ပထမတွဲ ၁၁၆၁

ANDROID

APPLICATION ရေးသားပန်တီးခြင်းဆိုင်ရာ

အခြေခံသင်္ကေတများ

ဝင်ဟိန်

B.C.Tech (UCS.Y), D.B.S (Y.I.Eco),

Dip I.T (NCC, UK)

Q: ဒီစာအုပ်ဟာ ဘာအကြောင်းရေးထားတဲ့ စာအုပ်လဲ ?

A: Android လို့ခေါ်တဲ့ မောင်းနှင်စနစ် သုံးသော မိုဘိုင်းဖုန်းတွေအတွက် ပရိုဂရမ်တွေ၊ application တွေ ရေးနည်းကို ရှင်းပြထားတဲ့ စာအုပ်ပါ။ programming စာအုပ်ပေါ့။ Android အတွက် application တွေ ရေးသားနည်း အခြေခံသင်တန်းတစ်ခုမှာ သင်ရမယ့် သင်ခန်းစာတွေကို တဆင့်ပြီး တဆင့် လွယ်ရာမှ ခက်လာစေသည့် အစီအစဉ်နဲ့ စုစည်းထားတဲ့ စာအုပ်ပါ။

Q: ဒီစာအုပ်ဟာ ဘယ်သူတွေအတွက် ရည်ရွယ်သလဲ ?

A: အဓိက ကတော့ programming ကို စိတ်ပါဝင်စားတဲ့သူတွေ အတွက်ပါ။ ဒီနေ့ခေတ်မှာ programming ဆိုရင် ကွန်ပျူတာတွေ အတွက် ပရိုဂရမ်တွေပဲ ဖန်တီးရေးသားတဲ့ကိစ္စ မဟုတ်တော့ဘူးလေ။ အဓိက လူအားလုံးရဲ့ လက်ထဲမှာ ရှိလာနေတာက မိုဘိုင်းဖုန်းတွေ ဖြစ်တော့ မိုဘိုင်းဖုန်းတွေအတွက် ပရိုဂရမ်တွေ ရေးခြင်းဟာလည်း ခေတ်ရဲ့ လိုအပ်ချက် တစ်ရပ် ဖြစ်နေပါပြီ။ ဒါကြောင့် မိုဘိုင်းဖုန်းများအတွက် ပရိုဂရမ်ဖန်တီးရေးသားခြင်း အတတ်ပညာဟာလည်း သတင်းဆက်သွယ်ရေး နည်းပညာ Information Communication Technology (ICT) ရဲ့ ပင်မအကိုင်းအခက်ကြီး တစ်ခု ဖြစ်လာနေပါပြီ။ ဒီအတွက် လိုအပ်ချက်တစ်ရပ်ကို တတ်စွမ်းသလောက် ဖြည့်ဆည်းတာပါ။ ပရိုဂရမ်ရေးသားခြင်းနှင့် အသက်မွေးရန်ကြိုးစားနေသူတွေ၊ electronic engineering ပိုင်းဆိုင်ရာကို စိတ်ဝင်စား လေ့လာလိုက်စားနေသူတွေ၊ နည်းပညာသစ်ရဲ့ အားသာချက်ကို အသုံးချပြီး စွန့်ဦးတီထွင်လိုသူတွေ၊ အင်ဂျင်နီယာပညာရပ်နဲ့ ကွန်ပျူတာ ပညာရပ်ဆိုင်ရာ ကျောင်းသားသင်တန်းသားတွေအတွက် အဓိက ရည်ရွယ်ပါတယ်။

Q: ဒီစာအုပ်ကို ဖတ်ပြီး ဘာအကျိုးကျေးဇူးတွေ ရသွားမလဲ ?

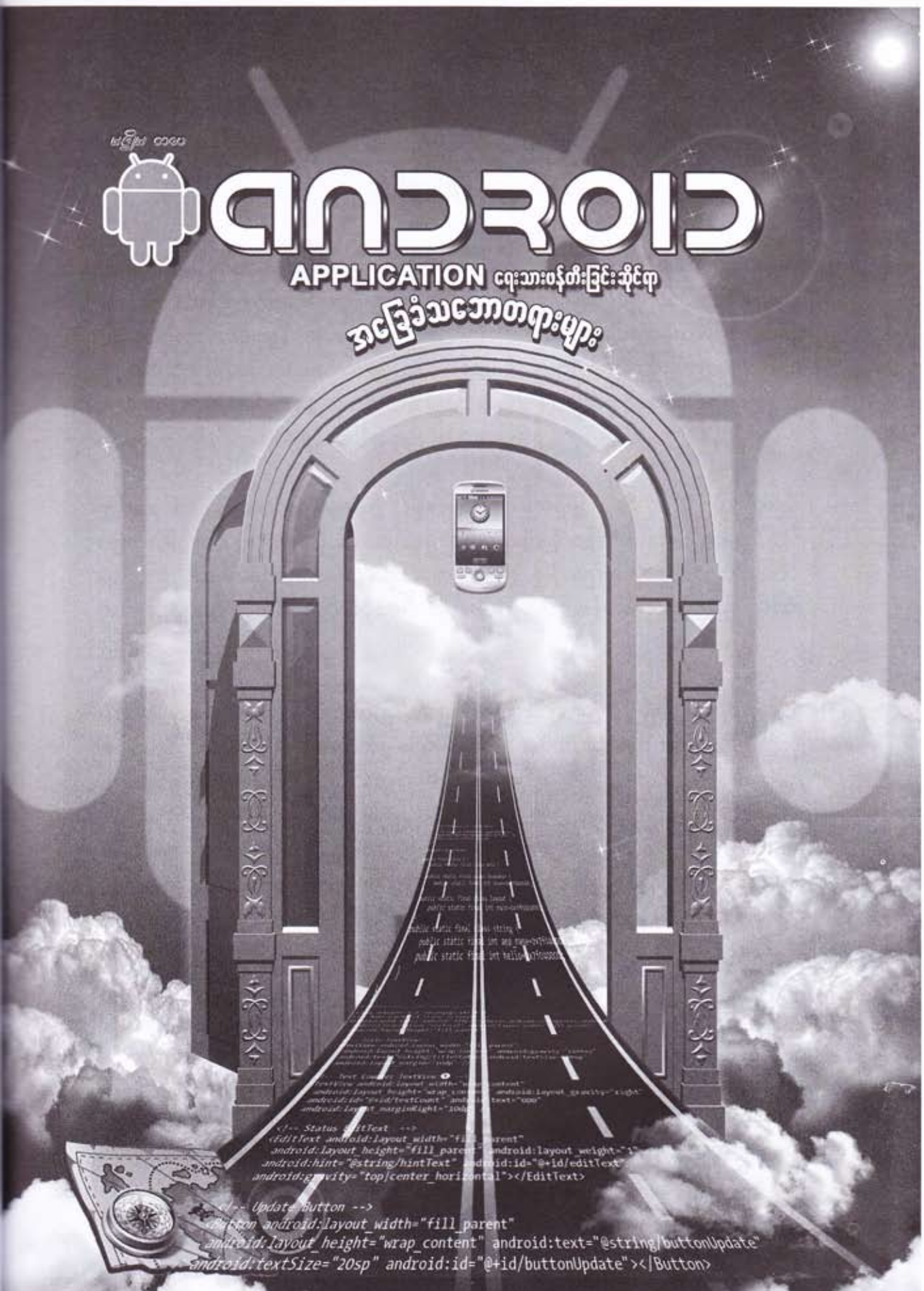
A: ဒီစာအုပ်ကို ပြုစုရခြင်းရဲ့ အဓိက ဦးတည်ချက် ဖြစ်သော Android application developing ဆိုင်ရာ အခြေခံသဘောတရားတွေကို ကိုယ်တိုင်ကိုယ်ကျ ထိုးထွင်းနားလည်သွားမယ်။ တဆက်တည်းမှာ Java နဲ့ C/C++ တို့ကို ကွန်ပျူတာမဟုတ်သော အခြား smart electronic device တွေ၌ ဘယ်လို ဆက်စပ်အသုံးချတယ်၊ Linux သည် Java တို့ C/C++ တို့နှင့် ဘယ်လို ယှဉ်တွဲအလုပ်လုပ်တယ် ဆိုတာတွေကိုပါ ချိတ်ဆက်လေ့လာသိရှိမယ်။ သာမန်ကာလျှံကာ ဖတ်လို့တော့ မရဘူးပေါ့။ သေသေချာချာလေး ဖတ်ပြီး လိုက်စဉ်းစားဖို့တော့ လိုပါတယ်။

မာတိကာ (၁)

မာတိကာ (၂)

မာတိကာ (၃)

မာတိကာ (၄)


ANDROID
APPLICATION ရေးသားပန်တီးခြင်းဆိုင်ရာ
အင်ဂြိုဟ်သင်္ဘောတရားများ
<!-- Status EditText -->
<EditText android:layout_width="fill_parent"
android:layout_height="fill_parent" android:layout_weight="1"
android:hint="@string/hintText" android:id="@+id/editText"
android:gravity="top|center_horizontal"></EditText>
<!-- Update Button -->
<Button android:layout_width="fill_parent"
android:layout_height="wrap_content" android:text="@string/buttonUpdate"
android:textSize="20sp" android:id="@+id/buttonUpdate"></Button>

ဤစာအုပ်.....

Android မောင်းနှင်စနစ်သုံးသည့် မိုဘိုင်းဖုန်းနှင့် အခြားသော ကြိုးမဲ့ကိရိယာများ အတွက် app များ ရေးသားလိုပါသလား။ ဤစာအုပ်သည် အခြေခံအားလုံးကို ပိုင်နိုင်သွားစေမည့် အကောင်းဆုံး လမ်းညွှန် ဖြစ်စေရပါမည်။ Android ၏ အခြေခံအစိတ်အပိုင်းများကို ဆက်စပ်သုံးစွဲ ပြီး user interface များ ဖန်တီးခြင်း၊ data များ သိမ်းဆည်း store လုပ်ခြင်း၊ network ကွန်ယက်များနှင့် ချိတ်ဆက်ခြင်းနှင့် အခြား များစွာသော သဘောတရားတို့ကို ကိုယ်တိုင် ဖန်တီးအသုံးချနိုင်စေပါမည်။

ဤစာအုပ်ပါ သဘောတရားများနှင့် နမူနာများကို ဖတ်ရှုရင်း ကိုယ်တိုင် တပါတည်း စမ်းသပ်ခြင်းဖြင့် twitter ကဲ့သို့သော application မျိုးကို ဖန်တီးနိုင်ပါမည်။ နောင်တွင် မည်သည့် Android app မျိုးဖြစ်မဆို အလွယ်တကူ ပြန်လည်အသုံးချနိုင်သော code-pattern များကို ကိုယ်တိုင် ကိုယ်ပိုင် toolbox တစ်ခုအဖြစ် ဖန်တီးနည်းကိုလည်း သိရှိပါမည်။

