاصی دارید (مثلا استفاده از سنسور شتاب دهنده) میتوانید با کلیک بر روی New آن را به لیست اضافه کنید.

با کلیک بر روی Create AVD به پنجره قبلی باز میگردید و میبینید که ماشین مجازی که ساختید به لیست اضافه شده است. حال در برنامه Eclipse وقتی برنامه تان را اجرا کنید ، Eclipse برنامه شما را به این ماشین مجازی میفرستد و شما میتوانید برنامه ای را که نوشتید آزمایش کنید.



شکل ۶- شبیه ساز اندروید

۳-۱۷-۳- نصب برنامه در شبیه ساز توسط کامپیوتر:

برنامه ها و بازی های نوشته شده برای اندروید با پسوند apk. ذخیره میشوند. Apk ها فایل های فشرده شده ای هستند که آندروید قابلیت شناسایی آنها را دارد. برای نصب برنامه های موردنظر مطابق زیر عمل کنید: (مراحل یک تا چهار برای نصب در گوشی و از آن به بعد برای گوشی و شبیه ساز یکسان است)

یک. در گوشی به مسیر Application → Setting بروید و گزینه ی Unknown Sources را فعال کنید.

دو. حال به Application → Development → Setting بروید و گزینه USB Debugging را فعال کنید.

سه. گوشی را با کابل به کامپیوتر متصل کنید و منتظر باشید تا آن را بشناسد (اگر گوشی شناسایی نشد از این دو درایور استفاده کنید : Google USB Driver Rev و Android SDK

چهار. حال SDK را در آدرس C:\AndroidSDK ذخیره کنید.

پنج. سپس برنامه ی مورد نظر با پسوند apk. را به مسیر AndroidSDK\platform-tools انتقال دهید.

شش. یک CMD در ویندوز باز کنید و به آدرس برنامه بروید.

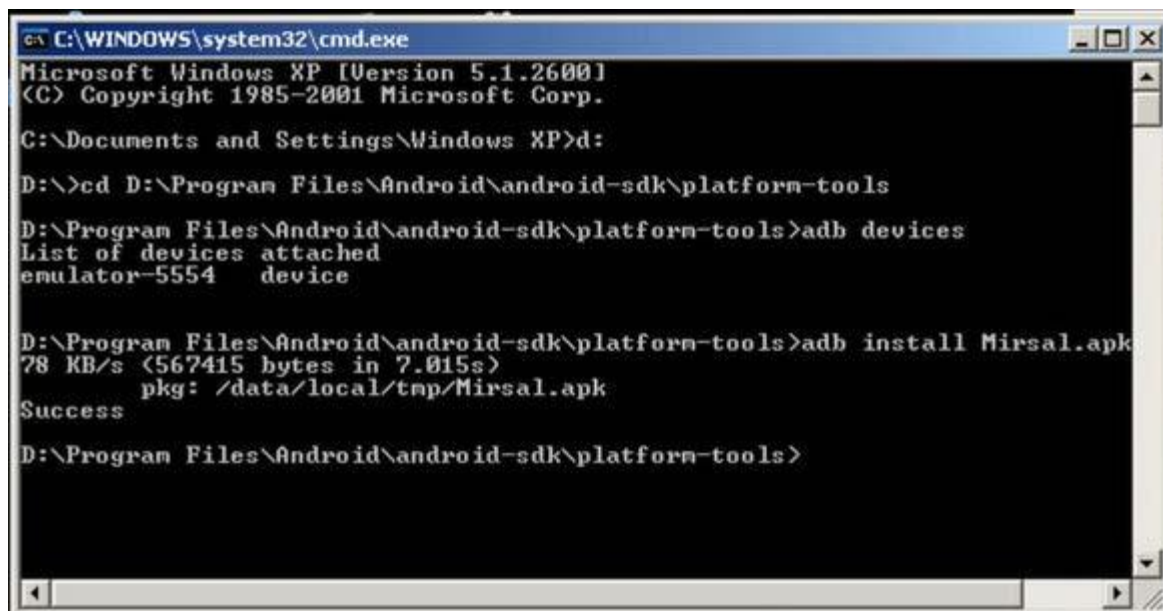
هفت. سپس در CMD دستورات زیر را تایپ کنید:

Adb devices

هشت. حال باید شماره سریال دستگاه را ببینید که نشان میدهد مراحل را درست انجام داده اید. حال این دستور را برای نصب برنامه بنویسید: (مثلا اگر نام برنامه Mirsal.apk است به این شکل:)

Adb install Mirsal.apk

نه. سپس ارتباط با گوشی برقرار شده و در انتها "Seccess" نوشته میشود.



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Windows XP>d:

D:\>cd D:\Program Files\Android\android-sdk\platform-tools

D:\Program Files\Android\android-sdk\platform-tools>adb devices
List of devices attached
emulator-5554    device

D:\Program Files\Android\android-sdk\platform-tools>adb install Mirsal.apk
78 KB/s (567415 bytes in 7.015s)
pkg: /data/local/tmp/Mirsal.apk
Success

