دانشکده‌ی فنی و مهندسی

پروژه برای درس نصب و راه اندازی شبکه

در رشته‌ی مهندسی کامپیوتر

پروژه چت با Socket Programing

سید مرتضی حسینی

جابر بابکی

استاد راهنما:

مهندس براری تاری

۱۳۹۴


بسم الله الرحمن الرحيم

فهرست مطالب

فصل ۱

مقدمه

۱-۱ مقدمه

جهت تولید اپلیکیشن تحت سیستم عامل اندروید[۱] همانند توسعه نرم‌افزار در سایر پلت‌فرم‌ها نیاز است تا بعد از فراهم کردن مقدمات توسعه، تکنیک‌ها و استراتژی‌های مهندسی نرم‌افزار در تمام طول فرایند تولید اپلیکیشن رعایت شود.

[۱] Android

فصل ۲

مفاهیم پایه‌ای

۲-۱ توضیحات و تاریخچه اندروید

۲ ۱ ۱ معرفی سیستم عامل اندروید

سیستم عامل اندروید به منزله اولین سیستم عامل جامع و کامل موبایل است که به صورت Open Source[1] منتشر شـده است. اندروید بر پایه هسته لینوکس ساخته شده است و بیشترین استفاده را در بین سیستم عامل های موبایل دارد.

۲ ۱ ۲ تاریخچه اندروید

در اوت ۲۰۰۵، گوگل شـــرکت کوچک اندروید که توســـط اندی رابین و همکارانش پایه‌گذاری شـــده بود و در زمینه تولید نرم‌افزار و برنامه‌های کاربردی برای تلفن‌های همراه فعالیت می‌کرد را خریداری نمود، اما این موضـــوع را اعلام نکرد تا در نیمه دوم سـال ۲۰۰۷ که رسـما اعلام کرد، قرار اسـت سیستم عامل متن باز برای تلفن‌های همراه منتشر کند. در نوامبر سال ۲۰۰۷ گروهی از تولید کنندگان مطرح تجهیزات الکترونیکی همچون موبایل و تجهیزات بی سیم به علاوه یکسـری از شرکت ها نرم افزاری که از آن جمله می توان به گوگل اشاره کرد که در ابتدا ۳۴ شرکت بودند و سـال به سال به تعداد اعضا افزوده شد، در کنار یکدیگر جمع شده و کنسرسیومی را تشکیل دادند که هدف آن ایجاد یک سـیسـتم عامل موبایل Open Source بود تا از این طریق بتوانند هزینه های تولید اپلیکیشـن های موبایل را کاهش دهند. در حقیقت از آنجا که شـرکت گوگل در این کنسـرسـیوم رویکردی نرم افزاری داشـت و از سـوی دیگر صـاحب اصلی این سیستم عامل هم محسوب می شد، از این رو به عنوان رهبر اصلی این سیستم عامل جدید انتخاب گردید.

[1] به شرایطی گفته می شود که چیزی به صورت رایگان در دسترس عموم قرار گیرد که هم بتوانند از آن استفاده و هم بسته به نیازهای خود تغییراتی در آن ایجاد کنند.

۲ ۱ ۳ معرفی کنسرسیوم توسعه اندروید

برای این منظور این کنسرسیوم که Open Handset Alliance نام داشت چیزی تحت عنوان SDK[۱] که در برگیرنده ابزارهای نرم افزاری مورد نیاز برای توسعه اپلیکیشن های اندروید بود را منتشر کرد. از سوی دیگر سایت http://developer.android.com توسط برنامه نویسان شرکت گوگل همواره برای ارائه آخرین نسخه های SDK، دانلود محیط های برنامه نویسی و همچنین مستندات این سیستم عامل به روز نگه داشته می شود.

دوران اندروید به طور رسمی از ۲۲ اکتبر ۲۰۰۸ میلادی، زمانی که گوشی T-Mobile G1 توسط HTC در ایالات متحده عرضه شد، آغاز گردید. در ابتدای امر، بسیاری از قابلیت‌هایی که نمی‌توان اندروید را بدون آنها متصور بود در این گوشی وجود نداشت. برای نمونه می‌توان به فقدان صفحه کلید مجازی (لمسی)، قابلیت چند لمسی، برنامه‌های کاربردی حرفه‌ای و... اشاره کرد، اما همین گوشی سنگ‌بنایی شد تا امروزه با اندرویدی چنین پیشرفته سروکار داشته باشیم. از آن سال به بعد شرکت های سازنده تلفن های همراه یکی پس از دیگری به این سیستم عامل روی آوردند و نه تنها تلفن های همراهی مجهز به این سیستم عامل طراحی کردند، بلکه ابزارهای الکترونیکی دیگری همچون تبلت، ساعت، کتاب خوان الکترونیک و حتی تلویزیون هم با این سیستم عامل طراحی و روانه بازارکردند.

۲ ۱ ۴ زبان برنامه نویسی سیستم عامل اندروید

زبان برنامه نویسی اصلی این سیستم عامل زبان جاوا و به طور حتم علت این بوده است که جاوا بسیار قدرتمند بوده و در عین حال Open Source نیز می باشد. اما این بدان معنا نیست که برنامه نویسانی با تسلط به دیگر زبان های برنامه نویسی نمی توانند برای این سیستم عامل اپلیکیشن طراحی کنند. در حقیقت کسانیکه که به HTML5 و JavaScript تسلط دارند خواهند توانست برای این سیستم عامل اپلیکیشن های تحت وب طراحی کنند. از سوی دیگر برنامه نویسان سی و همچنین سی پلاس پلاس می توانند با استفاده از NDK[۲]اقدام به توسعه اپلیکیشن های اندروید با استفاده از زبان های فوق الذکر نمایند.

تسلط به زبان برنامه نویسی جاوا برای کسب نتیجه بهتر در این سری از آموزش های اندروید یک امر ضروری محسوب می شود و آشنایی با XML یک مزیت است.

---

[۱] Software Development Kit به معنی پیکیج توسعه نرم افزار می باشد

[۲] Native Development Kit

۲ ۱ ۵ معرفی نسخه های مختلف اندروید

نسـخه اول سیستم عامل اندروید که ۱ بود نام خاصی نداشت و بیشتر کارایی اینترنتی داشت و در عمل توان رقابت با هیچ سیستم عاملی را نداشت.

نخسـتین بروزرسانی اندروید سه ماه پس از عرضه گوشی G1 در فوریه ۲۰۰۹ میلادی صورت پذیرفت. نسخه ۱,۱ را نمی‌توان به هیچ وجه محصـولی نوآورانه دانست؛ بلکه بیشتر به مثابه وصله‌های نرم‌افزاری برای بهینه‌سازی سیستم و رفع باگ‌های آن بود. از طرف دیگر این نسخه توانایی اندروید را برای بروزرسانی بی دردسر به خوبی نشان داد. همین قابلیت پیش‌پا افتاده در زمان عرضـــه نســـخه‌ی ۱,۱ یکی از نقاط قوت اندروید به حســـاب می‌آمد که تا به امروز نیز همچنان ادامه دارد، چرا که سایر سیستم‌عامل‌های موبایل فاقد قابلیت مشابهی بودند.

بسـیاری از ما اندروید ۱,۵ را بیشتر به نام Cupcake به معنی کیک‌یزدی (کیک فنجانی) می‌شناسیم، سیستم‌عاملی که حقیقتاً نقطه‌ی عطفی در تاریخچه‌ی اندروید به شمار می‌رود. در این نسخه برخی از قابلیت‌های کلیدی به اندروید افزوده شـد. از این گذشته اندروید ۱,۵ اولین نسخه از این سیستم‌عامل بود که از روش نامگذاری ویژه‌ی گوگل که تا حدودی عجیب ولی در عین حال جالب توجه اسـت بهره‌مند گردید. نسـخه های مختلف بر اساس نام دسرهای میان وعده نامگذاری شـــده اند. از آن پس بود که هر بروزرســـانی کلی با نام یکی از شـــیرینی‌جات (البته به ترتیب حروف الفبای لاتین) همراه می‌شد تا بر دلچسبی اندروید بیافزاید.

از جنبه‌های گوناگون اندروید کیک یزدی را می‌توان گامی بزرگ در جهت اصـلاح و بهبود این سـیستم عامل دانست. هم از نظر تکنیکی-فنی و هم از منظر زیبایی‌شناختی که رابط‌کاربری آن با تغییرات مثبتی همراه شد. بسیاری از این تغییرات آنچنان محسوس نبودند، برای نمونه ویجت جستجوی گوگل از این پس از ظاهری شفاف بهره‌مند شد.

نسـخه ۱,۶ که Donut به معنی پیراشـکی نام داشـت پس از نسـخه ۱,۵ انتشـار یافت. در این نسـخه شـاهد بروز بهبودهـای جزئی دیگری در کل پلتفرم هســـتیم که قابلیت‌های جدیدی را برای کاربران به ارمغان آورد. البته بخش بزرگی از این تغییرات مربوط به مواردی بود که در پشت‌پرده اتفاق می‌افتاد. برای نمونه نخستین بار در تاریخ اندروید پشـتیبانی از[۱] CDMA از این نسـخه میسـر گردید و بدین ترتیب شرکت‌هایی همچون Verizon توانستند به بازار امریکا و آسیا دست یافته و سود سرشاری را نصیب خود و گوگل کنند. پیراشکی برای نخستین بار موفق شد اندروید را قادر سازد تا در انواع صفحات با وضوح و نسبت نمایش گوناگون ظاهر شود.