Android မောင်းနှင်စနစ် platform ကို ခြုံငုံနားလည်သွားစေပြီး မိုဘိုင်းကိရိယာတို့၏ အချင်းချင်း အပြန်အလှန် ချိတ်ဆက်မှု ဂေဟာစနစ်တွင် Android platform ၏ မည်ကဲ့သို့ အံဝင်ခွင်ကျရှိနေစေပုံကိုလည်း ထိုးထွင်းသိမြင်စေပါမည်။

Android platform ၏ ဖွဲ့စည်းထားပုံ၊ Android platform ကိုသုံးသော application တို့၏ ဖွဲ့စည်းတည်ဆောက်ပုံ၊ တစ်ခုကိုတစ်ခု ချိတ်ဆက်တည်မှီလျှက်ရှိသော application package (APK) တို့၏ ဖွဲ့စည်းတည်ဆောက်ပုံနှင့် ပျံ့နှံ့တည်ရှိပုံတို့ကိုလည်း နားလည်သွားစေပါမည်။

ကိုယ်ပိုင် Android environment တစ်ခုကို ကိုယ်တိုင်ဖန်တီးနိုင်စေရန် ရိုးစင်းလွယ်ကူသော program များဖြင့် အစပြုထားပါသည်။

ဝင်းပါန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Android ၏ အခြေခံ user interface များ တည်ဆောက်ပုံ၊ မြင်ကွင်း view များနှင့် အခင်းအကျင်း layout များတွင် user interface ၏ အခြေခံအစိတ်အပိုင်း element များအား နေရာချ စုစည်း organize လုပ်ပုံတို့ကိုလည်း လေ့လာသိရှိသွားစေပါမည်။

ကိုယ်ဖန်တီးသော application တွင် နောက်ကွယ်လုပ်ဆောင်ချက် background process service ကိုသုံးပြီး data များ update လုပ်သည့် ဝန်ဆောင်မှု service ကို တည်ဆောက်နိုင်ပါမည်။

'အခန်း (၁) – Android ကို ခြုံငုံသုံးသပ်ခြင်း' သည် Android နှင့် ပတ်သက်သမျှ နောက်ခံအကြောင်းအရာများကို မိတ်ဆက်တင်ပြထားခြင်း ဖြစ်ပါသည်။

'အခန်း (၂) – ဖွဲ့စည်းပုံ' သည် Android မောင်းနှင်စနစ် operating system နှင့် ၄င်း၏ အစိတ်အပိုင်း အားလုံး၏ အဆင့်မြင့် ခြုံငုံသုံးသပ်ချက် ဖြစ်ပါသည်။

'အခန်း (၃) – စတင်ခြင်း' သည် Android application များ ဖန်တီးမှုအတွက် လုပ်ငန်းခွင် environment ကို တည်ဆောက်ရန် လမ်းညွှန် ဖြစ်ပါသည်။

'အခန်း (၄) – ပင်မ အစိတ်အပိုင်းများ' တွင် application တစ်ခု ဖြစ်လာစေရန် application developer များ လက်ကိုင်ထား အသုံးပြုသော Android ၏ အစိတ်အပိုင်း component များအကြောင်းကို ရှင်းလင်းပြထားပါသည်။

'အခန်း (၅) – Project မိတ်ဆက်' သည် Android ၏ အသီးသီးသော feature များကို သိရှိသွားစေရန် နမူနာအဖြစ် အသုံးပြုရှင်းလင်းထားသော project တစ်ခု ဖြစ်ပါသည်။

'အခန်း (၆) – Android User Interface မိတ်ဆက်' သည် ကိုယ့် application အတွက် user interface တည်ဆောက်ပုံအကြောင်း ဖြစ်ပါသည်။

အခန်း – ၁
Android ကို နိုင်ငံသုံးသပ်ခြင်း

Android သည် ကြိုးမဲ့ဆက်သွယ်ရေးကိရိယာများအတွက် ဦးတည်ကာ ဒီဇိုင်းထုတ်ထားသော၊ အများ နားလည်လွယ်ပြီး မည်သူမဆို ယူသုံးနိုင်သည့် open source platform တစ်ခု ဖြစ်ပါသည်။ ၄င်းကို Google မှ ပြည့်စုံသည့်အဆင့်ရောက်အောင် မွမ်းမံပြီး Open Handset Alliance မဟာမိတ် အုပ်စုမှ ပိုင်ဆိုင်ပါသည်။ OHA မဟာမိတ်အဖွဲ့၏ ရည်ရွယ်ချက်မှာ ကြိုးမဲ့ကိရိယာများကို ပိုမိုဆန်းသစ်စေရန်နှင့် အသုံးပြုသူများအား ပိုမိုစုံလင်ပြီး ဈေးသက်သာသည့် ကြိုးမဲ့ ဆက်သွယ်ရေး စွမ်းဆောင်ရည်များ ဖန်တီးပေးရန် ဖြစ်သည်။ Android သည် ထိုရည်ရွယ်ချက်ကို ဖြည့်ဆည်းပေးသည့် platform တစ်ခုပင်။

hardware နှင့် ၄င်းပေါ် တွင် အလုပ်လုပ်နေသည့် software ကို သီးသန့်စီ ခွဲထုတ်ပေးနိုင်သည့် ပထမဆုံး open platform စစ်စစ်အဖြစ် ပေါ်ထွက်လာသော Android သည် ကြိုးမဲ့မိုဘိုင်း ဆက်သွယ်ရေး ကိရိယာလောကကို လုံးဝပြောင်းလဲဆန်းသစ်စေခဲ့သည်။

၁.၁ ပြည့်စုံခြင်း

Android သည် comprehensive platform တစ်ခုဖြစ်သည်။ ဆိုလိုသည်မှာ ၄င်းသည် မိုဘိုင်းကြိုးမဲ့ဆက်သွယ်ရေး ကိရိယာတစ်ခုအတွက် ပြည့်စုံသော software အစုအဝေး တစ်ခု ဖြစ်သည်။ application developer များအဖို့ ကြိုးမဲ့မိုဘိုင်းဆက်သွယ်ရေး ကိရိယာများအတွက် application များကို လျှင်မြန်လွယ်ကူစွာ ဖန်တီးနိုင်သည့် tool များနှင့် framework များအားလုံးကို ပေးထားသည်။ Android device တစ်ခုအတွက် app တစ်ခု စတင်စမ်းသပ်ဖန်တီးတော့မည် ဆိုလျှင် Android ဖုန်းတစ်လုံးပင် ရှိရန် မလိုသေးပါ။ Android Software Development Kit (SDK) ရှိလျှင် ဖြစ်ပါပြီ။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

၁.၂ မည်သူမဆို ယူသုံးနိုင်သော Open Source Platform ဖြစ်ခြင်း

Android သည် chip များအတွင်းသာ အလုပ်လုပ်သည့်၊ မြင်ကွင်းတွင် မပေါ်သော low-level Linux module များမှသည် မည်သည့် Android ဖုန်းခ်ဒ်မဆို အလုပ်လုပ်သော code အစုအဝေး native libraries များထိအောင်၊ application framework များမှသည် အသင့်သုံး complete application များထိအောင် အကုန် မည်သူမဆို ယူသုံးနိုင်သည့် open source ဖြစ်ပါသည်။

ထိုမျှမကသေး၊ Android ၏ မူပိုင်ခွင့်သည် မည်သူမဆို မည်သည့် ရည်ရွယ်ချက်အတွက်မဆို လွတ်လပ်စွာ ဆင့်ပွားအသုံးပြုနိုင်သော မူပိုင်ခွင့်မျိုး ဖြစ်သည်။ သို့ဖြစ်၍ developer တစ်ဦးအနေဖြင့် Android platform တစ်ခုလုံး၏ source code ကို ရရှိနိုင်သည်။ ထိုသို့ ရနိုင်သည့်အတွက် Android operating system မောင်းနှင်စနစ်ကြီး မည်ကဲ့သို့ အလုပ်လုပ်သည်ကို အူသိမ်အူမမကျန် သိရှိနိုင်သည်။ ထုတ်လုပ်သူ manufacturer တစ်ဦး ရှုထောင့်က ကြည့်လျှင် ကိုယ်ထုတ်လုပ်သော မည်သည့် hardware မျိုးအတွက်မဆို Android OS ကို ယူကာ သင့်သလိုပြုပြင်ထည့်သွင်းနိုင်သည်။

၁.၃ ကြိုးမဲ့မိုဘိုင်းကိရိယာများအတွက် ပုံစံထုတ်တည်ဆောက်ထားခြင်း

Android ကို ကြိုးမဲ့မိုဘိုင်းကိရိယာများအတွက် သက်သက် ရည်ရွယ်ကာ ပုံစံထုတ်တည်ဆောက်ထားခြင်း ဖြစ်သည်။ သို့ဖြစ်၍ကြိုးမဲ့မိုဘိုင်းကိရိယာများ၏ ကန့်သတ်ချက်များကို ထည့်သွင်းစဉ်းစားထားသည်။ ဥပမာ – ကြိုးမဲ့မိုဘိုင်းကိရိယာများသည် ဘက်ထရီမှ ပါဝါယူကြသည်။ ဘက်ထရီတို့၏ စွမ်းဆောင်ရည်သည် ချက်ချင်း လက်ငင်း ခုနှစ်ပျံကာ တိုးတက် ကောင်းမွန်လာကြဦးမည် မဟုတ်။ ဒုတိယ ဥပမာ အနေဖြင့် မိုဘိုင်းကြိုးမဲ့ကိရိယာများ၏ သေးငယ်မှုဖြစ်မည် ဖြစ်သော အရွယ်အစားကြောင့် မှတ်သားသိလှောင်မှု memory နှင့် လုပ်ဆောင်နှုန်း speed အရကြည့်လျှင် ၄င်းတို့သည် အမြဲပင် အကန့်အသတ် ရှိနေမည် ဖြစ်သည်။

Android သည် ကနဦး စတင်တည်ဆောက်စဉ်ကတည်းက တလျှောက်လုံး ထိုသို့သော ကန့်သတ်ချက်များကို ထည့်သွင်းစဉ်းစားထားခဲ့ပြီး တည်ဆောက်ခြင်း ဖြစ်သောကြောင့် အသုံးပြုသူများအတွက် ပိုမိုကောင်းမွန်မှုသော အတွေ့အကြုံကို ရရှိစေပါသည်။

ထို့ပြင် Android သည် ၄င်းကို ထည့်သွင်းသုံးစွဲမည့် device ၏ မြင်ကွင်း အရွယ်အစား screen size ၊ ရုပ်ထွက်ပြင်းအား resolution ၊ လျှပ်စစ်ပတ်လမ်းဖွဲ့စည်းပုံ အစုအဝေး chipset စသည်တို့အတိုင်း လိုက်လျောညီထွေ ပြုပြင်ယူနိုင်သော သဘောရှိသည်။ Android ၏ core သည် portable design ဖြစ်သည်။

၁.၄ Android ၏ သမိုင်းကြောင်း

– ၂၀၀၅ ခုနှစ်တွင် Google က Android Inc., လုပ်ငန်းကို ဝယ်ယူလိုက်သည်။ Google ၏ မိုဘိုင်းဖုန်းတစ်မျိုး မကြာမီ ထွက်လာတော့မည်ဟု ကမ္ဘာက ထင်ကြသည်။ သို့သော် အချိန်အတော်ကြာ မည်သို့မျှ ထူးခြားခြင်း မရှိ။