D:\Program Files\Android\android-sdk\platform-tools>
```

شکل ۷- کدهای مربوط به نصب برنامه در شبیه ساز توسط CMD

برای Uninstall کردن برنامه ، به Manage Applications → Application → Setting گوشی یا شبیه ساز بروید. در آنجا اطلاعات کاملی از نرم افزارهای نصب شده روی گوشی یا شبیه ساز میبینید . سپس روی نرم افزار مورد نظر کلیک کنید و Uninstall را انتخاب کنید. بدین صورت برنامه مورد نظر پاک میشود.

۳-۱۸- Eclipse و برنامه نویسی با آن:

۳-۱۸-۱- برنامه نویسی آندروید چگونه است؟

آندروید دارای کتابخانه های برنامه نویسی قدرتمندی است که کار برنامه نویسان را بسیار ساده میکند. از جمله پایگاه داده رابطه ای کوچک شده ی SQLite، توابع گرافیکی سه بعدی OpenGL، موتور مرورگر webkit (همانند مرورگر Chrome) و موتور گرافیکی SGL. برنامه نویسی برای آندروید به زبان جاوا انجام میگردد و در نهایت کامپایلر Dalvik برنامه نوشته شده به زبان جاوا را به کدهای مقصد که قابل فهم برای سیستم عامل اندروید باشند تبدیل میکند و در آخر یک برنامه Package Manager که فایلی قابل نصب است با پسوند apk. تولید میکند.

۳-۱۸-۲- Eclipse چیست؟

اکلیپس یک محیط توسعه نرم افزاری چندزبانه برای محیط توسعه مجتمع با قابلیت اضافه کردن افزونه می باشد. این محیط توسعه در ابتدا با زبان جاوا و برای توسعه برنامه های این زبان استفاده می شد. در ادامه با افزوده شدن افزونه هایی به آن امکان توسعه زبان هایی چون سی، سی++، روبی، کوبول، پایتون، پرل، پی اچ پی، لاک و آندروید را فراهم می کند.

۳-۱۸-۲-۱- معماری:

سامانه ی زمان اجرای اکلیپس مبتنی بر اکویناکس که پیاده سازی استاندارد، سازگار با او اس جی ای است، می باشد. اکلیپس از افزونه هارا برای فراهم آوردن امکان استفاده از تمام قابلیت هایش بر روی سامانه اجرا، استفاده می نماید که برخلاف با بعضی از برنامه هایی که به صورت معمول با قابلیت کد سخت می باشند، است.

ساز و کار افزونه های در اکلیپس صورتی سبک تر از چهارچوب مهندسی نرم افزار بر اساس مولفه است. به علاوه برای اینکه اکلیپس قابل گسترش برای توسعه سایر زبان های برنامه نویسی مانند سی و پایتون باشد، چهارچوب افزونه ها به اکلیپس امکان کار با زبان های حروفچینی مانند لاک و برنامه های شبکه ای مانند تلنت و سامانه مدیریت پایگاه داده را فراهم می آورد. جاوا و پشتیبانی از سی وی اس در کیت توسعه نرم افزاری اکلیپس فراهم شده است.

به جز بخش کوچکی از هسته زمان اجرا، تمام بخش‌های در اکلیپس به صورت افزونه‌ای است. این بدان معناست تمام افزونه‌های در اکلیپس به شکل دقیقاً یکسانی توسعه می‌یابند و تمام خصوصیات آنها با هم برابر است. اکلیپس برای بخش عظیمی از خصوصیات افزونه‌هایی را فراهم می‌نماید، که بعضی از آنها از طرف شخص ثالث توسعه به اشکال رایگان یا تجاری توسعه می‌یابند. نمونه‌ای از این افزونه‌ها، افزونه یو ام ال برای نمودار ترتیبی و سایر دیاگرام‌های یو ام ال است، یا افزونه‌ای برای کاوش در پایگاه داده.

۳-۱۸-۲-۲- سکوی غنی مشتری:

اکلیپس، سکوی غنی مشتری اکلیپس را برای توسعه برنامه‌های همه منظوره را فراهم می‌آورد. اجزا زیر این بستر را فراهم می‌آورد:

- اکویناکس - یک چهارچوب بسته بندی استاندارد
- هسته سکو - بوت اکلیپس ، اجرای افزونه ها
- جی فیس - کلاسهای نمایشگر برای انتقال حالت از مدل کنترل نمایشگر به اس دبلیو تی، بافر فایل ها ، ویرایشگر متن ها
- میزکار اکلیپس^۱ - نمایش ها ، ویرایشگرها ، چشم اندازها.

۳-۱۸-۲-۳- تاریخچه:

اکلیپس به عنوان پروژه‌ای در شرکت ای بی ام شعبه کانادا ایجاد گردید که برای جایگزینی محیط‌های توسعه خانواده ی ویژوال ایج بود و ویژوال ایج مبتنی بر اسمال تاک بود در حالی که اکلیپس توسط او تی ای بر مبنای جاوا توسعه یافت. در نوامبر سال ۲۰۰۱، کنسرسیومی برای توسعه اکلیپس به عنوان یک نرم افزار متن باز ایجاد شده که در سال ۲۰۰۴، بنیاد اکلیپس تاسیس شد.

نگارش سوم اکلیپس (منتشر شده در ۲۱ ژوئن سال ۲۰۰۴) سکوی خدماتی او اس جی ای را به عنوان معماری زمان اجرای خود برگزید. اکلیپس در ابتدا تحت مجوز متداول عمومی و بعداً تحت مجوز عمومی اکلیپس منتشر شد. نرم افزار آزاد و متن باز بعدها اعلام نمود که هر دو مجوز از نوع نرم افزار آزاد هستند، ولی با اجازه نامه عمومی همگانی گنو کاملاً مطابق نیستند.

1. Eclipse Workbench

براساس گفته لی نکمن، مدیر ارشد فناوری شرکت ای بی ام بخش رشنال در آن زمان و که بعدها رییس بخش توسعه و پشتیبانی این بخش شد، انتخاب نام اکلیپس، برای مقابله با ویژوال استودیو متعلق به شرکت مایکروسافت بود و نه شرکت سان میکروسیستم جالب آنکه وی اکنون کارمند شرکت مایکروسافت است.

۳-۱۸-۲-۴ نسخه ها:

نسخه	تاریخ انتشار	نگارش سکو
مارس	۲۵ ژوئن ۲۰۱۵ (تیر ۱۳۹۴) (برنامه ریزی شده برای انتشار)	4.5
لونا	۲۵ ژوئن ۲۰۱۴ (تیر ۱۳۹۳)	4.4
کیپر	۲۰ ژوئن ۲۰۱۳ (تیر ۱۳۹۲)	4.3
جونو	۲۷ ژوئن ۲۰۱۲ (تیر ۱۳۹۱)	4.2 و 3.8
ایندیگو	۲۰ ژوئن ۲۰۱۱ (تیر ۱۳۹۰)	3.7
هلیوس	۲۳ ژوئن ۲۰۱۰ (تیر ۱۳۸۹)	3.6
گالیه	۲۴ ژوئن ۲۰۰۹ (تیر ۱۳۸۸)	3.5
گانیمده	۲۵ ژوئن ۲۰۰۸ (تیر ۱۳۸۷)	3.4
اروپا	۲۹ ژوئن ۲۰۰۷ (تیر ۱۳۸۶)	3.3
کالیستو	۳۰ ژوئن ۲۰۰۶ (تیر ۱۳۸۵)	3.2
اکلیپس 3.1	۲۸ ژوئن ۲۰۰۵ (تیر ۱۳۸۴)	3.1
اکلیپس 3.0	۲۸ ژوئن ۲۰۰۴ (تیر ۱۳۸۳)	3.0

جدول ۳-نسخه های Eclipse

۳-۱۸-۲-۵ افزونه ها :

برای برنامه اکلیپس افزونه های بسیاری ساخته شده است که هریک برای زبان و اعمال خاصی طراحی شده است.

۳-۱۸-۲-۶ شروع کار :

ابتدا باید Android-SDK را که از قبل نصب کردید در درایو ویندوز کپی کنید. سپس برنامه Eclipse را اجرا کنید: (با این توضیح که Eclipse نیازی به نصب شدن ندارد و شما باید آنرا از اینترنت دانلود کنید و سپس از حالت zip خارج کنید)



شکل ۸- نرم افزار Eclipse

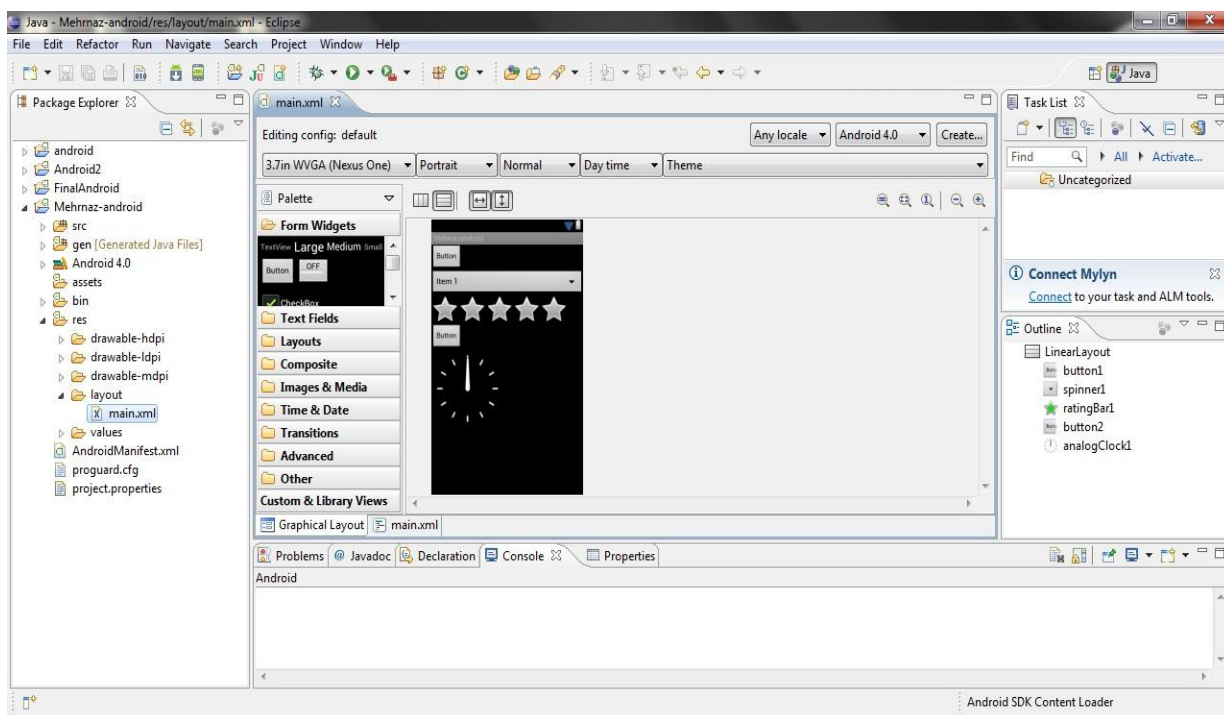
برای شروع کار مطابق زیر عمل کنید:

یک. به **Help → Install new software** بروید و پس از کلیک روی **Add** ، در **Name** نام دلخواه را وارد کنید و در قسمت **Location** مسیر **ADT** مورد نظر را که از قبل دانلود کردید انتخاب کنید.

دو. در پنجره ی **Available Software** تیک **Developer Tools** را انتخاب کنید و پس از آن **Next** را بزنید.

سه. سپس به **Window → Preferences** و شاخه ی **Android** بروید و مسر **SDK** را در **Location** وارد کنید.

چهار. سپس **File → New → Project → Android Project** را انتخاب کنید. پس از وارد کردن نام و انتخاب نسخه اندروید موردنظر برای برنامه ، Eclipse برای شروع کدنویسی آماده خواهد شد.



شکل ۹- محیط برنامه نویسی Eclipse

توسط **main.xml** در شاخه ی **layout** برنامه (مطابق شکل بالا) ، میتوانید با **Drag & Drop** به راحتی اجزای موردنظر را روی صفحه بکشید و کارهای گرافیکی برنامه خود را انجام دهید . به این صورت میتوانید نوشتن اولین برنامه خود را آغاز کنید.

پس از کدنویسی به **File → Export** بروید. سپس از شاخه **Android Export** ، **Application** را انتخاب کنید. پس از وارد کردن محل ذخیره سازی، نام و رمز موردنظر برای برنامه پنجره زیر نمایش داده میشود:

Export Android Application

Key Creation

⚠ A 25 year certificate validity is recommended.

Alias:

Password:

Confirm:

Validity (years):

First and Last Name:

Organizational Unit:

Organization:

City or Locality:

State or Province:

Country Code (XX):

شکل ۱۰- کادر ورود اطلاعات برای ساختن اپ

پس از وارد کردن اطلاعات مورد نظر و کلیک کردن روی **Next** ، برنامه با پسوند **.apk** ساخته خواهد شد.

۳-۱۸-۲-۷ نمونه های از برنامه های جاوا :

در زیر نمونه ای از برنامه ای که در جاوا نوشته شده است آورده شده است. البته برای کامپایل کردن این برنامه به کیت توسعه جاوا که قبلا نصب کرده اید نیاز دارید.