در اوایل نوامبر ۲۰۰۹ میلادی، تقریباً یک سـال پس از عرضـه گوشـیG1 ، اندروید ۲,۰ عرضـه شد. «بزرگ» صفت خوبی برای توصـیف نان خامه‌ای یا Eclair گوگل بود چرا که این نسـخه از سیستم‌عامل اندروید دستاوردی بزرگ با وعده‌هایی بزرگ بود که بر روی گوشـی‌های بزرگ کمپانی‌های بزرگ عرضـه می‌شد! در این نسخه بیشترین تغییرات

[۱] Code division multiple access

بر روی سیستم‌عامل گوگل هم از نظر بصری و هم معماری داخلی صورت گرفت. برنامه‌ی رایگان ناوبری به همراه اندروید ۲,۰ عرضه شد که از اطلاعات نقشه گوگل برای راهنمایی و هدایت کاربر استفاده می‌کرد.

تغییرات گسترده دیگری در نسخه ۲,۱ اندروید صورت گرفت. این نسخه از سیستم عامل گوگل آنچنان در زمان خود خوب بود که بسیاری از گوشی‌های جدید به آن مجهز شدند. از این گذشته بسیاری از ایرادات نرم‌افزاری دیگر نیز در این نسخه رفع گردیدند، با این حال همانند اندروید ۲,۰ نسخه ۲,۱ نیز از همان نام نان خامه‌ای بهره می‌برد.

اندروید ۲,۲ معروف به ماست بستنی یا همان Froyo در اواسط سال ۲۰۱۰ میلادی عرضه شد. گوگل در نسخه ماست بستنی قابلیت‌های بیشتری را به نمایش گذاشت. از همان ابتدا که کاربر گوشی را روشن می‌کند، صفحه‌ی خانگی با طراحی جدید چشم‌نواز است. بخش گالری اندروید ماست بستنی به طور کامل بازطراحی شده بود و برای نخستین بار قابلیت‌های سه‌بعدی این پلتفرم را به نمایش گذاشته بود. برای نمونه با کج کردن صفحه یا حرکت بین آلبوم‌های گوناگون و عکس‌ها، انیمیشن‌های با کیفیتی نمایش داده می‌شد.

حدوداً شش ماه پس از عرضه ماست بستنی گوگل کوشید تا نسخه ۲,۳ این سیستم عامل را با نام نان زنجبیلی یا Gingerbread عرضه نماید. نان زنجبیلی از جنبه‌های گوناگون یک بروزرسانی کوچک به حساب می‌آمد ولی تعدد این تغییرات کوچک تمایزات بسیار زیادی را موجب شدند. نان زنجبیلی را می‌توان حائز بیشترین تغییرات ظاهری از نسخه نان خامه‌ای به حساب آورد. این تغییرات به ظاهر کوچک تاثیر بسیار بزرگی بر ظاهر این پلتفرم داشتند. نسبت به نسخه‌های قبلی، نان زنجبیلی تمیزتر و مدرن‌تر به نظر می‌آمد، اما در حقیقت این ترفند هوشمندانه‌ی گوگل برای کاهش مصرف باتری بود چرا که آن همه اجزای روشن در صفحه نمایش انرژی زیادی را طلب می‌کردند.

اندروید نسخه ۳ را می‌توان پدیده‌ای عجیب و کم نظیر دانست. سیستم عاملی که در حقیقت برای گوشی‌های هوشمند تلفن همراه نوشته شده بود، حالا توجه خود را به تبلت‌ها معطوف ساخته بود. برای همین منظور گوگل نسخه ۳ تحت عنوان Honeycomb به معنی شانه عسل را معرفی کرد.

سپس نسخه ۴,۰ با نام ساندویچ بستنی یا Ice Cream Sandwich را می‌توان بی شک بزرگترین تغییر اندروید در حوزه گوشی‌های هوشمند به شمار آورد.گوگل برای ارائه حس بهتر به کاربرانش در این نسخه از فونت تازه‌ای با نام Roboto استفاده کرده است. یکی از خصوصیات ویژه‌ی اندروید همان پنجره‌ی اعلانات است که با وجود تغییر و بروزرسانی در طراحی هنوز یکی از ویژگی‌های مهم این سیستم عامل به شمار می‌رود. در نسخه‌های پیشین کاربر تنها می‌توانست کلیه اطلاع‌رسانی‌های نمایش داده شده را پاک کند، در حالی که در این نسخه می‌تواند موارد دلخواه را پاک کرده یا اینکه کلیه آنها را از صفحه حذف کند.

گوگل خیلی بی سر و صدا مشغول بهبود صفحه کلید مجازی خود از زمان ارائه در نسخه کیک یزدی به بعد بود و در حقیقت می‌توان این مورد را یکی از مهمترین پیشرفت‌های نسخه ساندویچ بستنی به حساب آورد. از این پس بود که وارد کردن متن، پشتیبانی از کلیپ‌بورد و کیفیت صفحه کلید مجازی اینچنین پیشرفته شد.

گوگل در سال ۲۰۱۲ از نسخه ۴,۱ سیستم عامل اندروید خود پرده برداری کرد. قابلیت‌های این نسخه نیز بسیار فراتر از آن عدد ۰,۱ است و می‌توان به نسبت نسخه ساندویچ بستنی ویژگی‌های بسیار بیشتری را در آن یافت. آبنبات پاستیلی یا Jelly Bean را می‌توان بازتعریف استراتژی گوگل در بازار تبلت‌ها دانست.

گوگل مدعی است که قابلیت‌های بصری و لمسی این سیستم عامل را با بکارگیری واحد گرافیک سه‌گانه ارتقا داده و بدین ترتیب بازنگاری کلیه تصاویر گرافیکی را به ۱۶ میلی ثانیه تقلیل داده است. این برنامه در جهت حرکت نرم و روان در بین منوها و صفحات صورت پذیرفت، به «پروژه کَره» معروف شد.

بی‌شک Google Now یکی از بزرگ‌ترین و مهم‌ترین بلندپروازی‌های گوگل است. اندروید هم‌اکنون به عنوان یک پلتفرم کامل محسوب می‌شود که قادر به پردازش انواع داده‌ها، زمانبدی امور، مکان، زمان و... است.

این سیستم عامل همچنین با بهره از زبان طبیعی برای جستجوهای خود و استفاده از طبیعی‌ترین سیستم گفتار به متن موجود بیشترین کارایی را به کاربران خویش بخشیده است. با آبنبات پاستیلی، برای نخستین بار کاربر حتی می‌توانند به صورت آفلاین نیز از قابلیت گفتار به متن بهره ببرد و این بدان معنی است که دیگر برای این منظور نیازی به اتصال به شبکه اینترنت نیست.

اکتبر ۲۰۱۳ زمان معرفی نسخه کاملا جدید و متفاوتی از سیستم عامل خوشمزه اندروید با نام KitKat یا کیت کت بود .نسته، تولیدکننده بیسکوئیت‌های کاکائویی مشهور کیت‌کت به همکاری با گوگل پرداخت و این دو شرکت کمپین‌های تبلیغاتی مشترک بسیاری را نیز برپا نمودند.

اندروید ۴,۴ مهم‌ترین تحول در ظاهر رابط کاربری سیستم عامل اندروید از زمان ارائه اندروید ۴ ساندویچ بستنی، محسوب می‌شد. تم آبی رنگی که در نسخه‌های مختلف آبنبات پاستیلی شاهد آن بودیم به رنگ سفید تغییر کرده بود. آیکون‌ها بزرگ‌تر شده و نوار اطلاع‌رسانی‌ها و دکمه‌های لمسی کنترلی نیز از پس‌زمینه‌ای شفاف بهره می‌بردند.

نوامبر ۲۰۱۴ نسخه نهایی اندروید آبنبات چوبی یاLollipop معرفی شد. اندروید آبنبات چوبی یک نسخه انقلابی از سیستم عامل موبایل گوگل بود. اصلی‌ترین این تغییرات مربوط به طراحی رابط کاربری و به کارگیری زبان طراحی جدیدی تحت عنوان طراحی متریال بود. در نسخه آبنبات چوبی اندروید، تنظیمات سریع و همچنین ظاهر نوار اطلاع‌رسانی ها و تنظیمات سریع به کلی تغییر کرده است. در صورتی که انگشت خود را از بالای صفحه به سمت پایین بکشید نوار نوتیفیکیشن‌ها مواجه خواهید شد. برای دسترسی به منوی تنظیمات سریع باید یک بار دیگر منو را به سمت پایی لمس کنید تا در زیر اطلاع‌رسانی‌ها تنظیمات مهم را مشاهده کنید.تم رنگی این بخش از مشکی به طوسی و سبز تغییر کرده.

نکته دیگری که در مورد نحوه نامگذاری نسخه های مختلف اندروید می بایست مد نظر قرار داده شود این است که نام دسرهای انتخابی بر اساس حروف الفبا پیش می روند. به طور مثال همانطور که در الفبای انگلیسی حرف D قبل از حرف E قرار می گیرد، نام انتخابی برای نسخه ۱,۵ معادل با Donut بود و پس از آن در نسخه ۱,۶ نام Eclair

انتخـاب شـــد(بر اســـاس حروف الفبـا کلمه Donut پیش از کلمه Eclair قرار می گیرد). در جدول زیر اطلاعات مختصری از نسخه های مختلف سیستم عامل اندروید ارائه شده:

Android Version	**Version Name**	**API Level**
Android 4.4	KitKat	19
Android 4.3	Jelly Bean	18
Android 4.2	Jelly Bean	17
Android 4.1	Jelly Bean	16
Android 4.0.3 – 4.0.4	Ice Cream Sandwich	15
Android 4.0 – 4.0.2	Ice Cream Sandwich	14
Android 3.2	Honeycomb	13
Android 3.1	Honeycomb	12
Android 3.0	Honeycomb	11
Android 2.3.3 – 2.3.7	Gingerbread	10
Android 2.3 – 2.3.2	Gingerbread	9
Android 2.2 – 2.2.3	Froyo	8
Android 2.1	Éclair	7
Android 2.0.1	Éclair	6
Android 2.0	Éclair	5
Android 1.6	Donut	4
Android 1.5	Cupcake	3
Android 1.1	Banana bread	2
Android 1.0	Apple pie	1

جدول (۲-۱) نسخه‌های اندروید

۲ ۱ ۶ محیط برنامه نویسی اکلیپس

برای شروع برنامه نویسی برای اندروید می توان به چند طریق اقدام کرد که در اینجا به طور خلاصه به چند مورد می پردازیم:

معروف ترین محیط برنامه نویسی برای توسعه اپلیکیشن های تحت سیستم عامل اندروید محیط برنامه نویسی اکلیپس به همراه ابزارهای SDK وADT[۱] است. از آنجا که محیط برنامه نویسی اکلیپس Open Source است شرکت گوگل تمایل به مراتب بیشتری برای فراهم آوردن ابزارهای توسعه اندروید برای این محیط برنامه نویسی قدرتمند از خود نشان داده است.