– ၂၀၀၇ တွင် Open Handset Alliance မဟာမိတ်အဖွဲ့ကို ထူထောင်ကြောင်း ကြေငြာသည်။ Android ကိုလည်း မည်သူမဆို ရယူသုံးစွဲနိုင်သောအရင်းအမြစ် open source အဖြစ် တရားဝင် ကြေငြာသည်။

– ၂၀၀၈ တွင် Android software development kit (SDK) 1.0 ကို ဖြန့်ချိခဲ့သည်။ HTC မှ ထုတ်လုပ်ပြီး အမေရိကရှိ T-mobile ကြိုးမဲ့ ဆက်သွယ်ရေးကွန်ယက်ဝန်ဆောင်မှု မှ ရောင်းချသော G1 မိုဘိုင်းဖုန်းတွင် မကြာမီ အသုံးပြုခဲ့သည်။

– ၂၀၀၉ တွင် Android ကိုအခြေခံသော ကိရိယာမျိုးစုံ ထွက်ပေါ်လာသည်။ Android Operating System ၏ မူကွဲသစ် new version များလည်း အသီးသီး ထွက်ပေါ်လာသည်။ ဥပမာ – Cupcake (1.5) ၊ Donut (1.6) နှင့် Eclair (2.0 နှင့် 2.1) ။

ဝင်းဟိန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

- ၂၀၁၀ တွင် Android သည် ဉာဏ်ရည်မြင့်ဖုန်း smart phone များတွင်သုံးသည့် မောင်းနှင်စနစ် platform များအနက် Blackberry ပြီးလျှင် ဒုတိယနေရာသို့ ရောက်ရှိလာခဲ့သည်။ Android ကို အသုံးပြုသော ကြိုးမဲ့မိုဘိုင်းဆက်သွယ်ရေး ကိရိယာ အရေအတွက် ၆၀ ကျော် ရှိလာခဲ့သည်။

၁.၅ Open Handset Alliance မဟာမိတ်

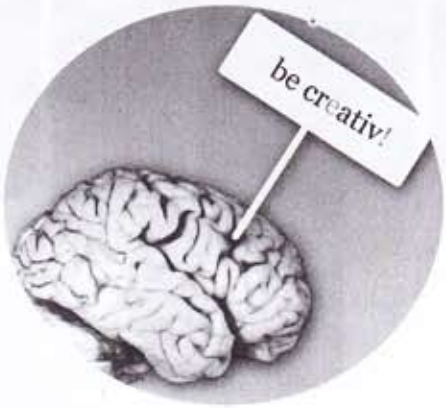
Android ကို Google တစ်ခုတည်းပိုင် ဆိုသည့်အနေအထားမ'က'ချေ။ Android ကို Open Handset Alliance မဟာမိတ်အဖွဲ့က ပိုင်ဆိုင်သည်။ Open Handset Alliance မဟာမိတ်အဖွဲ့ ဆိုသည်မှာ ပင်မ မိုဘိုင်းကွန်ယက်ဝန်ဆောင်မှုရောင်းချသူများ၊ မိုဘိုင်းကိရိယာ ထုတ်လုပ်သူများနှင့် အခြား ဆက်စပ်သူများ ပူးပေါင်း ထူထောင်ထားသော အမြတ်အစွန်းမဲ့ အဖွဲ့ ဖြစ်သည်။ OHA မဟာမိတ်အဖွဲ့သည် မိုဘိုင်းအသုံးပြုသူများ၏ အတွေ့အကြုံကို ပိုမိုကောင်းမွန်အောင် တီထွင်ဖန်တီးမှုနှင့် ပွင့်လင်းမှုကို ဦးတည်သည်။

၁.၆ Android ၏ မူကွဲ Version များ

အခြား မည်သည့် software မဆိုကဲ့သို့ပင် Android သည်လည်း တဖြည်းဖြည်း မွမ်းမံရင်း ကောင်းသည်ထက် ကောင်းမွန်ခဲ့ရခြင်းပင်။ ထိုသို့ မွမ်းမံရမှုများကို ၄င်း၏ မူကွဲအမှတ် version number များအရ သိနိုင်သည်။ တစ်ဖက်စာမျက်နှာပါ ဇယားတွင် Android ၏ မူကွဲအသီးသီးနှင့် ၄င်းတို့၏ အများသိ အမည်တို့ကို ယှဉ်တွဲ ဖော်ပြထားပါသည်။

Android version	API level	Nickname
Android 1.0	1	
Android 1.1	2	
Android 1.5	3	Cupcake
Android 1.6	4	Donut
Android 2.0	5	Eclair
Android 2.01	6	Eclair
Android 2.1	7	Eclair
Android 2.2	8	Froyo (frozen yogurt)
Android 2.3	9	Gingerbread
Android 2.3.3	10	Gingerbread
Android 3.0	11	Honeycomb

အရေးအပါဆုံးအချက်မှာ API level ဖြစ်သည်။ Application Developer တစ်ဦးအနေဖြင့် ကိုယ့် application သည် မည်သည့် ကိရိယာတွင် run မည် ဆိုသည်ကို API level အား အခြေခံပြီး ဆုံးဖြတ်ရသည်။ API level သည် ကိုယ့် application က မည်သည့် ကိရိယာများတွင် ဆိုလျှင် run မည်၊ မည်သည့် ကိရိယာများတွင် ဆိုလျှင်မူ run မည် မဟုတ် ဟူသော အချက်ကို သတ်မှတ်သည်။



ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Android application

ရေးသားစနစ်တိုးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

အခန်း – ၂
Android ဖွဲ့စည်းပုံ

Android application များကို ရေးမည့် developer တစ်ယောက်အနေဖြင့် Android system ၏ အခင်းအကျင်း layout များကို ကောင်းစွာ နားလည်ထားပါမှသာ Android platform ပေါ်တွင် အလွယ်တကူ လုပ်နိုင်သော အရာများနှင့် ဖန်တီးရန်မလွယ်သော အရာများ ကို ခွဲခြားနားလည်မည် ဖြစ်ပါသည်။

Android မောင်းနှင်စနစ် operating system သည် အလွှာအထပ်ထပ် ပါဝင်သော ကိတ်မုန့်တုံးကြီးနှင့် ပမာတူ၏။ အလွှာ တစ်လွှာချင်းစီတိုင်း၌ ကိုယ်ပိုင် ဝိသေသ လက္ခဏာများ နှင့် ရည်ရွယ်ချက်များ ရှိသည်။ အလွှာတစ်လွှာနှင့် တစ်လွှာသည် ရှင်းလင်းပြတ်သားစွာ ကွဲပြားနေခြင်းမျိုး မဟုတ်ဘဲ တစ်ခုကိုတစ်ခု စိမ့်ဝင်ရောယှက်နေခြင်းမျိုး ဖြစ်သည်။

Android သည် Linux ပေါ်တွင် အခြေခံထားခြင်း ဖြစ်သည်။ Linux ဆိုသည်မှာ အလွန်ကောင်းသော မောင်းနှင်စနစ် Operating System ကြီးတစ်ခု ဖြစ်ပြီး၊ မည်သူမဆိုရယူသုံးစွဲနိုင်သော ရင်းမြစ် Open Source ဟူသည့် စကားလုံး၏ 'ကိုယ်ပွား'ဟုပင် ဆိုနိုင်သည်။ Android ဖွဲ့စည်းပုံအဖြစ် Linux ကို အခြေခံရခြင်း၌ အကြောင်းအရင်းများစွာ ရှိသည်။ အဓိကမှာ ပေါ့ပါးခြင်း portability ၊ လုံခြုံခြင်း security နှင့် features စုံလင်မှုတို့ကြောင့် ဖြစ်သည်။

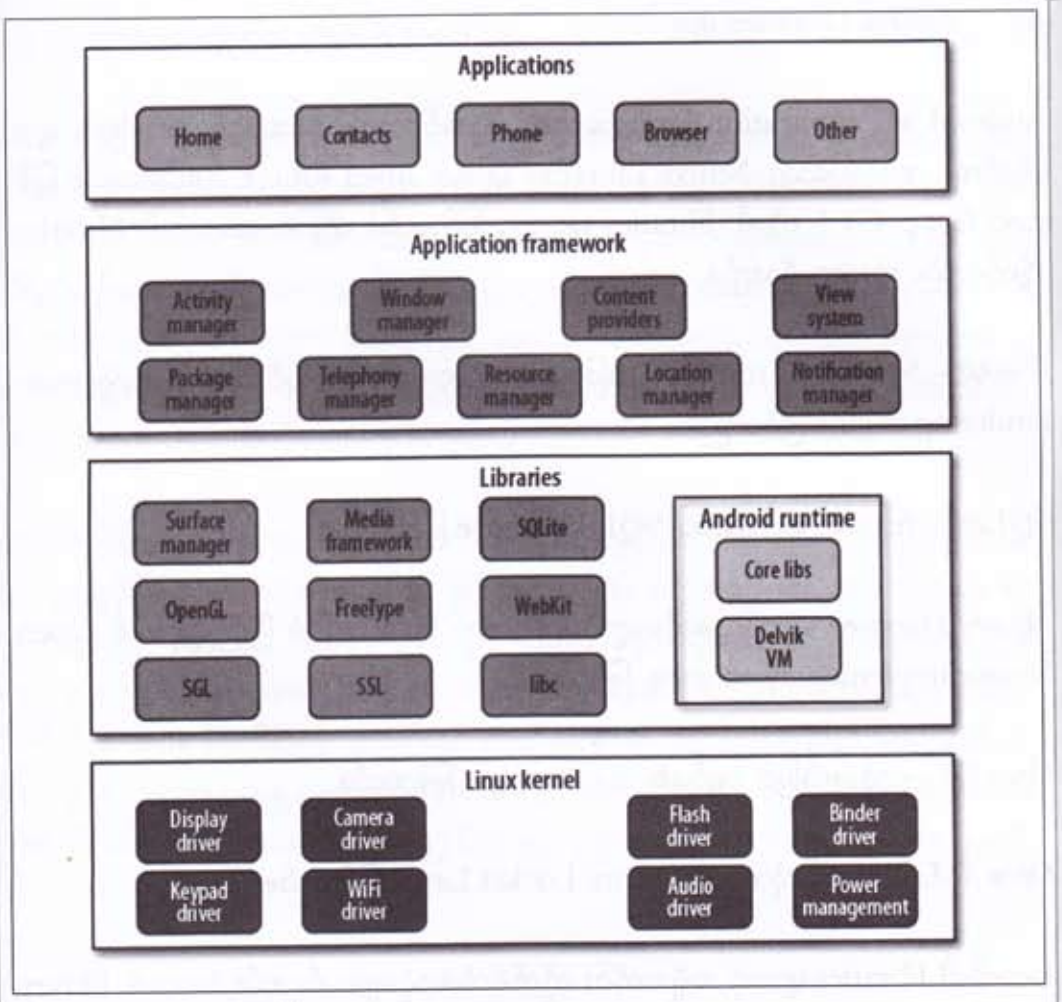
၂.၁ ပေါ့ပါးခြင်း (Portability)