```
public class Test{  
    public static void main(String[] args) {  
        System.out.println("HelloWorld!");  
    }  
}
```

برای اجرای برنامه بالا، ابتدا باید یک فایل به نام Test.java ساخته شود و سپس کامپایل شود:

```
$ javac Test.java
```

سپس یک فایل خروجی به نام Test.class دریافت می شود. بعد با استفاده از دستور زیر برنامه قابل اجرا است:

```
$ java Test
```

مثال:

برنامه Hello world به این صورت در زبان جاوا می تواند نوشته شود:

```
// HelloWorld.java  
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

بر طبق قرارداد فایل ها بعد از کلاس های عمومی نام گذاری می شوند. سپس باید پسوند java را به این صورت اضافه کرد. Hello world.java: این فایل اول باید با استفاده از کامپایلر جاوا به بایت کد کامپایل شود. در نتیجه فایل Hello world.class ایجاد می شود. این فایل قابل اجرا است. فایل جاوا ممکن است فقط یک کلاس عمومی داشته باشد. اما می تواند شامل چندین کلاس با دستیابی عمومی کمتر باشد.

کلاسی که به صورت خصوصی تعریف می شود ممکن است در فایل java ذخیره شود. کامپایلر برای هر کلاسی که در فایل اصلی تعریف می شود یک کلاس فایل تولید می کند. که نام این کلاس فایل همانم کلاس است با پسوند class.

کلمه کلیدی public برای قسمت هایی که می توانند از کدهای کلاس های دیگر صدا زده بشوند، به کار برده می شود. کلمه ی کلیدی static در جلوی یک تابع، یک تابع ایستا را که فقط وابسته به کلاس است و نه قابل استفاده برای نمونه هایی از کلاس، نشان می دهد. فقط تابع های ایستا می توانند توسط اشیا بدون مرجع صدا زده شوند. داده های ایستا به متغیرهایی که ایستا نیستند، نمی توانند دسترسی داشته باشند.

کلمه ی کلیدی void نشان می دهد که تابع main هیچ مقداری را بر نمی گرداند. اگر برنامه ی جاوا بخواهد با خطا از برنامه خارج شود، باید system.exit() صدا زده شود. کلمه ی main یک کلمه کلیدی در زبان جاوا نیست. این نام واقعی تابعی است که جاوا برای فرستادن کنترل به برنامه، صدا می زند. برنامه جاوا ممکن است شامل چندین کلاس باشد که هر کدام دارای تابع main هستند.

تابع main باید آرایه ای از اشیا رشته ای را بپذیرد. تابع main می تواند از آرگومان های متغیر به شکل public static void main(string...args) استفاده کند که به تابع main اجازه می دهد اعدادی دلخواه از اشیا رشته ای را فراخوانی کند. پارامتر string[]args آرایه ای از اشیا رشته ایست که شامل تمام آرگومان هایی که به کلاس فرستاده می شود، است.

چاپ کردن، قسمتی از کتابخانه ی استاندارد جاوا است. کلاس سیستم یک فیلد استاتیک عمومی به نام out تعریف کرده است. شی out یک نمونه از کلاس printstream است و شامل تعداد زیادی تابع برای چاپ کردن اطلاعات در خروجی استاندارد است. همچنین شامل println(string) برای اضافه کردن یک خط جدید برای رشته ی فرستاده شده اضافه می کند.

۳-۱۸-۲-۸ کلاس های خاص:

برنامک (برنامه های کاربردی کوچک) :

اپلت جاواها برنامه هایی هستند که برای کاربردهایی نظیر نمایش در صفحات وب، ایجاد شده اند. واژه ی import باعث می شود کامپایلر جاوا کلاس های java.awt.Graphics و javaapplet.Applet را به کامپایلر برنامه اضافه کند. کلاس Hello کلاس Applet را توسعه می دهد. کلاس اپلت چارچوبی برای کاربردهای گروهی برای نمایش و کنترل چرخه ی زندگی اپلت، درست می کند. کلاس اپلت یک تابع پنجره ای مجرد است که برنامه های کوچکی با قابلیت نشان دادن واسط گرافیکی برای کاربر را فراهم می کند. کلاس Hello تابع موروثی print(Graphics) را از سوپر کلاس container باطل می کند، برای اینکه کدی که اپلت را نمایش می دهد، فراهم کند. تابع paint شی های گرافیکی را که شامل زمینه های گرافیکی هستند را می فرستد تا برای نمایش اپلت ها استفاده شوند. تابع paint برای نمایش "Hello world" تابع drawstring(string,int,int) را صدا می زند.

: JSP

صفحه ی سرور جاوا قسمتی از سرور javaEE است که پاسخ تولید می کند. نوعاً صفحات HTML به درخواست های HTTP از مشتری. JSP ها کد جاوا در صفحه ی HTML را با استفاده از حائل </and/> اضافه می کنند. JSP به javax.servlet کامپایل می شود.

جاوا سرولت :

تکنولوژی servlet جاوا گسترش وب را به آسانی فراهم می کند. و شامل مکانیزم هایی برای توسعه ی تابعی سرور وب و برای دسترسی به سیستم های تجاری موجود است. servlet قسمتی از javaEE است که به درخواست های مشتری پاسخ می دهد.

```
// Hello.java
import java.io. *;
import javax.servlet. *;

public class Hello extends GenericServlet {
    public void service(ServletRequest request, ServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html");
        final PrintWriter pw = response.getWriter();
        pw.println("Hello, world!");
        pw.close();
    }
}
```

واژه ی import کامپایلر جاوا را هدایت می‌کند که تمام کلاس‌های عمومی و واسط‌ها را از بسته‌های java.io و javax.servlet را در کامپایل وارد کند.

کلاس Hello کلاس GenericServlet را توسعه می‌دهد. کلاس GenericServlet واسطی برای سرور فراهم می‌کند تا درخواست را به servlet بفرستد و چرخه ی زندگی servlet را کنترل کند.