۲ ۱ ۷ نسخه ADT Bundle محیط برنامه نویسی اکلیپس

این محیط برنامه نویسی دقیقاً همان محیط برنامه نویسی اکلیپس است با این تفاوت که شرکت گوگل خود تغییراتی در آن صورت داده و آن را منتشر کرده است. این محیط برنامه نویسی بر خلاف اکلیپس که برای توسعه اندروید می بایست تنظیمات خاصی در آن انجام داد، در برگیرندهٔ کلیه ابزارهای لازم برای توسعه اندروید از جمله SDK و ADTو Emulator بوده و کاربران دیگر نیازی به دانلود کردن این ابزارها به صورت مجزا نخواهند داشت. لازم به ذکر است توسعه دهندگانی که با نحوه کار با محیط برنامه نویسی اکلیپس آشنایی داشته باشند به راحتی خواهند توانست از این محیط هم در تولید اپلیکیشن اندروید استفاده کنند.

۲ ۱ ۸ محیط برنامه نویسی اندروید استودیو

اندروید استودیو [۲] محیطی برای توسعه برنامه های اندرویدی است که برای اولین بار خود شرکت گوگل بر پایه محیط برنامه نویسی قدرتمند IntelliJ IDEA طراحی کرده است که همانند اکلیپس ابزارهای لازم برای طراحی، توسعه و مشکل یابی برنامه ها را داراست. اولین نسخه از اندروید استودیو در کنفرانس سالیانه گوگل که در اردیبهشت ماه سال ۱۳۹۱ اتفاق افتاد تحت عنوان نسخه۱ معرفی شد.

---

[۱] Android Development Tools

[۲] Android Studio

۲-۲ مقدمات توسعه اپلیکیشن اندروید

۲ ۲ ۱ روش نصب JDK جاوا

جهت توسعه اپلیکیشن تحت سیستم عامل اندروید نیاز به JRE [۱] و JDK [۲]ست که می توان آخرین نسخه آن ها را از وبسایت اوراکل[۳] دریافت کنید. لازم به ذکر است چنانچه شما نسخه JDK نصب نمایید دیگر نیازی به نصب JREنخواهید داشت چرا که JDK در برگیرنده JRE نیز می باشد.

۲ ۲ ۲ معرفی SDK و ADT اندروید

پس از دریافت و نصب اکلیپس و همچنین دریافت و اضافه نمودن SDK اندروید و پلاگین ADT خواهیم توانست برنامه نویسی اندروید را شروع کنیم.
به طور خلاصه، SDK در برگیرندهٔ API های لازم و ضروری برای توسعه اندروید می باشد که توسط تیم برنامه نویسی گوگل طراحی شده اند. در واقع پس از دریافت SDK و نصب آن روی محیط برنامه نویسی اکلیپس، علاوه بر استفاده از API های جاوا که از طریق JDK در اختیار اکلیپس قرار می گیرند، به API های اختصاصی اندروید که با Syntax یا ساختار زبان برنامه نویسی جاوا نوشته شده اند نیز دسترسی پیدا نموده و از این طریق خواهیم توانست توسعه اندروید را آغاز نماییم. اما پیش از اینکار نیاز داریم تا ADT اندروید را نیز دریافت نموده و به محیط برنامه نویسی اکلیپس معرفی نماییم.
در حقیقت نصب پلاگین ADT این امکان را به ما می دهد تا بتوانیم از طریق محیط برنامه نویسی اکلیپس و استفاده از API های موجود در SDK که برای توسعه اندروید تعبیه شده اند دست به طراحی اپلیکیشن اندروید بزنیم. از سوی دیگر این پلاگین دارای یک ابزار طراحی GUI[۴] است که با استفاده از آن خواهیم توانست بدون کدنویسی و صرفا با Drag and Drop به طور مثال به ایجاد یک دکمه بپردازیم.

---

[۱] Java Runtime Environment

[۲] Java Development Kit

[۳] http://www.oracle.com/technetwork/java/javase/downloads/index.html

[۴] Graphical User Interface به معنی رابط گرافیکی کاربر

۲ ۲ ۳ دانلود ADT Bundle برای توسعه اندروید

علاوه بر محیط برنامه نویسی اکلیپس، شرکت گوگل نسخه ای از نرم افزار اکلیپس را تحت عنوان ADT Bundle تولید کرده است که کلیهٔ ابزارهای لازم برای توسعه اندروید از جمله خود محیط برنامه نویسی اکلیپس به علاوهٔ SDK و ADT را دارا است. این محیط برنامه نویسی به هیچ وجه نیاز به نتظیمات خاصی نداشته و صرفاً با اجرای برنامه خواهیم توانست توسعه اندروید را شروع نماییم.

برای دانلود این نرم افزار می بایست به سایت توسعه شرکت گوگل مراجعه نموده و این نرم افزار را به صورت کاملاً رایگان دریافت نمود ولی از آنجا که امکان دسترسی کاربران ایرانی به این وب سایت امکان پذیر نمی باشد، اکثر کاربران نخواهند توانست به سادگی ADT Bundle را دانلود کنند.

۲-۳ آشنایی با ساختار و اجزای تشکیل دهنده سیستم عامل اندروید

۲ ۳ ۱ اجزای تشکیل دهندهٔ سیستم عامل اندروید

سیستم عامل اندروید بر پایه سیستم عامل لینوکس طراحی شده است و ماشینی مجازی در این سیستم عامل تحت عنوان Dalvik تعبیه شده است که برای دستگاه های موبایل بهینه سازی شده است. به طور خلاصه می توان گفت که این ماشین مجازی دارای وظایف متعددی از جمله مدیریت حافظه، سهولت در استفاده از Sandbox در تولید اپلیکیشن، فشرده سازی بیشتر اپلیکیشن و سازگاری با CPU های مختلف در انواع مختلف دستگاه ها بدون نیاز به بازنویسی اپلیکیشن برای هر CPU خاص می باشد.

برای جستجو در اینترنت، سیستم عامل اندروید از موتور جستجوی Open Source یی تحت عنوان WebKit استفاده می کند. به منظور پردازش عکس های دو بعدی و سه بعدی، سیستم عامل اندروید از OpenGL ES استفاده می کند و برای ذخیره سازی اطلاعات از SQLite استفاده می کند.

همانطور که در آموزش اول اشاره شده، زبان برنامه نویسی اصلی سیستم عامل اندروید جاوا است. در واقع پس از تکمیل یک اپلیکیشن اندروید، این اپلیکیشن در قالب یک فایل با پسوند apk که مخفف Android Package است Compile می شود که این فایل را می توان در بازارهای مختلفی همچون Google Play و غیره در اختیار علاقمندان قرار داد تا بتوانند با نصب آن از اپلیکیشن ها استفاده نمایند.

به طور خلاصه یک اپلیکیشن اندروید از Component های مختلفی که در ادامه این آموزش به چهار مورد از اصلی ترین ها اشاره می شود، یک فایل Manifest و Resource های متفاوتی تشکیل می شود. در ادامه هر یک از این اجزا را مورد بررسی قرار خواهیم داد.

برای ساخت یک اپلیکیشن اندروید چهار Component اصلی وجود دارد که عبارتند از: Activity و Service و Content Provider و Broadcast Receive.

۲ ۳ ۲ معرفی Activity ها در اندروید

در صورتیکه بخواهیم معنای یک Activity را در سیستم عامل اندروید به خوبی متوجه شویم، می توان آن را به منزله یک صفحه از اپلیکیشن تصور کرد. برای روشن شدن این مسئله مثالی ذکر می کنیم. فرض کنیم که یک اپلیکیشن را اجرا می کنیم. این اپلیکیشن پس از اجرا وارد صفحه اصلی برنامه می شود. این صفحه اصلی یک Activity است. حال می بینیم که چندین دکمه در این صفحه اصلی برای رفتن به بخش های مختلف برنامه تعبیه شده به طور مثال یک دکمه جهت صفحه راهنما، یک دکمه جهت ارتباط با طراح اپلیکیشن، یک دکمه جهت وارد شدن به موتور جستجو در اینترنت و غیره. زمانیکه ما پس از زدن دکمه راهنما وارد صفحه راهنما می شویم در واقع وارد یک Activity دیگر شده ایم. اکنون این Activity جدید Activity قبلی که مربوط به صفحه اصلی برنامه بود را پس زده و جای آن را می گیرد. در این حین اگر دکمه Back تلفن همراه خود را فشار دهیم، Activity قبلی مجدد بالا آمده و Activity مربوط به صفحه راهنما را پس می زند.

اکنون ببینیم که یک Activity را به چه نحوی می توان ساخت. در حقیقت دو راه برای ساخت Activity ها در سیستم عامل اندروید وجود دارد: راه اول که به صورت دینامیک و پویا است با استفاده از برنامه نویسی جاوا می باشد و راه دوم با استفاده از XML است. روشی که در این مجموعه بیشتر مورد استفاده قرار خواهد گرفت، طراحی Activity ها با استفاده از XML است چرا که نه تنها این کار آسان تر است بلکه با پیروی از چنین رویکردی می توان کد های مربوط به GUI یا محیط گرافیکی را از کد های مربوط به نحوه عملکرد عناصر داخل این محیط گرافیکی که در زبان جاوا نوشته می شوند را از یکدیگر مجزا ساخت و این در حالی است که با اتخاذ چنین رویکردی Debug کردن اپلیکیشن ما هم به مراتب آسان تر خواهد شد.