Linux သည် hardware ဗိသုကာစနစ် တော်တော်များများတွင် အလွယ်တကူပေါင်းစပ်နိုင်သော portable platform တစ်ခု ဖြစ်သည်။ Linux မှသည် Android သို့ လက်ဆင့်ကမ်းပေးလိုက်သည်မှာ hardware များကို အတိုင်းအတာတစ်ခုအထိ ချုံ့လိုက်ခြင်း ဖြစ်သည်။ ဆိုလိုသည်မှာ Android ကို Linux ပေါ်တွင် အခြေခံထားခြင်းကြောင့် hardware တို့၏ ရှိနှင့်ပြီး feature များကို ထည့်သွင်းစဉ်းစားရန် မလိုလှတော့ချေ။ Linux ၏ low-level အစိတ်အပိုင်း တော်တော်များများကို ပေါ့ပါးသော C code များဖြင့် ရေးထားသည် ဖြစ်သောကြောင့် (Linux ၏ ဆင့်ပွား) Android ဖြင့် သုံးနေသော ကိရိယာမျိုးစုံအတွက် ဖြည့်စွက် third-party application များကို အလွယ်တကူ ထပ်ဆင့် သွားရောက်ပေါင်းစပ်နိုင်သည်။

Android OS ၏ ဖွဲ့စည်းပုံအလွှာအဆင့်ဆင့် (Android Stack) ကို တစ်ဖက်စာမျက်နှာပါ ပုံတွင် ကြည့်ပါ။

၂.၂ လုံခြုံခြင်း (Security)

Linux သည် လုံခြုံမှု မြင့်သော စနစ်တစ်ခု ဖြစ်သည်။ ကြမ်းတမ်းသော လုပ်ငန်းခွင်အနေအထားတို့၌ စမ်းသပ်မှတ်ကျောက်တင်ခြင်းကို ဆယ်စုနှစ်များစွာ အထပ်ထပ် ခံခဲ့ရသော စနစ်တစ်ခု ဖြစ်သည်။ Android က Linux အပေါ် ကြီးမားစွာ မှီခိုရခြင်း၏ အကြောင်းတစ်ချက်မှာ လုံခြုံမှုကြောင့် ဖြစ်သည်။ Android applicaton အားလုံးသည် Linux system က ချပေးသော သီးခြား ခွင့်ပြုချက်များဖြင့် သီးခြား Linux process လုပ်ဆောင်ချက်များ အဖြစ် အလုပ်လုပ်ကြခြင်း ဖြစ်သည်။ ဤသို့ဖြစ်သောကြောင့် လုံခြုံရေး ဆိုင်ရာ ပူပန်စရာ ကိစ္စ အတော်များများကို အလိုလို ကျော်လွှားသွားပြီး ဖြစ်စေသည်။ ဆိုလိုသည်မှာ app တစ်ခုကို တစ်ခု တည်မှီကာ အလုပ်လုပ်ကြခြင်း မဟုတ်သောကြောင့် app တစ်ခုမှ တစ်ခုသို့ ထိုးဖောက်ဝင်ရောက်ကာ လုံခြုံမှုကို ချိန်းချောက်သော ပြဿနာ ကင်းစေသည်။



၂.၃ Features စုံလင်မှု

Android သည် မိုဘိုင်းကြိုးမဲ့ဆက်သွယ်ရေး ကိရိယာများအတွက် Linux ကဲ့သို့ပင် မှတ်ဉာဏ်ကို ထိန်းချုပ်စီမံနိုင်ခြင်း memory management ၊ ပါဝါကို ထိန်းချုပ်စီမံနိုင်ခြင်း power management နှင့် ကွန်ယက်ဆက်သွယ်မှုစနစ် networking စသည့် စုံလင်လှသော feature များ ရှိသည်။

၂.၄ Native Libraries များ

Android ၏ application layer အတွက် လိုအပ်သော ဝန်ဆောင်မှု services များ ဖန်တီးရန် အသုံးပြုသော Native Libraries များမှာ open source community ဖြစ်သော C နှင့် C++ တို့၏ libraries များသာ ဖြစ်သည်။ ထို့ပြင် အောက်ပါ libraries တို့ကိုလည်း သုံးလေ့ ရှိသည်။

Webkit – Safari ၊ Chrome နှင့် အခြားသော browser များတွင် သုံးထားသည့် web-rendering engine ဖြစ်သည်။

SQLite – feature စုံပါရှိသော SQL Database ဖြစ်သည်။

Apache Harmony – မည်သူမဆိုရယူသုံးစွဲနိုင်သော Java အခြေခံ ပြီးပြည့်စုံစနစ် Open Source implementation တစ်ခု ဖြစ်သည်။

Open GL – သုံးဖက်မြင် ဂရပ်ဖစ် libraries များ ဖြစ်သည်။

Open SSL – အံဝုက်လုံအလွှာ Secure Locket Layer ဖြစ်သည်။

အထက်ပါ libraries များကို အရှိအတိုင်း တိုက်ရိုက်အသုံးချသော်လည်း စံထား C library ကို ပြန်လည်ရေးသားပြီး အသုံးပြုသောမူကွဲတစ်မျိုးအဖြစ် Bionic ကိုလည်း ထည့်သွင်းသုံးစွဲသည်။ Bionic ကို အသုံးပြုရခြင်းမှာ အောက်ပါ အကြောင်းနှစ်ရပ်ကြောင့် ဖြစ်သည်။

နည်းပညာ – သေးငယ်ပြီး ဘက်ထရီကို အသုံးပြုသော ကိရိယာများအတွက် ရည်ရွယ်တည်ဆောက်နိုင်ခြင်း၊
ခွင့်ပြုမှု – ၄င်းကို ပြုပြင်ယူကာ သုံးလိုသူများအတွက် အဆင်ပြေသော license မျိုးရှိခြင်း၊

၂.၅ သိထားသင့်သော Dalvik အကြောင်း

Dalvik ဆိုသည်မှာ Android အတွက် အထူးသီးသန့် ဒီဇိုင်းထုတ်ပြီး Google မှ Dan Bronstein ဦးဆောင်သော အဖွဲ့က တည်ဆောက်ထားသည့် Virtual Machine ဖြစ်သည်။ ရှိနှင့်ပြီးသား Java Virtual Machine ထက် ပိုမိုကောင်းမွန်အောင်၊ ကြိုးမဲ့မိုဘိုင်းကိရိယာများ အတွက် သက်သက် ဦးတည်လျက် ဖန်တီးထားခြင်း ဖြစ်သည်။ Dalvik ကို အဓိကအားဖြင့် battery ကြာချိန်နှင့် မောင်းနှင်တွက်ချက်နိုင်စွမ်းအား processing power တို့ကဲ့သို့သော ကြိုးမဲ့မိုဘိုင်းကိရိယာများ၏ အကန့်အသတ်များအတွက် အဆင်ပြေစေရန် အဓိက ဒီဇိုင်းထုတ်ထားသည်။

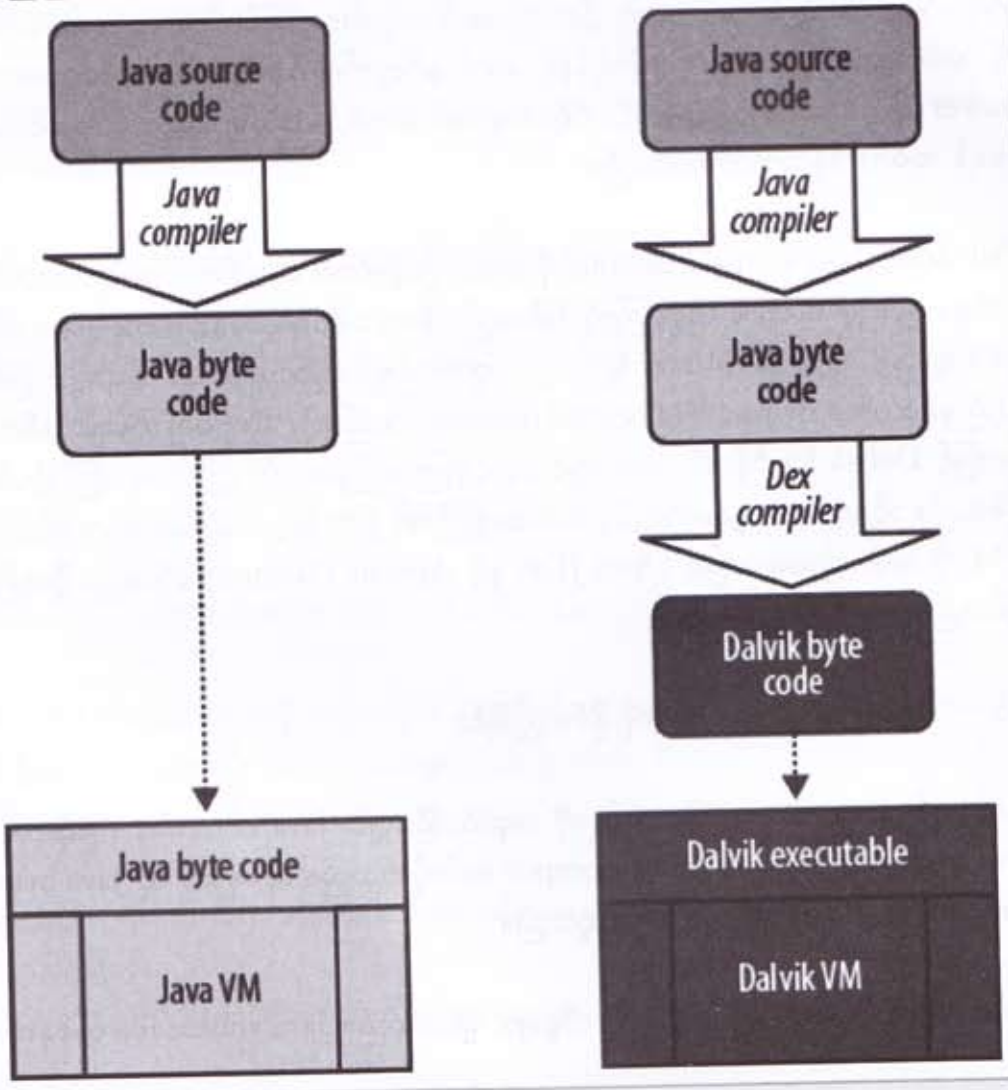
Dalvik ဖြင့် Java Virtual Machine ကို အစားထိုးရခြင်း၏ နောက်ထပ် အကြောင်းတစ်ချက်မှာ ခွင့်ပြုမှု license ကိစ္စကြောင့် ဖြစ်သည်။ Java ပရိုဂရမ်ဘာသာစကား၊ Java ၏ tool များနှင့် Java ၏ library များသည် လွတ်လပ်စွာ ယူသုံးနိုင်သော အရာများ ဖြစ်သည် မှန်သော်လည်း Java VM (virtual machine) သည်ကား free မဟုတ်ချေ။ အဓိကအားဖြင့် Dalvik စီမံကိန်းကို ၂၀၀၅ တွင် စတင်ခဲ့ကြသည်မှာလည်း ဤပြဿနာကြောင့်ပင် ဖြစ်သည်။ သို့သော် ယနေ့ခေတ်တွင်မူ Sun အဖွဲ့ကြီး၏ Java ဖြင့် တည်ဆောက်ထားသော VM ကို အစားထိုးစရာအဖြစ် Open JDK နှင့် Apache Harmony တို့လည်း ရှိနေပြီ ဖြစ်သည်။