: Generics

قبل از کلاس‌های عمومی برای هر متغیر باید یک نوع خاص تعریف می‌کردیم. به عنوان مثال برای کلاس‌های ظرف این امر مشکل بود زیرا را آسانی برای ایجاد یک container وجود نداشت که نوع‌های خاصی از اشیا را بپذیرد. کلاس‌های عمومی اجازه می‌دهند نوع زمان کامپایل، بدون نیاز به ایجاد تعداد زیادی از container، چک شود. همه آنها کدهای مشابهی دارند.

: سوییگ

Swing کتابخانه ی واسط گرافیکی کاربر است برای پلت فرم javaSE. ابزاری مشابه پنجره، GTK و motif توسط شرکت sun فراهم شده‌اند. این مثال کاربرد swing یک پنجره ی واحد همراه با Hello world را ایجاد می‌کند.

```
// Hello.java (Java SE 5)
import java.awt.BorderLayout;
import javax.swing. *;

public class Hello extends JFrame {
    public Hello() {
        super("hello");
        setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());
        add(new JLabel("Hello, world!"));
        pack();
    }

    public static void main(String[] args) {
        new Hello().setVisible(true);
    }
}
```

اولین جمله ی `import` کامپایلر جاوا را هدایت می کند تا کلاس `BorderLayout` را از بسته ی `java.awt` در جاوا به کامپایل اضافه کند. و `import` دوم همه ُ کلاس های عمومی و واسط آن ها را از بسته ی `javax.swing` اضافه می کند. کلاس `Hello` کلاس `JFrame` را توسعه می دهد. کلاس `JFrame` یک پنجره با میله ُ عنوان و کنترل بستن است.

زمانی که برنامه آغاز می شود، تابع `main` با `JVM` صدا زده می شود. این یک نمونه ی جدید از کلاس `Hello` را ایجاد کرده و با صدا زدن تابع `setVisible(Boolean)` با مقدار `true` نمایش داده می شود.

معرفی پروژه : بازی آندروید "تست حافظه"

بسیار خب ، تا حالا با سیستم عامل آندروید و زبان برنامه نویسی جاوا آشنا شدید. حالا به عنوان نمونه شروع می کنیم پروژه ی خودم رو قدم به قدم پیش میبریم.

اولین قدم ، ایده ی بازی است. ایده ی بازی من از بازی های تست حافظه ای کوچک که شامل چندین جفت عکس بودند گرفته شده ، با این تفاوت که من یک عامل برای باخت هم در نظر گرفتم.

دومین قدم ، نوشتن یه سناریو برای بازی ست ، بازی چجوری شروع شه ، چجوری ادامه پیدا کنه ، کجا تموم بشه ؛ مثلا در اینجا بازی با ۹ خانه ی یک شکل شروع میشه ، اینجوری ادامه پیدا میکنه که شما به طور شانس ی باید خانه ها را یک به یک کلیک کنید و به خاطر بسپارید پشت هر خانه چه شکلی است و شکل های یکسان را با هم جفت کنید. اگر خانه ی بمب را کلیک کنید همانجا بازی تمام میشود و اگر بتوانید همه ی عکس ها را جفت کنید بدون اینکه رو بمب کلیک کنید بازی را برده اید و زمان شما به عنوان امتیاز بازی ثبت میشود تا اگر از بهترین زمان کمتر باشد ، جایگزین آن شود.

قدم بعدی ، شروع کردن به نوشتن و کد زدن برای بازی است و قدم آخر طراحی مراحل یا همان انتخاب پس زمینه و شکل ها و باقی مسائل مربوط به شکل ظاهری اپلیکیشن می باشد.

در اینجا چند عکس از مراحل مختلف بازی را برایتان می گذارم :

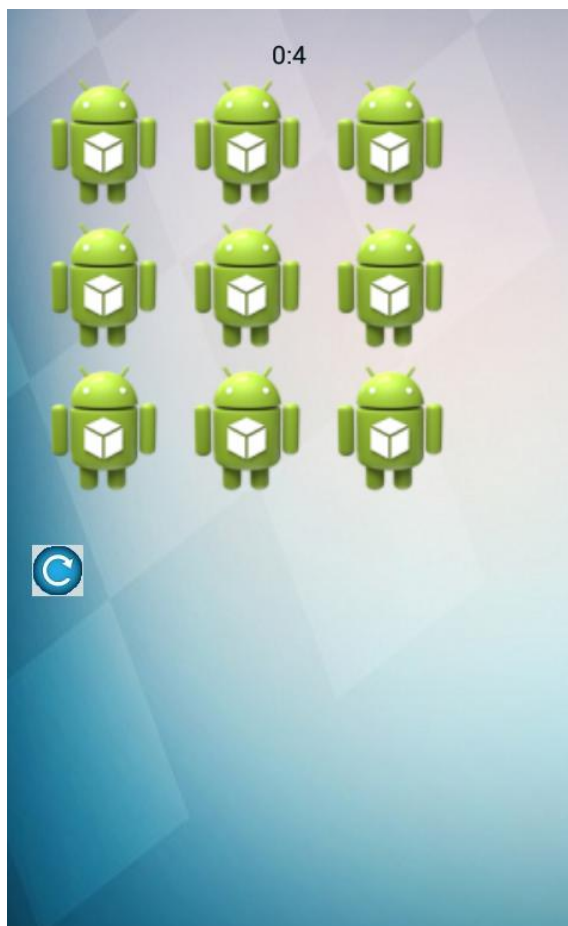


شکل ۱۱- صفحه ی خانه ی بازی

"صفحه خانه" بازی صفحه ی شروع بازی است که شامل دو گزینه ی Play برای رفتن به صفحه ی اصلی بازی و شروع آن و گزینه ی Exit برای خارج شدن از اپلیکیشن می باشد.

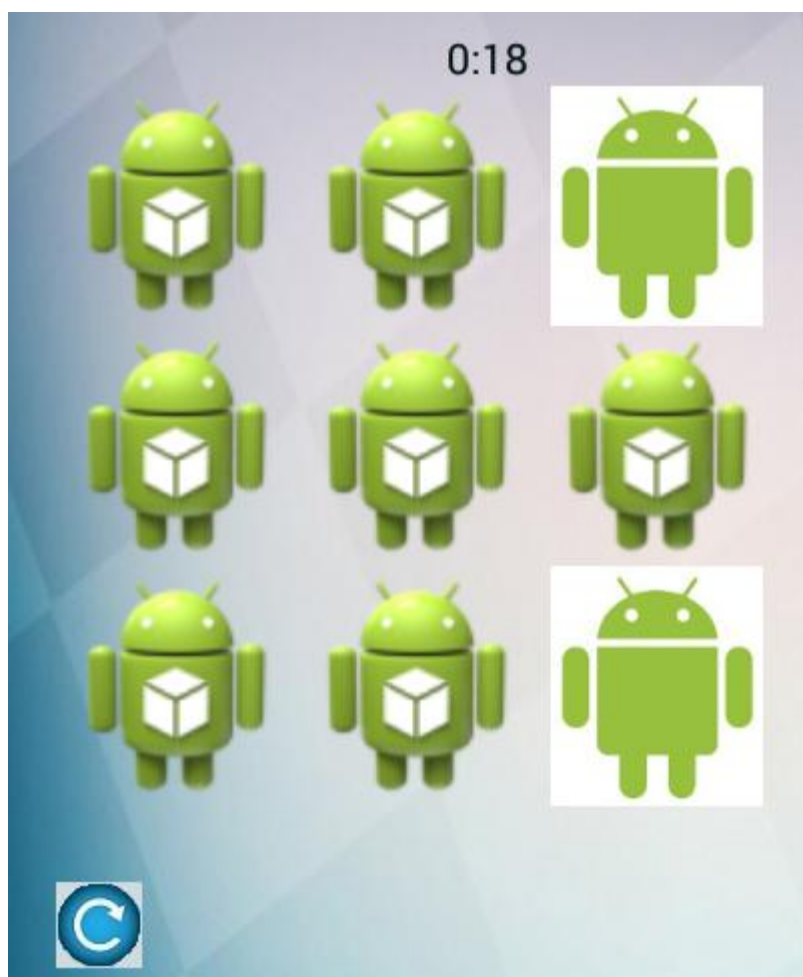
با زدن گزینه ی Play به صفحه ی زیر هدایت میشوید که در اینجا نه خانه ی بازی را مشاهده می کنید ، در پشت هر کدام از این خانه ها یک عکس قرار دارد که با کلیک کردن بر آن عکس شروع به چرخیدن میکند. در بالای Timer بازی را مشاهده میکنید که زمان پایان بازی را محاسبه میکند.