۲ ۳ ۳ معرفی Service ها در اندروید

دومین Component که از میان دیگر Component ها از اهمیت بسزایی برخوردار است Service است. در حقیقت در پروسه طراحی اپلیکیشن اندروید Service به اجرای عملیاتی اطلاق می شود که خارج از دید کاربر رخ می دهند که ممکن است این دسته از عملیات خواه برای مدت زمان کوتاهی صورت پذیرند و خواه برای مدت زمان طولانی تری اجرا گردند. نکته ای که در مورد Service ها جالب است این است که این دسته از Component ها دارای هیچ گونه GUI یی نمی باشند و در Background برنامه اجرا می شوند. برای روشن شدن مطلب مثالی

ذکر می کنیم. فرض کنیم که در حین اس ام اس دادن به یکی از دوســـتان خود تمایل داریم تا به آهنگ مورد علاقه مان نیز گوش فرا دهیم. در حقیقت زمانیکه موســیقی در حال پخش شــدن اســت و ما اقدام به اس ام اس دادن می کنیم ما هیچ اثری به جزء صـدای آهنگ نخواهیم دید و این همان خاصـیت Service ها است که گفته می شود در پس زمینه اپلیکیشن یا Background اتفاق می افتند.

نکته دیگری که در مورد Service ها می بایســت مد نظر قرار دهیم این اســت که Service ها مجزا از Activity هـایی کـه آن هـا را اجرا می کنند می توانند به کار خود ادامه دهند. فرض کنیم که یک Activity داریم که دارای یک دکمه Play اسـت. با زدن این دکمه پخش موسیقی آغاز خواهد شد و این در حالی است که اگر این Activity را ببندیم و یا یک Activity دیگر جای این Activity را بگیرد، Service یی که وظیفه داشـت پخش موسیقی را اجرا کند می تواند به کار خود ادامه دهد.

۲ ۳ ۴ معرفی Content Provider ها در اندروید

جایگاه ســوم در میان Component ها را Content Provider ها به خود اختصـــاص داده اند. وظیفه ای که بر عهده این گروه از Component ها گذاشـته شده است، ذخیره سازی اطلاعات و قرار دادن اطلاعات ذخیره شده در اختیار دیگر اپلیکیشن ها می باشد. به طور خلاصه تنها راه به اشتراک گذاری داده ها میان اپلیکیشن های مختلف به کارگیری Content Provider ها می باشـــد. چنانچه ما داده هایی داشـــته باشـــیم و بخواهیم این داده ها را در دسـترس دیگر بخش های اپلیکیشـن خود قرار دهیم، به سادگی می توانیم یک Content Provider ایجاد کرده و مابین بخش هایی که می خواهند از آن اسـتفاده کنند ارتباطی برقرار سـازیم. تعدادی از این Content Provider ها از پیش در سـیسـتم عامل اندروید تعبیه شـده اند که از آن جمله می توان به Contact و Media اشاره کرد. به طور مثال اگر بخواهیم در اپلیکیشـنی به لیسـت شماره های تلفن همراه خود دسترسی پیدا کنیم، صرفاً نیاز است تا ارتباطی مابین اپلیکیشن خود و Content Provider مرتبط با Contact سیستم عامل اندروید برقرار سازیم.

۲ ۳ ۵ معرفی Broadcast Receiver ها در اندروید

آخرین Component تحت عنوان Broadcast Receiver این وظیفه را دارا اسـت تا به هشدارهایی که در سطح کل سـیسـتم بوجود می آیند پاسـخ دهد. همانند Service ها Broadcast Receiver ها نیز دارای هیچ گونه GUIیی نمی باشـند. برای روشـن شـدن وظیفه Broadcast Receiver ها به مثالی اکتفا می کنیم. فرض کنیم می خواهیم برنامه ای بنویسـیم که زمانیکه انرژی باطری تلفن همراه ما به ۳۰ درصـد رسـید به ما هشـدار دهد. این

هشـــداری که به محض رســـیدن انرژی تلفن همراه به ۳۰ درصـــد روی صـــفحه تلفن مشـــاهده می شـــود یک نوع Broadcast Receiverمی باشد.

۲ ۳ ۶ آشنایی با فایل Manifest اندروید

پس از آشـنایی با چهار Component اصـلی سـیستم عامل اندروید، اکنون به بررسی یکی از مهم ترین فایل ها در سـاخت یک اپلیکیشـن اندرویدی که همان فایلManifest است می پردازیم. هر اپلیکیشن و یا بازی اندرویدی می بـایســـت دارای فـایلی تحـت عنوان AndroidManifest.xml در دایرکتوری اصـــلی خود کـه همـان Root Directory اسـت باشـد. کاری که این فایل انجام می دهد این اسـت که اطلاعاتی کلی پیرامون اپلیکیشـن یا بازی طراحی شـــده به ســـیســـتم اندروید می دهد. به عبارت دیگر، این فایل Component هایی را که ما در برنامه خود مورد اسـتفاده قرار داده ایم را به سـیسـتم عامل اندروید معرفی می کند. از سوی دیگر این فایل Permission ها یا مجوزهایی که کاربر برای نصـب برنامه می بایسـت صادر کند را نیز در بر می گیرد. فرض کنیم که برنامه ای طراحی کرده ایم که نیاز به دســترســی به اینترنت دارد. حال اگر کســی بخواهد این برنامه طراحی شــده توســط ما را مورد استفاده قرار دهد، در حین نصب اجازه دسترسی به اینترنت از آن کاربر توسط اپلیکیشن گرفته خواهد شد.

ســیســـتم عامل اندروید دارای نســخه های مختلفی اســت. چنانچه ما برنامه ای طراحی کنیم که برای مثلا نســخه Gingerbread یا نسـخه ۳,۰ طراحی شده باشد، این مسئله در فایل Manifest ذکر خواهد شد و چنانچه کاربری که سـیسـتم عامل اندروید وی پایین تر از نسـخه ۳,۰ باشـد بخواهد برنامه طراحی شـده ما را استفاده کند در حین نصب فایل Manifest این مسئله را خواهد فهمید و از نصب برنامه جلوگیری به عمل خواهد آورد.

۲-۴ ساخت اولین پروژه در اکلیپس

۲ ۴ ۱ تعیین مسیر Workspace

پس از نصـــب موفقیت آمیز محیط برنامه نویســـی اکلیپس و ابزارهای مورد نیاز برای توســـعه اندروید که عبارتند از JDK و ADT و SDK و انجام تنظیمات اولیه توسـعه اندروید در محیط اکلیپس اکنون می توانیم اقدام به ساخت اولین پروژه اپلیکیشن اندرویدی خود در اکلیپس نماییم.

محیط برنامه نویسـی اکلیپس نیازی به نصـب روی سـیسـتم نداشته و صرفاً با کلیک کردن روی آیکون این نرم افزار خواهیم توانســـت آن را اجرا کنیم. پس از اجرای نرم افزار اکلیپس، این نرم افزار از کاربر می خواهد تا مســـیری را به

عنوان مسـیر workspace روی هارد مشـخص کند که از این پس این مسـیر به عنوان مسیری خواهد بود که کلیه پروژه ها در آن ذخیره خواهند شد.

۲ ۴ ۲ مراحل ساخت یک پروژه جدید

در حقیقت برای سـاخت یک پروژه جدید اپلیکیشـن اندروید در محیط برنامه نویسی اکلیپس راه های متفاوتی وجود دارد. آسان ترین راه برای کلیک کردن روی گزینه ای است که در تصویر زیر نشان داده شده است:

شکل (۲ ۱) ساخت پروژه جدید

با قرار دادن نشـــانگر موس روی این گزینه عبارت Opens a wizard to help create a new Android projectنشـان داده خواهد شـد(در تصـویر فوق با خط قرمز نشان داده شده است). ترجمه تحت الفظی این عبارت "بازکردن پنجرۀ سـاخت یک پروژه جدید اندروید" می باشـد. پس از کلیک کردن روی این گزینه پنجره ای به شکل زیر مشاهده خواهد شد:

شـکل (۲ ۲) ساخت پروژه جدید

در این پنجره در بخش Project Name می بایســـت نامی برای پروژه خود انتخاب نماییم. در حقیقت این نام به منزله نام اپلیکیشــن ما نخواهد بود بلکه نامی اســت که از آن طریق پروژه ما در Workspace ذخیره خواهد شــد. اکنون نـام My First Android Project به معنی "اولین پروژه اندروید من" را مقابل Project Name می نویسـیم. سپس می بایست یکی از سه گزینه موجود را انتخاب نماییم و از آنجا که قصد داریم یک پروژه جدید ایجاد کنیم، گزینه Create new project in workspace به معنی "ســـاخت پروژه جدید در workspace ایجاد شــده" را انتخاب می کنیم. ســـپس با تیک دار کردن گزینه Use default location به معنی "اســـتفاده از محل ذخیره ســازی پیش فرض"، این دســتور را به اکلیپس می دهیم که همان مســیری که در ابتدای نصب اکلیپس برای Workspace انتخاب کردیم را برای ذخیره ســازی این پروژه نیز استفاده کند. در صورتیکه بخواهیم پروژه خود را در مســیر دیگری ذخیره ســازیم، می توانیم تیک این گزینه را برداشــته و مســیر جدید را وارد کنیم. حال می توانیم روی دکمه Next کلیک کنیم:

شکل (۲ ۳) ساخت پروژه‌ی جدید

همانطور که در تصویر فوق مشاهده می شود در این پنجره می بایست نوع Build Target را مشخص کنیم. به طور خلاصــه منظور از Build Target همان نســخه ای از ســیســتم عامل اندروید است که قصد داریم اپلیکیشن خود را اختصاصاً برای آن نسخه طراحی کنیم. در این پروژه نسخه ۲,۱ را مد نظر قرار می دهیم. به عبارت دیگر کاربرانی که

روی تلفن همراه ایشان نسخه ۲,۱ اندروید یا نسخه های پایین تر نصب باشد خواهند توانست از اپلیکیشن ما استفاده کنند. همانطور که قبلاً توضیح داده شد، هرچه از API های بالاتر مثلاً ۴,۱ استفاده کنیم به امکانات بیشتری در توسعه اندروید دسترسی خواهیم داشت اما این در حالی است که تعداد کاربران کمتری خواهند توانست از اپلیکیشن ما استفاده کنند چرا که مدت زمانی به طول خواهد انجامید تا تلفن های همراه موجود در دست کاربران با آخرین نسخه های سیستم عامل اندروید به روز رسانی شود.