၂.၆ Android နှင့် Java တို့ကို နှိုင်းယှဉ်ခြင်း

Java တွင် Java source file တစ်ခုကို ရေးသားပြီးလျှင် Java compiler ကိုသုံးကာ Java byte code အဖြစ် ပြောင်းလဲ compile လုပ်လိုက်သည်။ ထို့နောက် ထို Java byte code ကို Java VM ပေါ်တွင် run ယူသည်။

Android ပေါ်တွင်မူ မတူတော့သည့် ကိစ္စများ ရှိလာသည်။ Java source file တစ်ခုကို

ရေးသားပြီးလျှင် Java compiler ကိုသုံးကာ Java byte code အဖြစ် ပြောင်းလဲ compile လုပ်လိုက်သည်အထိကား အတူတူဖြစ်သည်။ သို့သော် ထိုအဆင့်တွင် Dalvik compiler ကို ထပ်ဖြတ်ပြီး Dalvik byte code အဖြစ် ထုတ်ယူသည်။ ရရှိလာသော Dalvik byte code ကိုမှ Dalvik VM ပေါ်တွင် အလုပ်လုပ်စေခြင်း ဖြစ်သည်။ အောက်ပါ သရုပ်ဖော်ပုံကို ကြည့်ပါ။



Dalvik byte code ဖြစ်အောင် Java မှ တိုက်ရိုက် compile လုပ်မယူဘဲ Dalvik compiler ကို တဆင့် ထပ်ခံခြင်းကြောင့် အကျိုးဆက်များ ဖြစ်လာသည်။ ဥပမာ – Java byte code ဖြစ်စေသော (Python သို့မဟုတ် Ruby စသည့်) မည်သည့် ပရိုဂရမ်ဘာသာစကားဖြင့် မဆို Android အတွက် application များ ရေးသားနိုင်သွားခြင်းမျိုး ဖြစ်သည်။

နောက်တစ်ချက်မှာ Android အတွက် Java သည် Java classes တို့တွင် စံထား class မဟုတ်ချေ။ Java ၏ စံထားအဆင့်များမှာ

Java Standard Edition (*JSE*) – အခြေခံ desktop computer များအတွက် application များ တည်ဆောက်ရာတွင် အသုံးပြုသည်။

Java Enterprise Edition (*J2EE* or *JavaEE*) – လုပ်ငန်းကြီးသုံး Enterprise application များ တည်ဆောက်ရာတွင် အသုံးပြုသည်။

Java Micro Edition (*J2ME* or *JavaME*) – ကြိုးမဲ့မိုဘိုင်းကိရိယာသုံး application များ အတွက် ဖြစ်သည်။

Android တွင်ပါဝင်သည့် Java library set များမှာ Java Standard Edition နှင့် အနီးစပ်ဆုံး ဖြစ်သည်။ ထိုနှစ်ခု၏ အဓိက ကွာခြားချက်မှာ JSE မှ user interface libraries များကို ဖယ်ထုတ်လိုက်ပြီး Android အဓိက user interface libraries များ ကို ဖြည့်သွင်းထားခြင်း ဖြစ်သည်။ သို့သော် Java ၏ standard feature များကို ထောက်ပံ့ပေးထားသည်နှင့် တပြိုင်နက်တွင် feature သစ်များကိုလည်း Android ၌ ဖြည့်စွက် ထည့်သွင်းထားပေးပါသည်။

၂.၇ Application Framework နှင့် ပတ်သက်၍

Android ၏ application framework သည် application developer တစ်ယောက်အဖို့ အလုပ်ပြီးမြောက်ရေးအတွက် အထောက်အကူပေးနိုင်သည့် service များစွာပါသော လုပ်ငန်းခွင် environment တစ်ခု ဖြစ်သည်။ Application Framework သည် Android platform ကြီး တစ်ခုလုံးအနက် developer များနှင့် အသက်ဆိုင်ဆုံး၊ ထိုပြင် အရေးအကြီးဆုံး အစိတ်အပိုင်းလည်း ဖြစ်သည်။

Android အတွက် အထူးတည်ဆောက်ထားသည့် Java libraries ပေါင်းများစွာကိုလည်း Application Framework အလွှာပေါ်တွင် တွေ့ရမည် ဖြစ်သည်။ ကိုယ်တည်ဆောက်သော application မှ ဖောက်ဝင်သုံးစွဲနိုင်သည့် ဝန်ဆောင်မှု service များ (Android အခေါ် အားဖြင့် *manager* များ) ဖြစ်သည့် တည်နေရာရှာစနစ်များ locations ၊ အာရုံခံမှုစနစ်များ sensors ၊ ကြိုးမဲ့အဝေးမြန်နှုန်းမြင့်ဆက်သွယ်ရေးစနစ် Wi Fi ၊ တယ်လီဖုန်းဆိုင်ရာ telephony စသည် တို့ကိုလည်း ဤအလွှာပေါ်၌ပင် တွေ့ရမည်။

Android application development ကို လေ့လာသူတစ်ဦးအနေဖြင့် Android stack ပေါ်တွင် ဤအပိုင်းသည် အဓိက ဦးတည်လေ့လာရမည့် အလွှာဖြစ်ပြီး၊ ဤ Application Framework ပေါ်မှ ပါဝင်သောအရာ component များစွာကိုလည်း ဆွဲယူအသုံးပြုရမည် ဖြစ်သည်။

၂.၈ Application များနှင့် ပတ်သက်၍

Application တစ်ခုသည် တစ်စုတည်း single application package (APK) ဖိုင်တစ်ဖိုင် ဖြစ်သည်။ APK ဖိုင်တစ်ခုတွင် အကြမ်းအားဖြင့် ပါဝင်မှု သုံးခု ရှိသည်။

Dalvik executable – *Dalvik executable* အဖြစ် ပြောင်းလဲထားသော application ဆိုင်ရာ Java source code အားလုံးစုထားသည့် ဖိုင်ဖြစ်ပြီး ၄င်းသည် ကိုယ့် application ကို တကယ် run ပေးမည့် code ဖြစ်သည်။

Resources – code မဟုတ်သော အရာအားလုံး အစုအဝေး ဖြစ်သည်။ ကိုယ့် application တွင် ရုပ်ပုံများ၊ အသံဖိုင် ရုပ်သံဖိုင် audio / video clip များ၊ အခင်းအကျင်း layout နှင့် ဘာသာစကားဆိုင်ရာ language pack အစုအဝေးတို့ကို သတ်မှတ်ပေးမည့် XML file များ စသည်တို့သည် resources များပင် ဖြစ်သည်။

Native libraries – ရံဖန်ရံခါတွင် C / C++ libraries များကဲ့သို့သော native code များ ကိုယ့် application တွင် ပါခဲ့လျှင် ဤ libraries များ ကို APK file တွင် ပေါင်းထည့် နိုင်သည်။

Android application များကို device တစ်ခုခုတွင် ထည့်သွင်း install လုပ်မည်ဆိုလျှင် ပထမဦးစွာ sign လုပ်ရသည်။ နမူနာ applicationကို debug key ဖြင့် sign လုပ်ပြထားပါ သည်။ ကိုယ်ပိုင် application များကို ဈေးကွက်သို့ဖြန့်ချိမည်ဆိုလျှင် မူ ကိုယ်ပိုင် key ဖြင့် sign လုပ်သင့်ပါသည်။

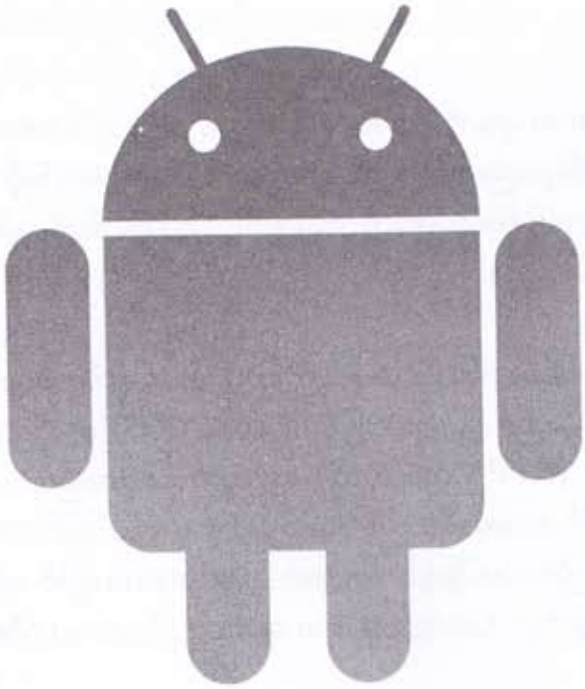
Android နှင့် အခြားသော platform များ ကွဲပြားသော အချက်တစ်ချက်မှာ application များ ဖြန့်ချိခြင်း ဖြစ်သည်။ ဥပမာအားဖြင့် iPhone (iOS) ကဲ့သို့သော မောင်းနှင်စနစ် platform များတွင် (APP store ကဲ့သို့) ဈေးကွက်တစ်ခုတည်းက application များ ရောင်းချမှုကို ချုပ်ကိုင်ထားလေ့ရှိသည်။ Android မှာမူ လွတ်လပ်သော ဈေးကွက်အသွင် ကို ပိုမိုဖော်ဆောင်သည်။ လက်ရှိတွင် Android platform အတွက် အကြီးဆုံး Android application ဈေးကွက်မှာ Google မှ run နေသော 'Android Market' ဖြစ်သည်။

ဝင်းဟန်း

B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Application များကို web (internet) မှ ဖြန့်ဖြူးပါက၊ ကြိုးမဲ့မိုဘိုင်းဆက်သွယ်ရေး ကိရိယာ၏ browser မှတဆင့် APK ဖိုင်ကို download ဆွဲလျှင်၊ ထို APK ဖိုင် အဖြစ် တည်ဆောက်ထားသော application သည် ကြိုးမဲ့မိုဘိုင်းဆက်သွယ်ရေး ကိရိယာတွင် အလိုလျှောက် install လုပ်သွားပါမည်။

Android ၏ လွတ်လပ်သော ဖြန့်ချိမှုသဘောသဘာဝကြောင့် user တစ်ဦးသည် ဒုက္ခပေးမည့် malicious application တစ်ခုခုကို မသိနိုးနား install လုပ်မိမည့် ဖြစ်နိုင်ချေ ရှိပါသည်။