در پایین خانه ها هم دکمه Reload بازی برای بازگشتن به حالت اولیه قرار دارد.



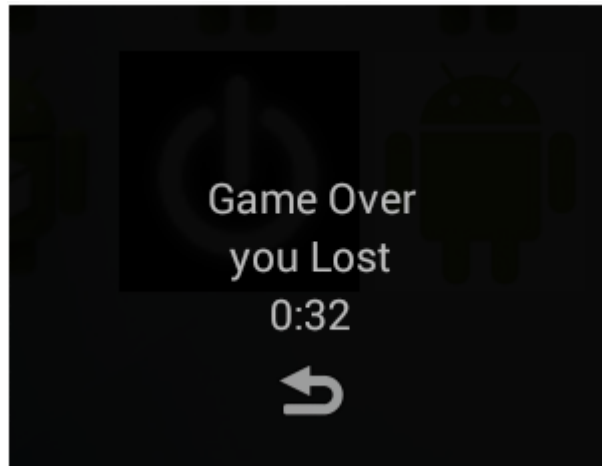
شکل ۱۲- صفحه ی اصلی بازی

خب ، همانطور که گفتم با کلیک کردن روی هر خانه ، آن عکس شروع به چرخیدن میکند و عکس پشت خود را نشان میدهد ، هدف بازی این است که مانند شکل زیر ، عکس های یکسان با هم جفت شوند تا همه عکس ها به غیر از بمب پیدا شود و بازی تمام شود.



شکل ۱۳ - هدف بازی

و در پایان یک پیغام به شما نشان داده میشود که حکایت از آن است که شما برنده یا بازنده شده اید. در عکس زیر پیغام تمام شدن را مشاهده میکنید ، که در آن میگوید شما بازی را باختید و در پایین پیغام زمان بهترین برد را میتوانید ببینید.



شکل ۱۴ - پیغام پایان بازی

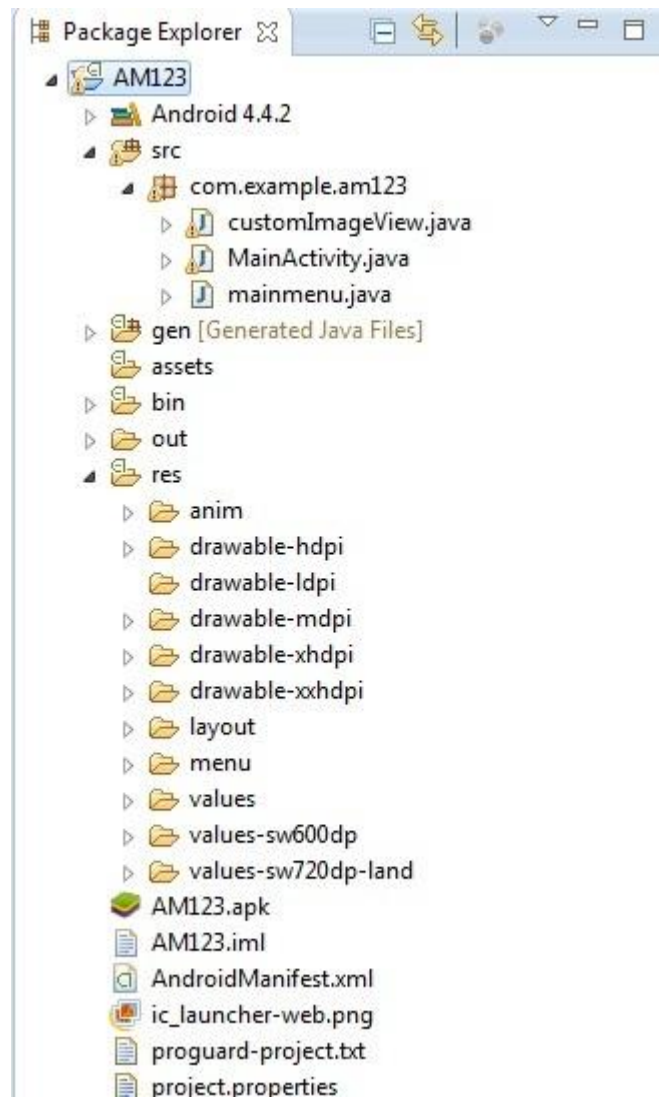
با زدن دکمه ی Reload پیغام بازی به حالت اولیه برمیگردد و زمان سنج صفر میشود. خوب ، چگونگی شروع ، ادامه و پایان بازی را مشاهده کردید ، حال در فصل آینده بصورت فنی به چگونگی ساخت بازی میپردازیم.

توضیحات فنی

و

کدها

برای شروع بهتر است بدانید که اکثر کار ما برای نوشتن این بازی ، با دو پوشه src و res میباشد.محتویات این دو پوشه را در شکل زیر میتوانید ببینید :



شکل ۱۵-محتویات پروژه

پوشه ی src برای ایجاد کلاس ها و نوشتن کدهای پروژه و پوشه ی res برای نوشتن Layout صفحات برنامه ، ذخیره ی عکس ها ، ذخیره ی اسم و آیکون برنامه و باقی خصوصیات ظاهری میباشد.

در ابتدا به کلاس ها و کدهای مربوط به آنها میپردازیم:

برای راحت کار کردن با کدها و اجرای عملیات ، کلاس `CustomImageView` را میسازیم که کلاس `ImageView` را `Extend` کرده و با رابط ^۱ `AnimationListener` از زیرکلاس `Animation` آن را ایمپلمنت ^۲ میکنیم:

```
public class customImageView extends ImageView implements Animation.AnimationListener
```

حال مقادیر زیر را نوشته و مقداردهی میکنیم:

```
Context context;  
boolean isMiddle=false;  
boolean showFace;  
int tile=R.drawable.ic_launcher;  
int face;  
private boolean freez=false;  
public boolean tempFreez=false;  
public customImageView(Context context) {  
    super(context);  
    this.context= context;  
}  
public customImageView(Context context, AttributeSet attrs, int defStyle, int mode){  
    super(context, attrs, defStyle);  
  
}  
public customImageView(Context context, AttributeSet attrs){  
    super(context,attrs);  
}
```

1. Interface

2. Implement

سه ساختمان^۱ CustomImageView نوشته شده برای آن است که بعداً بتوانیم از آن بصورت com.example.am123.customImageView به عنوان رابط بین XML و جاوا استفاده کنیم.

حال سه ساختمان کد زیر را به صورت Override برای استفاده از Animation و چرخش عکس هنگام کلیک شدن مینویسیم:

@Override

```
public void onAnimationStart(Animation animation) {  
}
```

@Override

```
public void onAnimationEnd(Animation animation) {  
    if (isMiddle){  
        if (showFace){  
            setImageResource(face);  
        }  
        else{  
            setImageResource(tile);  
        }  
        clearAnimation();  
        Animation anim2=AnimationUtils.loadAnimation(context,R.anim.flip2);  
        anim2.setAnimationListener(this);  
        setAnimation(anim2);  
        startAnimation(anim2);  
        isMiddle=false;  
    }  
    tempFreez=false;  
}
```

1. Constrature

@Override

```
public void onAnimationRepeat(Animation animation) {
```

```
}
```

در مورد دستور **OnAnimationEnd** ابتدا بررسی میکند آیا بلوک انتخاب شده در حالت وسط قرار دارد یا نه. اگر بود عکس مورد نظر تصادفی را جایگزین میکند و در غیر اینصورت عکس کاشی (عکس پشت ای که در حالت اولیه ما هر نه خانه را با آن عکس میبینیم) را قرار میدهد.