از سوی دیگر لیست API های موجود در این پنجره بسته به تعداد SDK هایی که در مرحله آماده سازی محیط برنامه نویسی اکلیپس دانلود می کنیم می تواند متفاوت باشد. به طور مثال در این نسخه از SDK که دانلود نموده ایم از نسخه های ۱,۱ و ۱,۶ و ۳,۲ به طور مثال خبری نیست. علت این مسئله هم به نظر می رسد این باشد که مدیر وب سایتی که ما SDK را در آموزش پیشین از آن دانلود کرده ایم ممکن است بر این باور باشند که تفاوت بسیار چندانی مابین نسخه های نزدیک به هم وجود ندارد(مثلاً مابین نسخه ۲ و ۲,۱) و این در حالی است که باور ایشان کاملاً درست است.

حال می توانیم روی دکمه Next کلیک کنیم:

شکل (۲ ۴) ساخت پروژه‌ی جدید

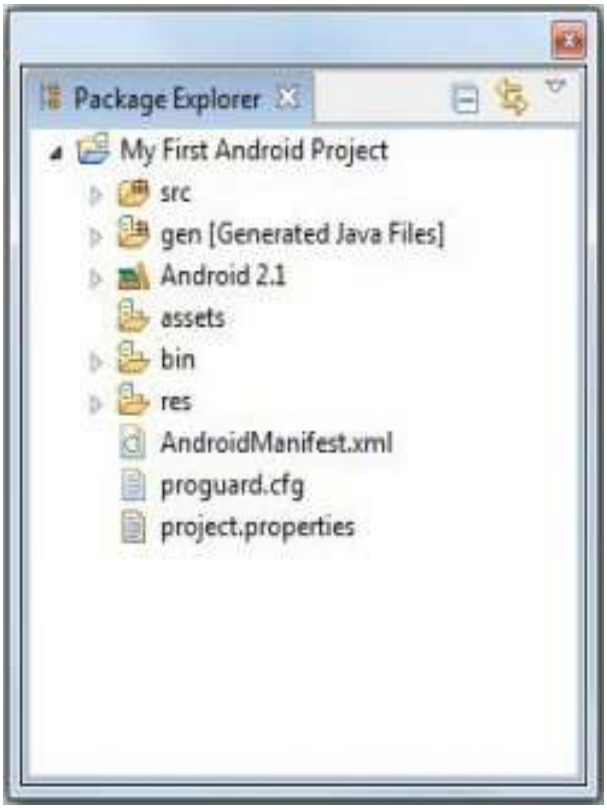
در این پنجره می بایست اطلاعات کلی اپلیکیشن خود را وارد نماییم. همانطور که در تصویر فوق مشاهده می شود بر اســـاس نـامی که در مرحله پیش برای پروژه خود انتخاب نمودیم، جاهای خالی این پنجره پر شـــده اند. در بخش Application Nameبه معنی "نام اپلیکیشـن" می بایسـت نامی که می خواهیم اپلیکیشن ما داشته باشد را وارد کنیم. به عبارت دیگر این همان نامی اسـت که کاربران پس از نصب اپلیکیشن ما روی تلفن همراه خود خواهند دید. نام وارد شـــده را تغییر نداده به مرحله بعد می رویم. در بخش Package Name به معنی "نام پیکج" می بایســت نامی برای پکیج اپلیکیشـن خود انتخاب کنیم. در واقع هر اپلیکیشن حداقل از یک پکیج تشکیل شده است و کار این پکیج قرار دادن فـایـل هـای مرتبط بـه یکـدیگر در اپلیکیشـــن مـا در کنـار یکدیگر اســـت. در این پروژه بنده نام com.alirezaamiri.mainpackage را انتخـاب می کنم. در واقع این نام بســـته به شـــخص یا شـــرکتی که اپلیکیشن را طراحی می کند متفاوت است.

همانطور که در بخش‌های پیشـین توضـیح داده شـد، هر اپلیکیشـن اندروید حداقل از یک Activity تشـکیل شده اسـت. در واقع با تیک زدن گزینه Create Activity به معنی "سـاخت Activity جدید" می توانیم Activity اصـلی که اپلیکیشن ما با آن شروع خواهد شد را در این مرحله بسازیم. به طول معمول این بخش هم توسط اکلیپس به طور پیش فرض برای ما پر می شـــود اما این در حالی اســـت که این نام نه تنها طولانی اســـت بلکه گویا هم نمی باشد، برای این منظور نام موجود را به نام MainActivity به معنی "یک Activity اصلی" تغییر می دهیم.

در بخش Minimum SDK به معنی "حداقل SDK مورد نیاز" عدد ۷ به طور پیش فرض توســـط خود اکلیپس وارد شده است. این بدان معنا است که اپلیکیشن ما برای آنکه اجرا شود حداقل به API سطح ۷ نیاز دارد. به عبارت دیگر این اپلیکیشن روی تلفن همراهی اجرا خواهد شد که نسخه اندروید آن Éclair باشد. در واقع این عدد با توجه به نوع نسخه اندروید که در مرحله پیش انتخاب کردیم انتخاب شده است. به عبارت دیگر از آنجا که ما نسخه ۲,۱ را انتخاب کردیم و API اندروید به کار گرفته شـده در این نسخه API 7 است، از این رو عدد ۷ وارد شده است. نکته ای که در اینجا می بایسـت حتما مد نظر داشـته باشـیم این اسـت که API وارد شده در Minimum SDK نمی بایسـت از API متناظر با نسـخه تعیین شده در مرحله پیش بزرگ تر باشد. به طور مثال از آنجا که در مرحله پیش نسخه ۲,۱ انتخاب شده است از این رو برای Minimum SDK نمی توانیم مثلاً API 15 را انتخاب کنیم.

اکنون مراحل ساخت اولین پروژه اندروید ما به پایان رسیده و می توانیم دکمه Finish را بزنیم.

پس از زدن دکمه Finish به محیط اکلیپس بازگشـته و پروژه ای تحت عنوان My First Android Project در بخش Package Explorer اکلیپس مشاهده خواهد شد:



شکل (۲ ۵) ساخت پروژه جدید

۲-۵ پروژه ساده یک

همان طور که توضیح داده شد، اپلیکیشن‌های اندروید از تعدادی Activity درست شده‌اند. حال در این برنامه سعی شـده اسـت در یک Activity سـاعت سیستم را به کاربر نمایش دهیم. توسعه یک Activity به دو بخش تقسیم می‌شود:

۱.UI یا طراحی گرافیکی

۲. کلاس جاوا مربوطه

بعد سـاختن یا new کردن یک Activity دو فایل در پروژه شـما ساخته می‌شود. نخست برای طراحی گرافیک به سراغ فایل XML مربوطه می‌رویم.

برای نمایش سـاعت نیاز به یک TEXTVIEW داریم تا رشـته را روی آن نمایش دهیم به همین خاطر در فایل XML یک TEXTVIEW اضافه می‌کنیم. کد نهایی به شکل زیر خواهد بود:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:tools="http://schemas.android.com/tools"
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   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">

<TextView
      android:id="@+id/textView1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_alignParentLeft="true"
      android:layout_alignParentTop="true"
      android:layout_marginLeft="89dp"
      android:layout_marginTop="123dp"
      android:text="Large Text"
      android:textAppearance="?android:attr/textAppearanceLarge">
</RelativeLayout>
```

بعد از این بخش سراغ قسمت دوم می‌رویم. در این قسمت نیاز است تا فایل XML که طراحی کرده ایم را از طریف کلاس جاوا به Activity نسبت دهیم. برای این کار از دستور setContentView استفاده می‌کنیم.

بعد از این کار نیاز اســت تا TEXTVIEW را که در XML قرار دادیم را در کلاس جاوا پیدا کنیم تا اعمال لازم را روی آن اعمال کنیم. این کار با دستور findViewById قابل انجام است.

بعد از اینکار تنها کافی است ساعت سیستم را بخوانیم با دستور setText در TEXTVIEW لحاظ کنیم.

کد نهایی کلاس جاوا به شکل زیر خواهد بود:

```
public class MainActivity extends Activity {
@   Override
   protected void onCreate(Bundle savedInstanceState){
      super.onCreate(savedInstanceState);
     setContentView(R.layout.activity_main);
```

```
    TextView  text = (TextView ) findViewById(R.id.textView1);
      CharSequence timeAgo = DateUtils.getRelativeTimeSpanString(
                                    Long.parseLong("1404389921").
                                    System.currentTimeMillis(), DateUtils.SECOND_IN_MILLIS);
    text.setText(timeAgo);
  }
}
```

۲-۶ پروژه ساده دو

هدف از توسعه این برنامه مدیریت یک رویداد است. مهم ترین رویداد در اندروید لمس کردن یک VIEW است. در اینجا قصد داریم تا بعد از لمس شدن دکمه توسط کاربر، ساعت سیستم برای آن به نمایش درآید.

همانطور که متوجه شده‌اید UI یا طراحی گرافیک این ACTIVITY مشابه برنامه قبلی است با این تفاوت که یک دکه یا BUTTON به آن اضافه شده است. کد فایل XML به صورت زیر خواهد بود:

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  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">
<TextView
    android:id="@+id/textView1"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:layout_marginTop="78dp"
    android:text="Time Goes Here"..
    android:textAppearance="?android:attr/textAppearanceLarge"/>
```

```
  <Button
      android:id="@+id/button1"
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:layout_below="@+id/textView1"
      android:layout_centerHorizontal="true"
      android:layout_marginTop="32dp"
      android:text="Touch Here!"/>
</RelativeLayout>
```

کلاس جاوا نیز مشـــابه مثال قبل خواهد بود با این تفاوت که علاوه بر TextView باید Button را نیز پیدا کنیم تا اعمال لازم را روی آن انجام دهیم.