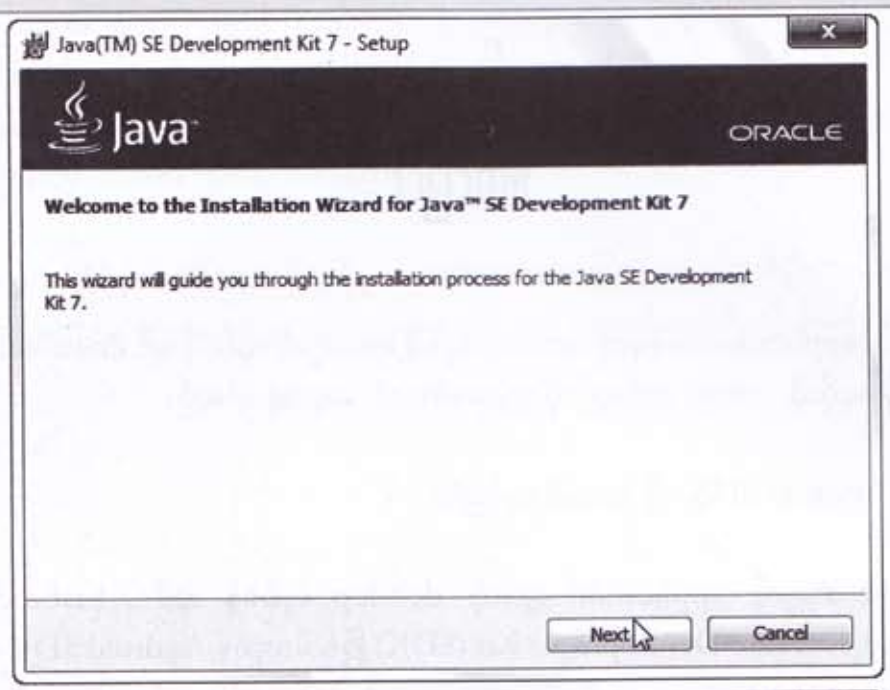
အခန်း – ၃
စတင်ခြင်း

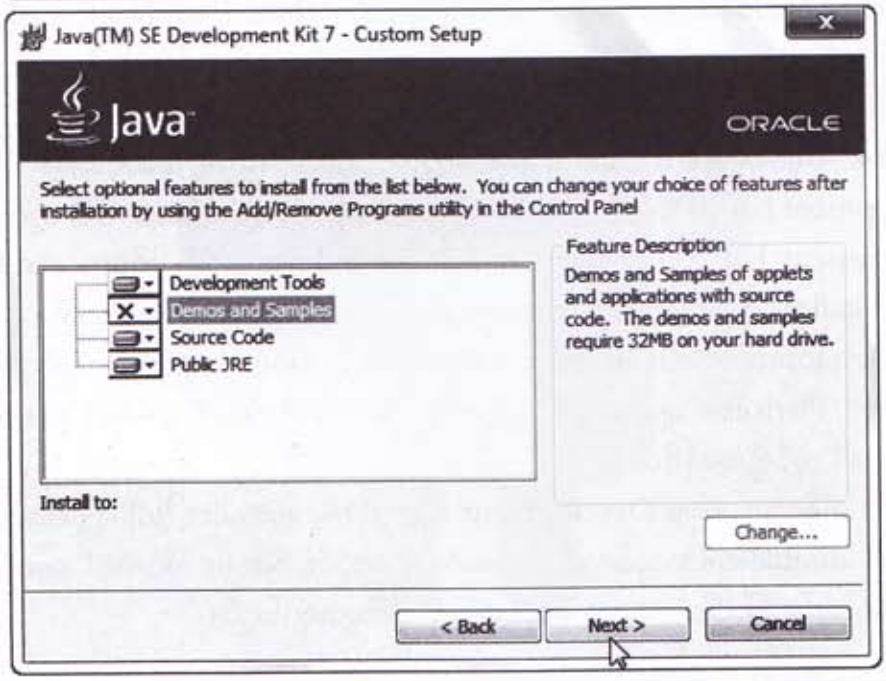
Android application တစ်ခုကို develop လုပ်ဖို့ရာအတွက် လုပ်ငန်းခွင် environment တစ်ခုကို မည်သို့ set up လုပ်မည်ကို ဤအခန်းတွင် ဆွေးနွေးပါမည်။

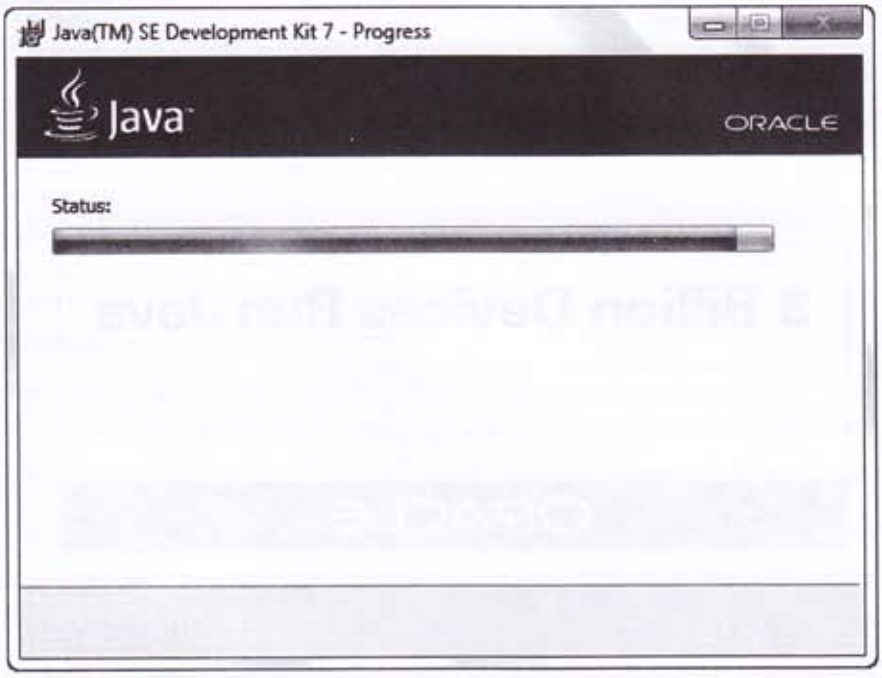
၃.၁ Android SDK ကို install လုပ်ခြင်း

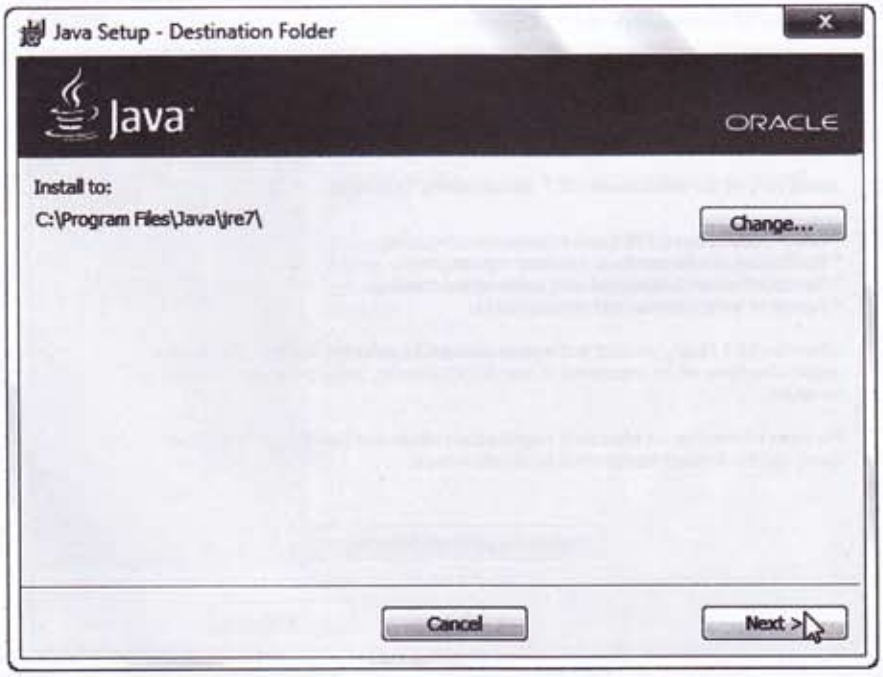
Android အတွက် application များကို develop လုပ်ရန် အဓိကလိုအပ်သည်မှာ Android Software Development Kit (SDK) ဖြစ်ပါသည်။ Android SDK တွင် tool အစုအဝေးတစ်ခုသာမက ကိုယ် develop လုပ်လိုက်သော application ကို run ပြီး မည်ကဲ့သို့ အလုပ်လုပ်သည်ကို ကြည့်နိုင်ရန် platform တစ်ခုပါ ပါရှိပါသည်။

Android ပရိုဂရမ်များကို ရေးသားရန်အတွက် ပထမဦးဆုံး အဆင့်အဖြစ် Java Development Kit (JDK) ရှိရပါမည်။ ဤစာအုပ်နှင့် တွဲလျက်ပါသော CD တွင် Java Development Kit (JDK) installer (jdk-7-windows-i586) ကိုရော Android SDK installer (installer_r13-windows) ကိုပါ ထည့်သွင်းပေးထားပါသည်။
(Java Development Kit (JDK) installer (jdk-7-windows-i586) သည် 32-bit Windows Platform များအတွက် ဖြစ်ပါသည်။ Windows 7 64-bits platform တွင် install လုပ်၍ မရနိုင်ပါ။)
CD တွင် ပါရှိသော Java Development Kit (JDK) installer (jdk-7-windows-i586) ကို double-click လုပ်လျှင် တစ်ဖက်ပါပုံအတိုင်း Set-up Wizard အဆင့်ဆင့် တက်လာပါမည်။ မြားပြသွားသော ခလုပ်များကို နှိပ်သွားရပါမည်။


Java(TM) SE Development Kit 7 - Setup
Java
ORACLE
Welcome to the Installation Wizard for Java™ SE Development Kit 7
This wizard will guide you through the installation process for the Java SE Development Kit 7.
Next
Cancel


Java(TM) SE Development Kit 7 - Custom Setup
Java
ORACLE
Select optional features to install from the list below. You can change your choice of features after installation by using the Add/Remove Programs utility in the Control Panel
Development Tools
Demos and Samples
Source Code
Public JRE
Feature Description
Demos and Samples of applets and applications with source code. The demos and samples require 32MB on your hard drive.
Install to:
Change...
< Back
Next >
Cancel