در ادامه دستور بارگذاری انیمیشن و حرکت کردن بلوک توسط کد **flip2** (کدهای **flip1** و **flip2** را در ادامه خواهیم دید) و بعد از آن **false** کردن "وسط بودن" بلوک و از **freeze** درآوردن باقی بلوک ها را مشاهده میکنیم. کد بالا برای زمانی وسط اجرا میباشد و کد زیر برای زمان از لحظه ی انتخاب شدن بلوک تا لحظه ای که به حالت وسط میرسد است :

```
public void startFlip(boolean showFace,int face){
```

```
    this.face=face;
```

```
    this.showFace=showFace;
```

```
    clearAnimation();
```

```
    Animation anim1= AnimationUtils.loadAnimation(context,R.anim.flip1);
```

```
    anim1.setAnimationListener(this);
```

```
    setAnimation(anim1);
```

```
    startAnimation(anim1);
```

```
    isMiddle=true;
```

```
}
```

دو دستور زیر نیز در رابطه با ثابت کردن^۱ بلوک ها میباشد :

```
public void setFreez(){
    freez=true;
}

public boolean isFreez(){
    return freez;
}
```

در پایین دو دستور flip را مشاهده میکنیم که این دستورها بصورت xml در پوشه ی anim زیرشاخه ی پوشه ی res پروژه ی ما قرار دارند. طرز کار این دستور به این صورت است که ما مقدار چرخش و زمان طول کشیدن را مشخص میکنیم؛ flip1 :

```
<scale
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXScale="1.0" android:toXScale="0.0"
    android:pivotX="50%"
    android:fromYScale="1.0" android:toYScale="1.0"
    android:pivotY="50%"
    android:duration="1000" />
```

flip2:

```
<scale
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:fromXScale="0.0" android:toXScale="1.0"
    android:pivotX="50%"
    android:fromYScale="1.0" android:toYScale="1.0"
    android:pivotY="50%"
    android:duration="1000" />
```

1. freeze

حال برای ادامه و درک بهتر کلاس mainmenu.java ما ابتدا فایل activity_mainmenu.xml را که در پوشه ی layout زیرشاخه ی res قرار دارد را بررسی میکنیم.فایل های xml این پوشه برای کارهای گرافیکی و ساختن نما و مشخص کردن اندازه ها میباشند.

کد زیر مشخص کننده ی اندازه ی کل صفحه ، نحوه دید بصورت عمودی یا افقی و عکس پس زمینه است:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:background="@drawable/background">
```

دو دستور زیر نیز مشخص کننده ی اندازه ی دکمه ها ، مقدار فاصله از عنصر بالایی ، ساختن یک شناسه برای ارتباط دادن با کلاس جاوا ، مکان قرار گرفتن در تصور (وسط یا گوشه) و عکس پس زمینه ی دکمه ها هستند:

```
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="150dp"
    android:id="@+id/playGameImageView"
    android:layout_gravity="center"
    android:background="@drawable/play"/>
```

```
<ImageView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="50dp"
    android:id="@+id/ExitImageView"
    android:layout_gravity="center"
    android:cropToPadding="false"
    android:background="@drawable/exit"/>
```

نکته : تفاوت مفهوم `wrap_contant` و `fill_parrent` در این است که در اولی اندازه ی عنصر به میزان لازم مقادیر داخل آن است و فضای بیشتری را اشغال نمیکند اما در `fill_parrent` اندازه ی عنصر به اندازه ی عنصر والد آن می باشد.

برای درک بهتر کدهای بالا بهتر است یک بار دیگر به شکل ۱۱-صفحه ی خانه ی بازی (صفحه ۵۷) مراجعه کنید.

اکنون به کدهای کلاس جاوای مربوط به صفحه ی خانه میپردازیم. یک کلاس میسازیم که کلاس `Activity` را گسترش دهد و دو شی `ImageView` برای دو دکمه ی صفحه میسازیم:

```
public class mainmenu extends Activity {
```

```
    ImageView playGameImageView;
```

```
    ImageView exitImageView;
```

در تابع زیر شی های ساخته شده را با شناسه ی فایل های `xml` بالا مرتبط میکنیم:

```
public void onCreate(Bundle savedInstanceState) {
```

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

```
    setContentView(R.layout.activity_mainmenu);
```

```
    playGameImageView=(ImageView)findViewById(R.id.playGameImageView);
```

```
    exitImageView=(ImageView)findViewById(R.id.ExitImageView);
```

و سپس دستور هرکدام را مینویسیم که به صفحه ی اصلی بازی برویم یا از بازی خارج شویم:

```
playGameImageView.setOnClickListener(new View.OnClickListener() {
```

```
    @Override
```

```
    public void onClick(View view) {
```

```
        startActivity(new Intent(mainmenu.this,MainActivity.class));
```

```
    }
```

```
});
```

```
exitImageView.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View arg0) {
        finish();
    }
});
```

حال میرسیم به کدهای اصلی بازی در کلاس MainActivity و فایل activity_main.xml. باز هم ابتدا به فایل xml میپردازیم، کد زیر مشخص کننده ی مقادیر اصلی صفحه، عکس پس زمینه و context مربوطه، فاصله عناصر داخلی و گرایش^۱ آن میباشد:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:background="@drawable/background"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    tools:context=".MainActivity"
    android:layout_gravity="center"
    android:gravity="center" >
```

کد LinearLayout زیر مقادیر کلی عناصر صفحه را مشخص میکند:

1. orientation

<LinearLayout

```
android:layout_width="fill_parent"
android:layout_height="fill_parent"
android:orientation="vertical"
    android:gravity="center"
    android:layout_gravity="center">
```

کد TextView زیر برای زمان سنج بازی و مشخصات اندازه و رنگ و ساختن شناسه آن است :

<TextView

```
android:id="@+id/timerTextView"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:textColor="#011210" />
```

کد LinearLayout زیر برای یک ردیف از خانه های صفحه اصلی بازی است که شامل سه بلوک میباشد و در آنها اندازه ردیف و بعد آن اندازه ی هر بلوک ، عکس کاشی (پشت بلوک) ، تگ آن و ارتباط دادن آن با CustomImageView نوشته شده است:

<LinearLayout

```
android:layout_width="fill_parent"
android:layout_height="wrap_content" >
<com.example.am123.customImageView android:id="@+id/image1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:src="@drawable/ic_launcher"
    android:tag="1"/>
```

کد بالا را برای هر نه خانه ی بازی و هر سه ردیف مینویسیم و سپس کد زیر را برای دکمه "restart" مینویسیم:

```

<LinearLayout android:id="@+id/panel"
    android:orientation="vertical"
    android:layout_height="fill_parent"
    android:layout_width="fill_parent">
    <Button android:id="@+id/restartButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:background="@drawable/reset"
        android:layout_marginTop="25dp"/>

</LinearLayout>

```

برای دیدن گرافیکی کدهای بالا به شکل ۱۲-صفحه ی اصلی بازی (صفحه ۵۸) مراجعه کنید.

حالا میرسیم به قسمت کدهای کلاس جاوا MainActivity. کلاس را ساخته و کلاس اصلی Activity را در آن گسترش میدهیم:

```

public class MainActivity extends Activity {

```

نُه شی از کلاس CustomImageView میسازیم:

```

customImageView image1;
customImageView image2;
customImageView image3;
customImageView image4;
customImageView image5;
customImageView image6;
customImageView image7;
customImageView image8;
customImageView image9;

```


مقادیر زیر را نیز برای استفاده های آتی میسازیم:

```

Button resartButton;

TextView timerTextView;

Handler handler=new Handler();

int[] bmpArray;

boolean isFirst=true;

customImageView tempCustomImageView;

int score=0;

int bombRes= drawable.bomb;

int time=0;

boolean isGameOver=false;

```

حال تابع زیر را مینویسیم که در آن به فایل `activity_main.xml` وصل شده ، سپس یک آرایه ی نه خانه ای ساخته و عکس های بازی را به هرخانه ی آن مقداردهی میکنیم تا سپس با تابع `randomize` بتوانیم آنها را درهم آمیخته کنیم.

دو تابع `InitView()` و `fitInScreen()` را هم که کدهای آنها را در ادامه خواهیم دید مینویسیم تا در تابع اولی هر شی ساخته شده ی `CustomImageView` را به یک خانه ی فایل `xml` صفحه اصلی بازی وصل کنیم. در تابع دومی نیز معین میکنیم که اندازه ی خانه ها در هر دستگاه چگونه باشد.