پیدا کردن View و اضـافه کردن متن به TextView در مثال قبل توضیح داده شد. اما برای مدیریت رویداد نیازبه یکEvent Listenerداریم. ایونت لیســنر برای رویداد لمس کردن با دستور setOnClickListener قابل اضافه کردن خواهد بود. کد نهایی کلاس جاوا به صورت زیر خواهد بود:

```
public class MainActivity extends Activity{
        @Override
        protected void onCreate(Bundle savedInstanceState){
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
                Button btn = (Button) findViewById(R.id.button1);
                btn.setOnClickListener(new View.OnClickListener(){
                        @Override
                        public void onClick(View v){
                                TextView txtv= (TextView) findViewById(R.id.textView1);
                                Date date = new Date();
                                SimpleDateFormat ft = new SimpleDateFormat ("hh:mm:ss");
                                String str = ft.format(date);
                                txtv.setText(str);
                        }
                });
        }
}
```

فصل ۳

برنامه نویسی سوکت

۳-۱ مقدمه

سـنگ بنای تمام برنامه‌های کاربردی لایه چهارم، مفهومی به نام سـوکت است که این مفهوم توسط طراحان سیستم عامل یونیکس به زیبایی به منظور برقراری ارتباط برنامه‌های تحت شبکه و تبادل جریان داده بین پروسه‌ها ابداع شد و در این فصل باید مفهوم را کالبد شکافی کنیم.

شـاید شـما این جمله معروف را شـنیده باشید که "در نیای یونیکس هر چیزی می تواند به صورت یک فایل تلقی و مدل شود"! تمام عوامل و انواع ورودی و خروجی ها (I/O) می توانند توسط سیستم فایل مدل شوند. مثلا چاپگر می تواند یک فایل باشد (مثلا فایلی با نام PRN). حال وقتی سیستم عامل، چاپگر را به صورت یک فایل استاندارد مدل کرده باشد شما با مفاهیمی که از فایلها و چگونگی بکارگیری آنها در محیط برنامه نویسی اموخته اید، برای راه‌اندازی چاپگر و چاپ یک متن، می توانید در برنامه خود عملیات ساده و در عین حال استاندارد را انجام دهید.

حال که ذهن شـما این نکته رو پذیرفت که هر نوع I/O در دنیای سـیسـتم عامل به صورت یک فایل استاندارد قابل عرضه و مدل کردن است، شما را با یک سوال کلیدی مواجه می کنیم:

" آیا ارتباط دو کامپیوتر روی شبکه ومبادله اطلاعات بین آن دو، ماهیت ورودی و خروجی ندارد؟ "

اگر ساختار فایل را برای ارتباط شبکه ای تعمیم بدهیم آنگاه برای برقراری ارتباط بین دو برنامه روی کامپیوتر ها راه دور روال زیر پذیرفتنی است:

الف) در برنامه خود از سـیستم عامل بخواهید تا شرایط را برای برقراری یک "ارتباط" با کامپیوتر خاص (با ادرس IP مشـخص) و برنامه ای خاص روی آن کامپیوتر (با آدرس پورت مشـخص) فراهم کند یا اصـطلاحا سوکتی را بگشاید. اگر این کار موفقیت آمیز بود، سـیسـتم عامل برای شـما یک اشاره گر بر می گرداند و در غیر اینصورت طبق معمول مقدار پوچ به برنامه شما باز خواهد گرداند.

ب) در صـورت موفقیت آمیز بودن مرحله قبل، به همان صـورتی که درون یک فایل می نویسـید یا از آن می خوانید، می توانید با توابع send() و recv() اقدام به مبادله داده ها کنید.

ج) عملیات مبادله داده ها که تمام شد، ارتباط را همانند یک فایل معمولی ببندید.(با تابع close())

برای آنکه در برنامه خود همانند فایل یک "اشـاره گر سـوکت" را از سیستم عامل طلب کنید تا برایتان مقدمات یک ارتباط را فراهم کند بایستی تابع سیستمی socket() را در برنامه خود صدا بزنید. در صورتی که عمل موفقیت آمیز بود، یک اشاره گر غیر پوچ بر می گردد که از آن برای فراخوانی توابع و روالهای بعدی استفاده خواهد شد.

دقیقـا همـانـنـد فـایلهـا که می توان همزمان چندین فایل را در یک برنامه واحد باز کرد و بر روی هر یک از آنها (با استفاده از اشاره گر فایل) نوشت یا از آنها خواند، در یک برنامه تحت شبکه نیز می توان بطور همزمان چندین ارتباط فعال و باز بین چندین پروسه داشت و با مشخصه هر یک از این ارتباط ها روی هر کدام از آنها مبادله داده انجام داد.

۳-۲ انواع سوکت و مفاهیم آنها

اگر بخواهیم از نظر اهمیت انواع سوکت را معرفی کنیم، دو نوع سوکت بیشتر وجود ندارد.(انواع دیگری هم هستند ولی کم اهمیت ترند). این دو نوع سوکت عبارتند از :

- سوکت های نوع استریم که سوکت های اتصال گرا (Connection Oriented) نامیده می شوند.
- سوکت های نوع دیتاگرام که سوکت های بدون اتصال (Connectionless) نامیده می شوند.

اگر عادت به پیش داوری دارید برای تمایز بین مفهوم این دو نوع سوکت، تفاوت بین مفاهیم ارتباط نوع TCP و UDP را مد نظر قرار بدهید! روش ارسال برای سوکت های نوع استریم همان روش TCP است و بنابراین داده ها با رعایت ترتیب، با اطمینان صد در صد و با نظارت کافی بر خطا های احتمالی مبادله می شوند. سوکت نوع دیتاگرام نامطمئن است و هیچگونه تضمینی در ترتیب جریان داده ها وجود ندارد.

اکثر خدمات و پروتکهای که در لایه چهارم تعریف شده اند و نیازمند حفظ اعتبار و صحت داده‌ها و همچنین رعایت ترتیب جریان ها بایتها هستند.

سوکت یک مفهوم انتزاعی از تعریف ارتباط در سطح برنامه نویسی است و برنامه نویس با تعریف سوکت عملا تمایل خود را برای مبادله داده ها به سیستم عامل اعلام کرده و بدون درگیر شدن با جزئیات پروتکل TCP یا UDP از سیتم عامل می خواهد تا فضا و منابع مورد نیاز را جهت برقراری یک ارتباط، ایجاد کند.

۳-۳ مفهوم سرویس دهنده / مشتری (Client/Server)

در برنامه نویسی سوکت، این نکته قابل توجه است که هر ارتباطی دو طرفه و بین دو پروسه کاربردی تعریف می شود لذا طرفین ارتباط بایستی در لحظه شروع، تمایل خود را برای مبادله داده ها به سیستم عامل اعلام کرده باشند. در هر ارتباط یکی از طرفین، " شروع کننده ارتباط" تلقی می شود.

عملیات در سمت سرور به شرح زیر است:

الف) یک سوکت را که مشخصه یک ارتباط یا یک نشست است به وجود بیاورید تا اینجا فقط به سیستم عامل اعلام کرده اید که نیازمند تعریف یک ارتباط هستید. در همین مرحله به سیستم عامل نوع ارتباط درخواستی خود را (TCP یا UDP) معرفی می نمایید این کار توسط تابع socket() انجام می شود.

ب) به سوکتی که باز کردید یک آدرس پورت نسبت بدهید. این کار توسط تابع سیستمی bind() انجام می شود

ج) در مرحله بعد به سیستم عامل اعلام می کنید که کارش را برای پذیرش تقاضاهای اتصال TCP شروع نماید. این کار توسط تابع سیستمی listen() انجام می شود و چون ممکن است تعداد تقاضاهای اتصال متعدد باشد باید

حداکثر ارتباط TCP را که می توانید پذیرای آن باشـید، تعیین کنید. چرا که سیستم عامل باید بداند برای پذیرش ارتباطات TCP چقدر فضا و منابع در نظر بگیرد.

د) نهایت با اسـتفاده از تابع ()accept از سـیستم عامل تقاضا کنید یکی از ارتباطات معلق را به برنامه شما معرفی و تسلیم کند

ه) از دستورات ()send و ()recv برای مبادله داده ها استفاده نمایید.

و) نهایتا ارتباط را خاتمه بدهید. این کار به دو روش امکان پذیر خواهد بود:

- قطع ارتباط دو طرفه ارسال و دریافت (تابع ()close)
- قطع یکطرفه یکی ز عملیات ارسال یا دریافت (تابع ()shutdown)

عملیات در سمت کلاینت به شرح زیر است:

الف) یک سـوکت را که مشخصه یک ارتباط است، به وجود بیاورید. تا اینجا فقط به سیستم عامل اعلام شده است که نیازمند تعریف یک ارتباط هستید.