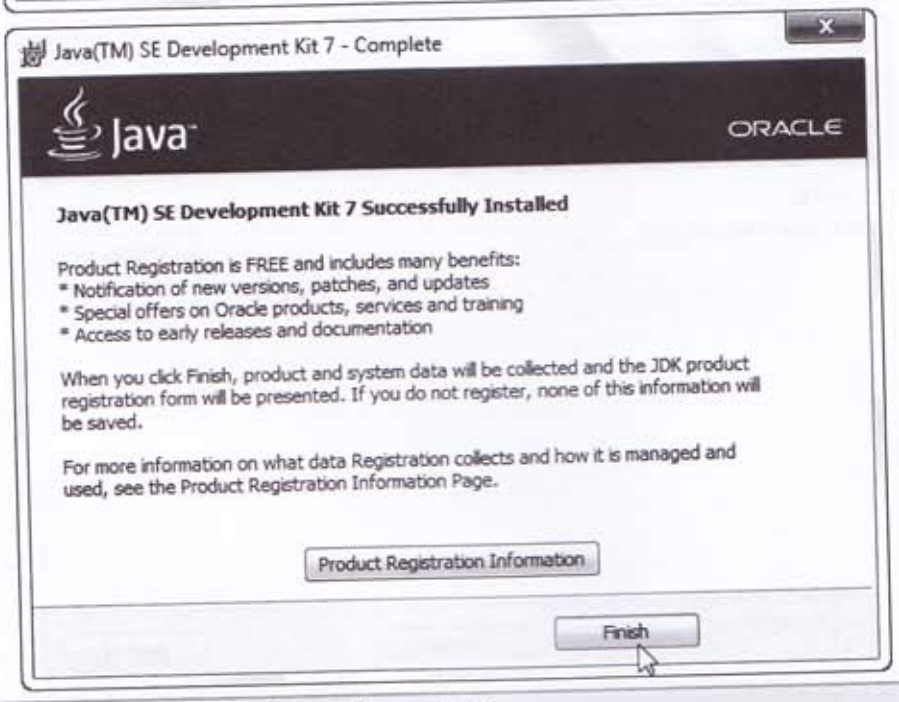


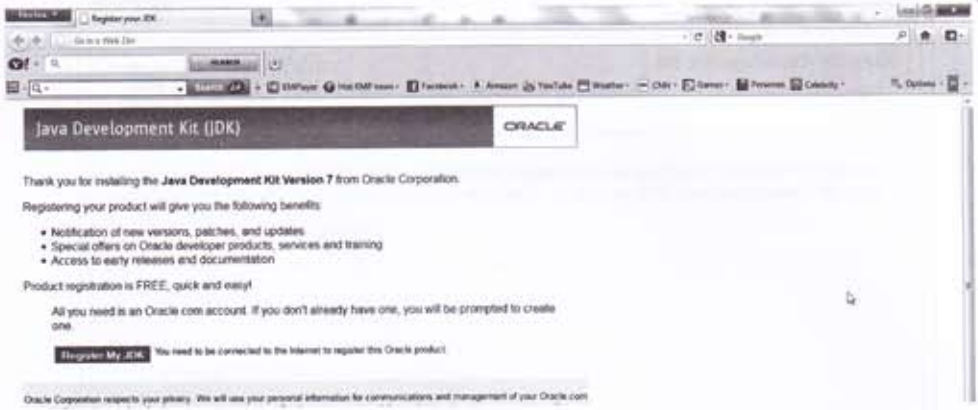


ဝင်းဟန်း

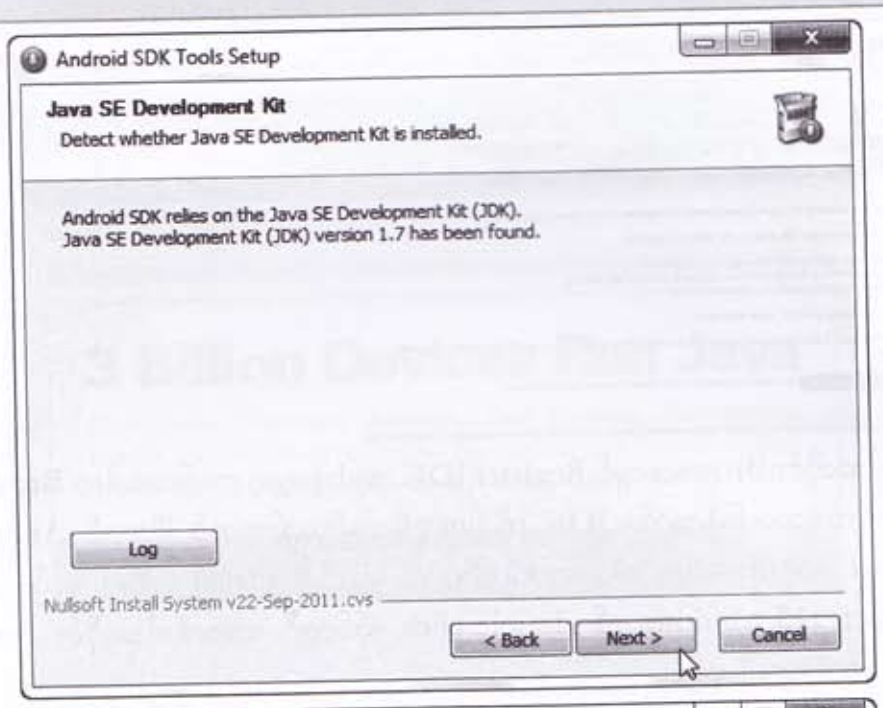
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

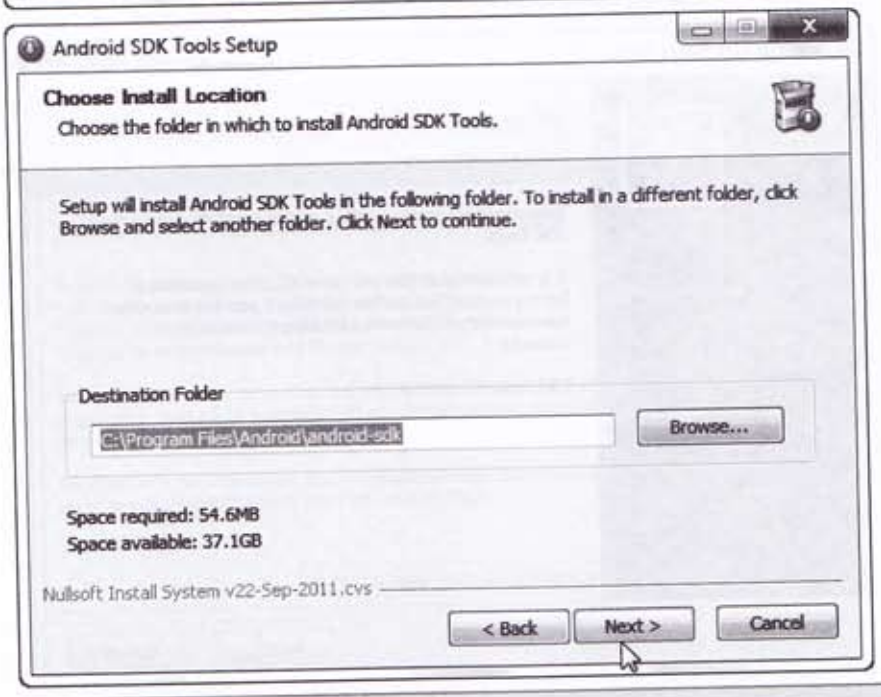

Java Setup - Progress
Java
ORACLE
Status:
3 Billion Devices Run Java
Computers, Printers, Routers, Cell Phones, BlackBerry, Kindle, Parking Meters, Public Transportation Passes, ATMs, Credit Cards, Home Security Systems, Cable Boxes, TVs...
ORACLE

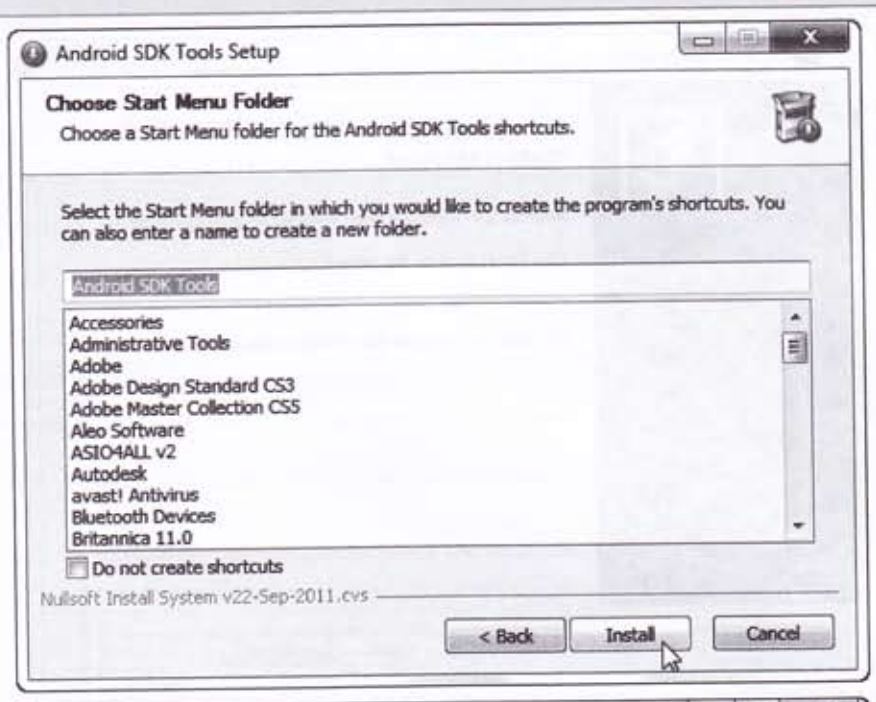

Java(TM) SE Development Kit 7 - Complete
Java
ORACLE
Java(TM) SE Development Kit 7 Successfully Installed
Product Registration is FREE and includes many benefits:
* Notification of new versions, patches, and updates
* Special offers on Oracle products, services and training
* Access to early releases and documentation
When you click Finish, product and system data will be collected and the JDK product registration form will be presented. If you do not register, none of this information will be saved.
For more information on what data Registration collects and how it is managed and used, see the Product Registration Information Page.
Product Registration Information
Finish