بعد از نحوه ی کار کردن زمان سنج را مشخص کرده ایم که در ابتدا زمان بازی 0 باشد و با پیشرفت بازی میزان زمان به دقیقه و ثانیه تبدیل میشود :

```

@Override

protected void onCreate(Bundle savedInstanceState) {

    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_main);

    bmpArray= new int[9];

```

```

        bmpArray[0]=drawable.a;
        bmpArray[1]=drawable.b;
        bmpArray[2]=drawable.c;
        bmpArray[3]=drawable.d;
        bmpArray[4]=drawable.a;
        bmpArray[5]=drawable.b;
        bmpArray[6]=drawable.c;
        bmpArray[7]=drawable.d;
        bmpArray[8]=drawable.bomb;

        bmpArray=randomize(bmpArray);
        InitView();
        fitInScreen();
        timerTextView.setText("0:0");
        handler.postDelayed(new Runnable() {
            @Override
            public void run() {
                time++;
                int min=time/60;
                int sec=time%60;
                timerTextView.setText(min+": "+sec);
                if(!isGameOver)
                    handler.postDelayed(this,1000);
            }
        },1000);
    }
}

```

نکته : استفاده از handler سه خاصیت دارد ؛ اول-میتواند کدی را با تاخیر اجرا کند. دوم-thread است ، یعنی زیرمجموعه ای از برنامه است . سوم-به context اصلی برنامه دسترسی دارد.

کد تابع randomize() :

```
private int[] randomize(int[] bmparr){  
    Random rgen = new Random(); // Random number generator  
  
    for (int i=0; i<bmparr.length; i++) {  
        int randomPosition = rgen.nextInt(bmparr.length);  
        int temp = bmparr[i];  
        bmparr[i] = bmparr[randomPosition];  
        bmparr[randomPosition] = temp;  
    }  
  
    return bmparr;  
}
```

کد تابع fitInScreen() :

```
private void fitInScreen(){  
    android.view.Display display = ((android.view.WindowManager)  
getSystemService(Context.WINDOW_SERVICE))  
        .getDefaultDisplay();  
  
    LinearLayout.LayoutParams llp1=new  
    LinearLayout.LayoutParams((display.getWidth()*25)/100, (display.getWidth()*25)/100);  
  
    llp1.gravity= Gravity.CENTER;  
  
    image1.setLayoutParams(llp1);  
    image2.setLayoutParams(llp1);
```

```

        image3.setLayoutParams(llp1);
        image4.setLayoutParams(llp1);
        image5.setLayoutParams(llp1);
        image6.setLayoutParams(llp1);
        image7.setLayoutParams(llp1);
        image8.setLayoutParams(llp1);
        image9.setLayoutParams(llp1);

    }

```

کد تابع InitView() :

```

private void InitView(){
    image1=(customImageView)findViewById(R.id.image1);
    image2=(customImageView)findViewById(R.id.image2);
    image3=(customImageView)findViewById(R.id.image3);
    image4=(customImageView)findViewById(R.id.image4);
    image5=(customImageView)findViewById(R.id.image5);
    image6=(customImageView)findViewById(R.id.image6);
    image7=(customImageView)findViewById(R.id.image7);
    image8=(customImageView)findViewById(R.id.image8);
    image9=(customImageView)findViewById(R.id.image9);

    image1.context=getBaseContext();
    image2.context=getBaseContext();
    image3.context=getBaseContext();
    image4.context=getBaseContext();
    image5.context=getBaseContext();
    image6.context=getBaseContext();

```

```
image7.context=getBaseContext();  
image8.context=getBaseContext();  
image9.context=getBaseContext();
```

```
image1.setOnClickListener(onclick());  
image2.setOnClickListener(onclick());  
image3.setOnClickListener(onclick());  
image4.setOnClickListener(onclick());  
image5.setOnClickListener(onclick());  
image6.setOnClickListener(onclick());  
image7.setOnClickListener(onclick());  
image8.setOnClickListener(onclick());  
image9.setOnClickListener(onclick());
```

```
resartButton=(Button)findViewById(R.id.restartButton);
```

```
resartButton.setOnClickListener(new OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        recreate();  
    }  
});
```

```
timerTextView=(TextView)findViewById(R.id.timerTextView);
```

```
}
```

از دیگر اتفاقاتی که در تابع بالا رخ داده اینست که context هر شی تصویر از context اصلی برنامه گرفته شده و هر شی تصویر موقع کلیک شدن تابع onclick() را فراخوانی میکند.

تابع recreate() در دستور دکمه ی restart نیز همانطور که از اسمش پیداست صفحه ی بازی را بارگذاری مجدد میکند.

حالا میرسیم به اصلی ترین قسمت کد برنامه ، جایی که ما کلیک را انجام میدهیم و برنامه بررسی میکند که آیا کلیک اول است یا دوم و درنهایت نتیجه ی آن کلیک ها چیست (برد ، باخت ، یا هیچکدام)

کد زیر را برای ساختن تابعی که اشیا تصویر ما روی آن setOnClickListener شده اند مینویسیم:

```
private OnClickListener onclick(){  
return new View.OnClickListener() {
```

تابع onclick() را بصورت Override مینویسیم:

```
public void onClick(final View arg0) {  
int imageRes=bmpArray[Integer.parseInt(((customImageView)arg0).getTag().toString())-1];
```

اتفاقی که در بالا رخ میدهد بدین شرح است که : بلوکی که روی آن کلیک کرده ایم در arg0 قرار میگیرد و تگ آن تصویر (که در فایل xml به هر تصویر دادیم) گرفته میشود ، چون آبجکت است به string تبدیل میشود و سپس آن را parse میکنیم به int و درنهایت یکی از آن کم میکنیم (چون آرایه ی ما از صفر شروع شده و تگ های ما از یک)

حالا کد زیر را برای وقتی که ما کلیک سوم را انجام داده ایم و به هیچ نتیجه ای نرسیده ایم ، مینویسیم.منظور از کلیک سوم اینست که ما عکس اول و دوم را انتخاب کردیم اما نه بردی بدست آورده ایم و نه بمب را مشاهده کرده ایم.پس لازم است تا عکس ها از حالت freeze دربیایند:

```
if (((customImageView)arg0).isFreez() || ((customImageView)arg0).tempFreez ||  
((customImageView)arg0).equals(tempCustomImageView) ) {  
return;  
}
```

در غیر اینصورت پس یا کلیک اول است یا کلیک دوم ، پس لازم است بلوک انتخاب شده شروع به چرخیدن کند:

```
((customImageView) arg0).startFlip(true,imageRes);
```

بررسی میکنیم آیا بلوک انتخاب شده "بمب" است یا خیر ، اگر بود بازی تمام شده است :

```
if (((customImageView)arg0).face==bombRes){  
    gameOver(true);  
    return;  
}
```

اگر بمب هم نبود ، حالا اگر کلیک اول باشد آن را در tempCustomImageView قرار میدهیم تا با کلیک بعدی آن را مقایسه کنیم ، سپس isFirst را در حالت false قرار میدهیم:

```
if (isFirst){  
tempCustomImageView=((customImageView) arg0);  
isFirst=false;  
}
```

و اگر کلیک دوم بود کد زیر را نوشته ایم تا دو کلیک انجام شده با هم مقایسه شوند و در صورت برابری freeze شوند و یک score اضافه شود.بعد بررسی میشود اگر score ما به 4 رسیده بود به معنای آنست که بازی با برد تمام شده و زمان نهایی برای ارزش گذاری ثبت میشود.در غیر اینصورت بازی ادامه پیدا میکند و باقی بلوک ها unfreeze میشوند و isFirst باز true میشود :

```
else{  
freezAll();  
int tempImageRes=bmpArray[Integer.parseInt(tempCustomImageView.getTag().toString())-1];  
if (tempImageRes==imageRes){  
    tempCustomImageView.setFreez();  
    ((customImageView) arg0).setFreez();  
    score++;  
}
```

```

new Timer().schedule(new TimerTask() {
    @Override
    public void run() {
        MainActivity.this.runOnUiThread(new Runnable() {
            @Override
            public void run() {
                unfreezAll();
            }
        });
    }
},2500);
if (score==4) gameOver(false);
else{
    new Timer().schedule(new TimerTask() {
        @Override
        public void run() {
            MainActivity.this.runOnUiThread(new Runnable() {
                @Override
                public void run() {
                    tempCustomImageView.startFlip(false, 0);
                    ((customImageView) arg0).startFlip(false, 0);
                    unfreezAll();
                }
            });
        }
    }, 2500);
}    isFirst=true; }