ب) در مرحله بعد لازم نیسـت همانند برنامه سـرویس دهنده به سوکت خود آدرس پورت نسبت بدهید یعنی لزوم به اسـتفاده از ()bind وجود ندارد، زیرا برنامه سـمت مشـتری منتظر تقاضـای ارتباط از دیگران نیسـت بلکه خودش متقاضـی برقراری ارتباط با یک سـرویس دهنده دهنده اسـت. بنابراین در مرحله دوم به محض آنکه نیازمند برقراری ارتباط با یک سـرویس دهنده شـد آن تقاضا را با استفاده از تابع سیستمی ()connect به سمت آن سرویس دهنده می فرستد.

ج) از توابع ()send و ()recv برای ارسال داده ها اقدام می نماید.

د) ارتباط را با توابع ()close یا ()shutdown به صورت دوطرفه یا یکطرفه قطع می نمایید.

۳-۴ امکانات جاوا و C# برای برنامه نویسی سوکت

Java.net یک بسـته از بسـته های جاوا اسـت که کلاس هایی را برای کار با شـبکه، سوکت ها و URL ارائه کرده اسـت. در این بسته چندین نوع کلاس سوکت و دیتاگرام برای برنامه نویسی شبکه تعریف شده است. ابتدا از امکانات جاوا در خصـوص سـوکت های اسـتریم آغاز می کنیم. جاوا دو کلاس Socket و ServerSocket را برای برنامه نویسی با سوکت های استریم معرفی کرده است:

- Socket کلاسی جهت برقراری ارتباط و مبادله داده در سمت مشتری است.
- ServerSocket کلاسی جهت تعریف ارتباط و مبادله داده در سمت سرویس دهنده است

کلاس Socket : در کلاس Socket بیش از ۳۰ متد تعریف شـده که مهمترین آنها در اینجا و فصـل بعد توضـیح داده شده است :

Socket(String host,int port)
Socket(InetAddress address,int port)
Synchronized void close()
InputStrem getInputStream()
OutputStream getOutputStream()

Socket(String host,int port)

برای ایجادکلاس سـوکت از طریق این متد، مشـتقیما آدرس نام حوزهیک ماشـین و شـماره پورت سرویس دهنده مورد نظر در آرگومان های ان مشـخص می شود. اگر از این متد برای خلق یک سوکت استفاده شود، قیل از به وجود آمدن آن، نام حوزه به صـورت خودکار توسـط DNS ترجمه شـده و آدرس IP آن باز خواهد گشـت و در صـورت موفقیت آمیز این عمل، ادامه کار ممکن نخواهد بود.

مثال :

قطعه کد زیر تلاش می کند یک سوکت نوع استریم ایجاد نماید :

```
Try{
        Socket sock = new Socket("cs.wust.edu",25)

}

Catch (unknownHostExeption){

   System.out.printLn("can't find host.")

}

Catch( IoExeption e){

     System.out.printLn("Error connection to host.")

}
```

: Socket(Int Address,int port)

برای ایجاد ســوکت از طریق این متد، آدرس IP یک ماشــین و شــماره پورت پروســه ســرویس دهنده مورد نظر در آرگومانهای اول و دوم تحویل می شود.

```
Try}
        Socket sock = new Socket("192.168.1.2",25)
{
Catch (unknownHostExeption)}
   System.out.printLn("can't find host.")
{
Catch( IoExeption e)}
     System.out.printLn("Error connection to host.")
}
```

فصل ۴

پیاده‌سازی نرم‌افزار

۴-۱ کدهای اصلی

۴ ۱ ۱ اسکن کردن IP

```
 private boolean searchNetwork () {
    String range = "192.168.56;".
    for (int i = 1; i <= 254; i++)}
       String ip = range + i;
       try{
          socket = new Socket;()
          socket.connect(new InetSocketAddress(ip, 9000), 10);
          showOnline(i + "  online");
           return true;
       }
        catch (Exception e){
      }
   }
     return false;
}
```

۴ ۱ ۲ ایجاد یک ارتباط جدید

```
private void runChatServer() {
   try {
      ServerSocket serverSocket = new ServerSocket(9000);
      socket = serverSocket.accept();
      showOnline("" + socket.getInetAddress());
   }
   catch (IOException e) {}
}
```

۴ ۱ ۳ ارسال کاراکتر

```
btnSend.setOnClickListener(new OnClickListener() {
        @Override
        public void onClick(View v) {
          if (outputStream == null) {
            return;
          }
          try {
            String message = input.getText().toString() + "\n";
            myLog(message);
            outputStream.write(message.getBytes());
          }
          catch (IOException e) {
            e.printStackTrace();
          }

        }
      });
```

۴ ۱ ۴ ارسال فایل

```
 Toast.makeText(WifiChatActivity.this, "لطفا منتظر بمانید", Toast.LENGTH_SHORT).show();
   Thread thread = new Thread(new Runnable() {

      @Override
      public void run() {
        File myFile = new File(fileAddress);
        while (true) {
          byte[] mybytearray = new byte[(int) myFile.length()];
          try {
            BufferedInputStream bis = new BufferedInputStream(new FileInputStream(myFile));

            bis.read(mybytearray, 0, mybytearray.length);
```

```
            outputStream.write(mybytearray, 0, mybytearray.length);
          }
          catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
          }

        }
      }
    });
    thread.start();

    String message = " send File " + "\n";
    try {
      outputStream.write(message.getBytes());
    }
    catch (IOException e) {
      // TODO Auto-generated catch block
      log(e.toString());
      e.printStackTrace();
    }
    textSend.setText(textSend.getText() + time + "=> " + message);
```

۵ ۱ ۴ دریافت فایل

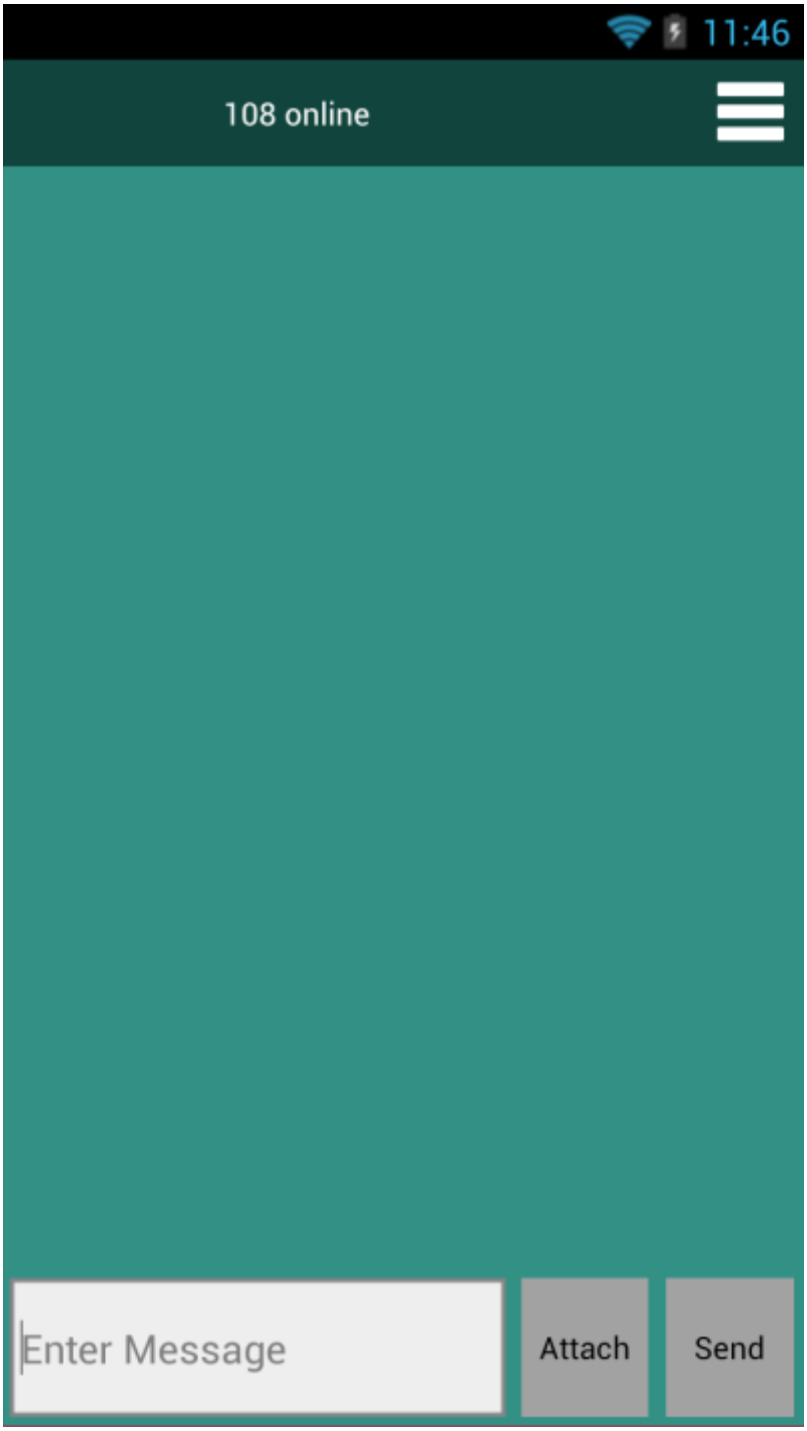
```
private void recive} ()
  byte[] mybytearray = new byte;[۱۰۲۴]
  InputStream is;
  try}
    is = socket.getInputStream;()
    FileOutputStream fos = new FileOutputStream(fileAddress);
    BufferedOutputStream bos = new BufferedOutputStream(fos);
    int bytesRead = is.read(mybytearray, 0, mybytearray.length);
```

```
            bos.write(mybytearray, 0, bytesRead);
            bos.close;()
}
        catch (IOException e)}
            e.printStackTrace;()
}

    }
```