```


کدهای freeze و unfreeze کردن بلوک ها:

```
private void freezAll(){
    image1.tempFreez=true;
    image2.tempFreez=true;
    image3.tempFreez=true;
    image4.tempFreez=true;
    image5.tempFreez=true;
    image6.tempFreez=true;
    image7.tempFreez=true;
    image8.tempFreez=true;
    image9.tempFreez=true; }

private void unFreezAll(){
    image1.tempFreez=false;
    image2.tempFreez=false;
    image3.tempFreez=false;
    image4.tempFreez=false;
    image5.tempFreez=false;
    image6.tempFreez=false;
    image7.tempFreez=false;
    image8.tempFreez=false;
    image9.tempFreez=false;

    }
```

آخرین تابع بازی هم تابع gameOver() است که مربوط به پایان بازی و نشان دادن dialog می باشد. حال ابتدا به فایل dialog_layout.xml میپردازیم :

```

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="200dp"
    android:layout_height="100dp"
    android:orientation="vertical">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Game Over"
        android:id="@+id/gameOverTextView"
        android:layout_gravity="center_horizontal"/>
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="time"
        android:id="@+id/gameOverDialogTimeTextView"
        android:layout_gravity="center"/>
    <TextView android:id="@+id/bestScore"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"/>
    <ImageButton
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/restartButton"
        android:background="@android:drawable/ic_menu_revert"
        android:layout_gravity="center"/>
</LinearLayout>

```

مثل فایل های xml قبلی در اینجا لازم است از لحاظ گرافیکی دکمه ها را بسازیم و به آنها اندازه و در صورت نیاز تصویر یا شناسه ی جدید بدهیم. برای دیدن dialog box به شکل ۱۴-پیغام پایان بازی (صفحه ی ۶۰) مراجعه کنید.

حال دوباره به کلاس جاوا MainActivity برمیگردیم. تابع gameOver() را مینویسیم و هر کدام از اجزای آن را به قسمت مربوطه در xml مرتبط میکنیم:

```
private void gameOver(boolean lost){  
    isGameOver=true;  
    final Dialog dialog=new Dialog(this);  
    dialog.setContentView(R.layout.dialog_layout);  
    TextView  
    timeTextView=(TextView)dialog.findViewById(R.id.gameOverDialogTimeTextView);  
    TextView bestScoreTextView =(TextView)dialog.findViewById(R.id.bestScore);  
    ImageButton resetButton=(ImageButton)dialog.findViewById(R.id.restartButton);
```

یک SharedPreferences میسازیم تا به فضای در اختیار اپلیکیشن (خارج از SD Card) دسترسی داشته باشیم :

```
SharedPreferences shp =getSharedPreferences("score",MODE_WORLD_WRITEABLE);  
  
دستور مربوط به دکمه ی restart :
```

```
resetButton.setOnClickListener(new OnClickListener() {  
    @Override  
    public void onClick(View view) {  
  
        recreate();  
        dialog.dismiss();  
    }  
});
```

دستور مربوط به پایان بازی ، اگر بازی با باخت تمام شده بود ، پیغام "You Lost" نمایش داده میشود و اگر بازی را برده باشیم زمان برد محاسبه میشود و با "بهترین امتیاز" مقایسه میشود تا در صورت پایین تر و بهتر بودن زمان جایگزین آن شود سپس با پیغام "You Win! Your Time is" به شما نشان داده میشود:

```
if (lost){
    timeTextView.setText("you Lost");
}else{
    timeTextView.setText("you win. your time "+ time/60+"."+time%60);
    bestScoreTextView.setText("");
    if (shp.getInt("bestScore",0)>time){
        SharedPreferences.Editor shpeditor=shp.edit();
        shpeditor.putInt("bestScore",time).commit();
    }
}
```

محاسبات بهترین زمان و ذکر این نکته که اگر بهترین زمانی وجود نداشت نیازی به محاسبه نیست:

```
int x=shp.getInt("bestScore", 0);
if (x!=0)
    bestScoreTextView.setText(x/60+"."+x%60);
```

و در نهایت نشان دادن dialog :

```
dialog.show();
```

حالا نوشتن بازی را به پایان رسانده اید.

برای تعیین اسم و آیکون بازی : به فایل string.xml در پوشه ی Values زیرشاخه ی res بروید و کد زیر را بنویسید و اسم بازی تان را انتخاب کنید (در اینجا Memory Test نام بازی میباشد)

```
<string name="app_name">Memory Test</string>
```

برای آیکون بازی هم عکس دلخواه خود را در پوشه ی drawable-hdpi (که پوشه قرار گرفتن همه ی عکس های بکار رفته در بازی است) زیرشاخه ی res قرار دهید.

حالا به فایل AndroidManifest.xml در پکیج برنامه ی خود بروید و کدهای زیر را در تگ application بنویسید:

```
android:icon="@drawable/iconam"
```

```
android:label="@string/app_name"
```

و کد زیر را هم بنویسید تا نسخه ی سیستم عامل هدف و حداقل نسخه ی سیستم عامل برای اجرای بازی را مشخص کرده باشید.(البته برای تعیین نسخه سیستم عامل باید شماره sdk آن را بنویسید)

در اینجا حداقل سیستم عامل لازم برای نصب بازی نسخه 3.0 آندروید و نسخه ی هدف بازی آندروید 4.4 میباشد :

```
<uses-sdk
```

```
    android:minSdkVersion="11"
```

```
    android:targetSdkVersion="19" />
```

اکنون شما تمام قسمت های مربوط کدزنی و ساختن بازی را گذرانده اید ، حالا به آخرین قسمت پروژه که ساختن فایل نصب برنامه با پسوند apk. هست میرویم :

روی بسته (Package) اصلی پروژه راست-کلیک کنید و گزینه ی Export را بزنید.پوشه ی Android را باز کنید و گزینه ی Export Android Application را انتخاب کنید و به صفحه ی بعد بروید . در این صفحه برای بررسی پروژه ایست که میخواهید از آن خروجی بگیرید. نام پروژه ی شما در کادر نوشته شده است پس به صفحه ی بعد بروید.

برای ساخت فایل apk نیاز به یک keystore دارید ؛ اگر از پیش ساخته اید مکان آن را انتخاب کنید و رمزعبور خود را بنویسید و به صفحه ی بعد بروید. اگر هم keystore ندارید یکی بسازید.

*برای ساخت keystore در صفحه ی بعد باید نام و رمزعبور ، مقدار اعتبار و حداقل یکی از خانه های مشخصات دلخواه (مثل اسم و فامیل) را پر کنید.

در صفحه ی آخر مسیر ایجاد فایل را انتخاب کنید و Finish را بزنید.

منابع:

1. آندروید برای برنامه نویسان – مولفان : جیمز استیل، نلسون تو – مترجم: احمد رضا بقالی

2. درگاه ویکی پدیا

3. [Android.developer.com](https://androiddeveloper.com)

4. [Github.com](https://github.com)

5. [Wrox java learning for android developers](https://www.wrox.com)

Abstract:

This documentary is for an android game's project. This project has been written in java language in an eclipse environment for android operating systems.

For this reason in this documentary there are first some short explanations about java programming language and then about eclipse environment and finally game building codes.

The minimum operating system's versions for operating this android game are "honeycomb" and game's objective version "jelly bean".

In this game you will face with nine blocks that depending on chance and memory must choose similar photos and take care of bomb. If you choose four similar photos you will have won the game.

Keywords: Game, Operating System, Android, Programming Language, Java.



Allameh Dekhoda Higher Education Institute

**DEPARTMENT OF COMPUTER ENGINEERING
“B.Sc” THESIS**

**SUBJECT:
Andriod Game: Memory Test**

**THESIS ADVISOR:
Mr.Khazaei**

**BY:
Mohammad Amin Amini**

September 2014