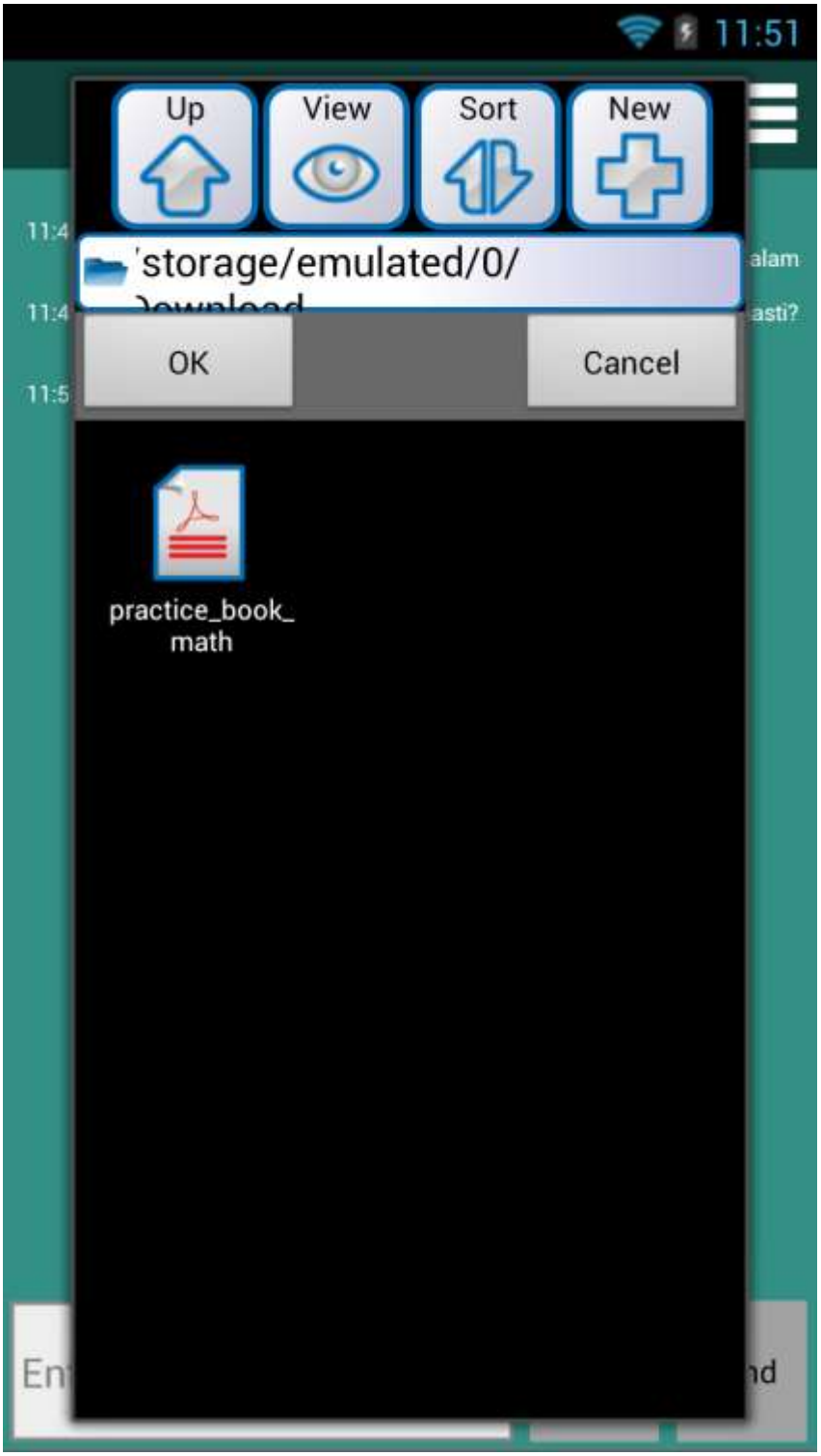
۶ ۱ ۴ بستن سوکت

```
      private void shutdown() {
    try {

       if (socket != null) {
          socket.close();
       }
    }
    catch (IOException e) {
       e.printStackTrace();
    }
    System.exit(0);
  }
```

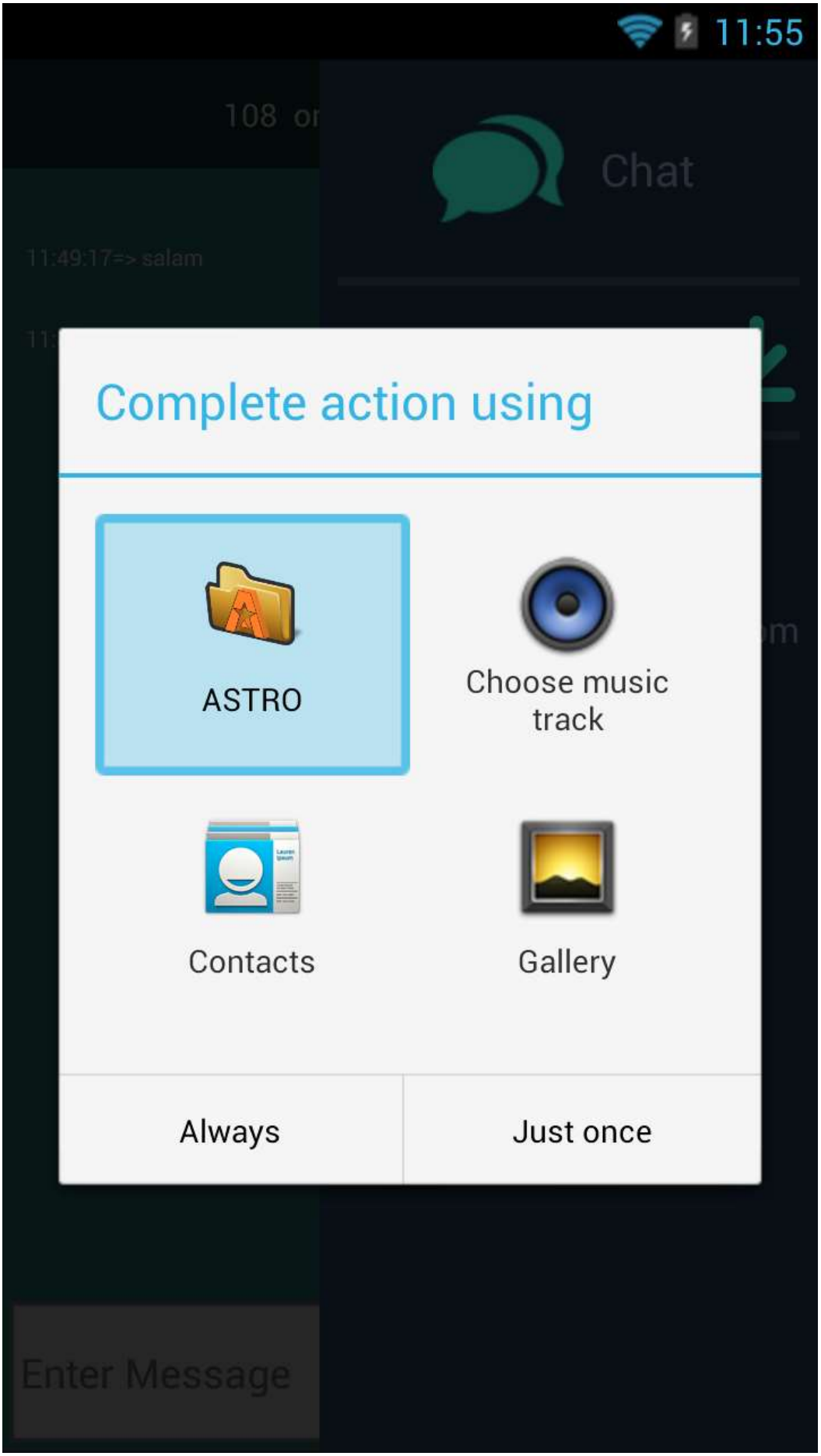
۲-۴ خروجی نرم‌افزار




11:50
108 online
11:49:9=> salam
11:49:17=> salam
11:49:37=> i am jaberALU
11:49:54=> khar hasti?
11:50:7=> are
khar hasti?
Attach
Send


11:51
Up
View
Sort
New
'storage/emulated/0/
OK
Cancel
practice_book_
math


11:55
108 or
11:49:17=> salam
11:49:54=> khar hasti?
Enter Message
Chat
Download
Developer
Jaber Babaki
jaber.babaki94@gmail.com


11:55
Chat
11:49:17=> salam
Complete action using
ASTRO
Choose music track
Contacts
Gallery
Always
Just once
Enter Message

فصل ۵

جمع‌بندی و پیشنهادها

۵-۱ خلاصه

برای تولید اپلیکیشـــن اندروید همانند توســعه نرم‌افزار در ســایر پلت فرم‌ها نیاز اســت تا بعد از فراهم کردن مقدمات توسعه، تکنیک‌ها و استراتژی‌های مهندسی نرم‌افزار در تمام طول فرایند تولید اپلیکیشن رعایت شود.

برنامه نوشــته شــده فوق به عنوان یک پروژه ســاده چت با اســتفاده از سوکت بوده که قابل توسعه می باشد، در حال حاضــر برنامه های چت به سرعت در حال گسترش می باشد و روز به روز به محبوبیت این این نرم افزار ها افزوده می شــود . ســوکت نویســی یکی از بحث های جذاب برنامه نویسی که در ارتباط با شبکه هست می باشد به این دلیل که همواره می توان از این تکنیک در موارد بسیاری استفاده نمود .

مراجع

مراجع

۱. نردبان؛ آموزش اندروید؛www.nardebaan.com.

2. Dan Gookin; Android Phones For Dummies, 2nd Edition, For Dummies, ۲۰۱۴.

3. Google; Android Developers; developer.android.com.

4. کتاب اصول مهندسی اینترنت ، دکتر احسان ملکیان