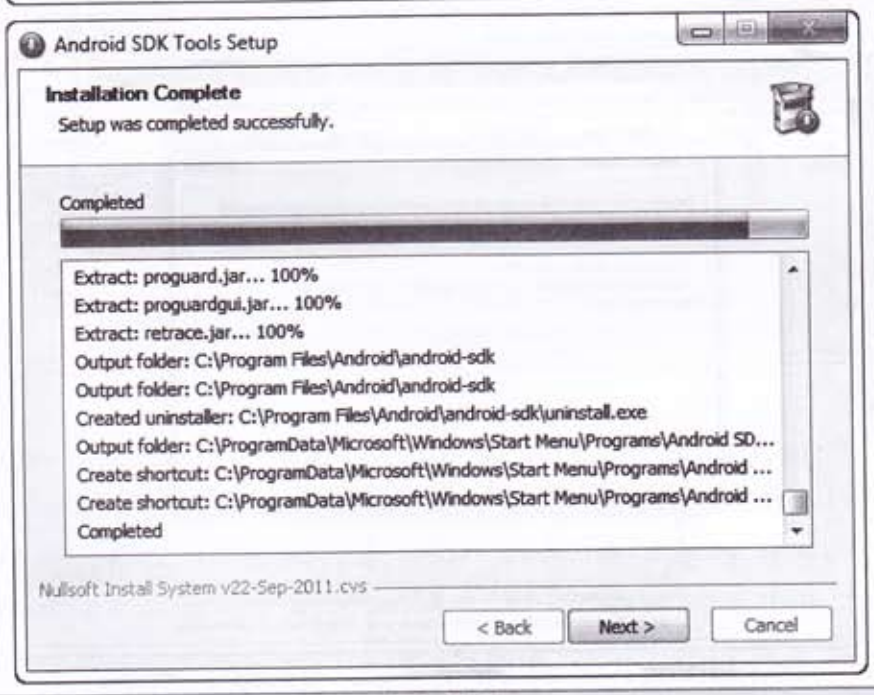


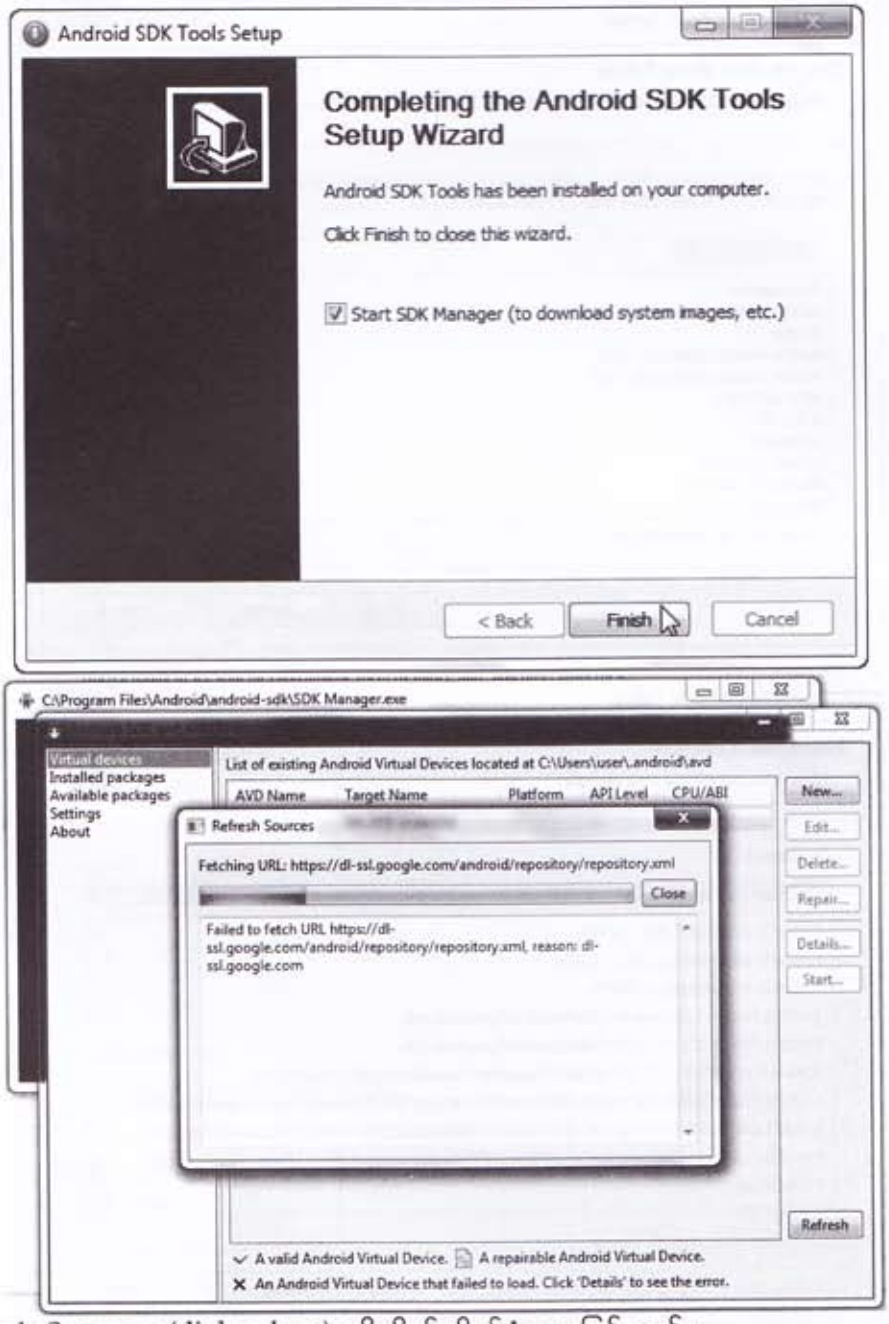
အထက်ပါအတိုင်း Browser တွင် Register JDK (web page) တက်လာပါက Browser ကို ပိတ်လိုက်ရုံသာ ဖြစ်သည်။ JDK ကို install လုပ်သည့်အဆင့် ပြီးလျှင် Android SDK installer (installer_r13-windows) ကို ဆက်၍ install လုပ်ရပါမည်။ CD ရှိ installer_r13-windows ကို double-click လုပ်လျှင် အောက်ပါအတိုင်း SetUp Wizard တက်လာပါမည်။


Android SDK Tools Setup
Java SE Development Kit
Detect whether Java SE Development Kit is installed.
Android SDK relies on the Java SE Development Kit (JDK).
Java SE Development Kit (JDK) version 1.7 has been found.
Log
Nullsoft Install System v22-Sep-2011.cvs
< Back
Next >
Cancel


Android SDK Tools Setup
Choose Install Location
Choose the folder in which to install Android SDK Tools.
Setup will install Android SDK Tools in the following folder. To install in a different folder, click Browse and select another folder. Click Next to continue.
Destination Folder
C:\Program Files\Android\android-sdk
Browse...
Space required: 54.6MB
Space available: 37.1GB
Nullsoft Install System v22-Sep-2011.cvs
< Back
Next >
Cancel

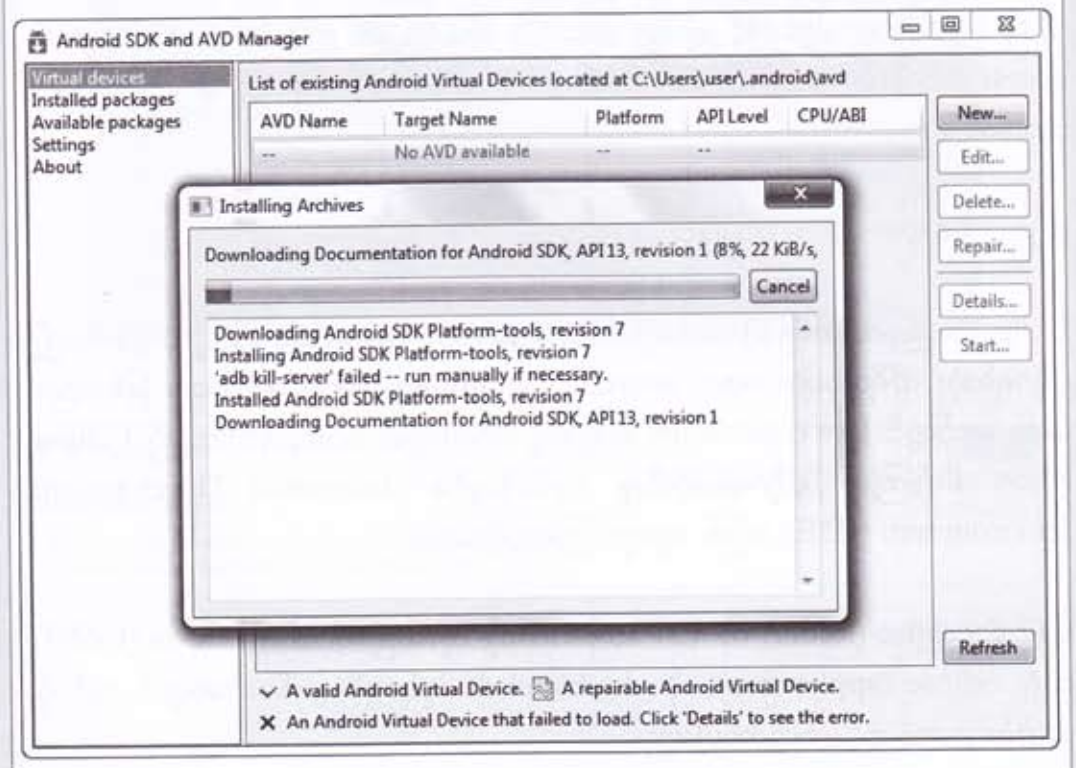

Android SDK Tools Setup
Choose Start Menu Folder
Choose a Start Menu folder for the Android SDK Tools shortcuts.
Select the Start Menu folder in which you would like to create the program's shortcuts. You can also enter a name to create a new folder.
Android SDK Tools
Accessories
Administrative Tools
Adobe
Adobe Design Standard CS3
Adobe Master Collection CS5
Aleo Software
ASIO4ALL v2
Autodesk
avast! Antivirus
Bluetooth Devices
Britannica 11.0
Do not create shortcuts
Nullsoft Install System v22-Sep-2011.cvs
< Back
Install
Cancel


Android SDK Tools Setup
Installation Complete
Setup was completed successfully.
Completed
Extract: proguard.jar... 100%
Extract: proguardgui.jar... 100%
Extract: retrace.jar... 100%
Output folder: C:\Program Files\Android\android-sdk
Output folder: C:\Program Files\Android\android-sdk
Created uninstaller: C:\Program Files\Android\android-sdk\uninstall.exe
Output folder: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Android SD...
Create shortcut: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Android ...
Create shortcut: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Android ...
Completed
Nullsoft Install System v22-Sep-2011.cvs
< Back
Next >
Cancel



Refresh Sources (dialog box) ကို ဝိတ်လိုက်ရုံသာ ဖြစ်သည်။

Android SDK ၏ package များကို အပြည့်အစုံ အသုံးပြုလိုလျှင် မိမိ ကွန်ပျူတာကို အင်တာနက်နှင့် ချိတ်ဆက်ပြီး SDK manager ကို ဖွင့်လိုက်ပါက (ရှေ့စာမျက်နှာပါ ဒုတိယပုံအတိုင်း) Refresh Sources တက်လာချိန်တွင် အောက်ပါအတိုင်း download ရယူသွားပါမည်။



Windows (operating system) ၌ Android SDK project များအတွက် folder များ တည်ဆောက်ရာတွင် အမည်များ၌ space များ မပါသင့်ပါ။command line များ ဖြင့် အလုပ်လုပ်ရမည် ဖြစ်သောကြောင့် folder တို့၏ name များမှ space များသည် command line ရေးရာတွင် အလုပ်ပိုစေပါသည်။ C:\ အောက်တွင် apps ဟု အမည် ပေးကာ folder ဆောက်လိုက်ရန် အကြံပြုလိုပါသည်။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Android SDK တွင် အဓိက tool များ အကုန်ပါဝင်သည့် folder တစ်ခု ရှိပါသည်။ ဤ tool များကို command-line ဖြင့် ခေါ်ယူသုံးစွဲမည် ဖြစ်သောကြောင့် system PATH တွင်

C:\android-sdk\tools\ နှင့်

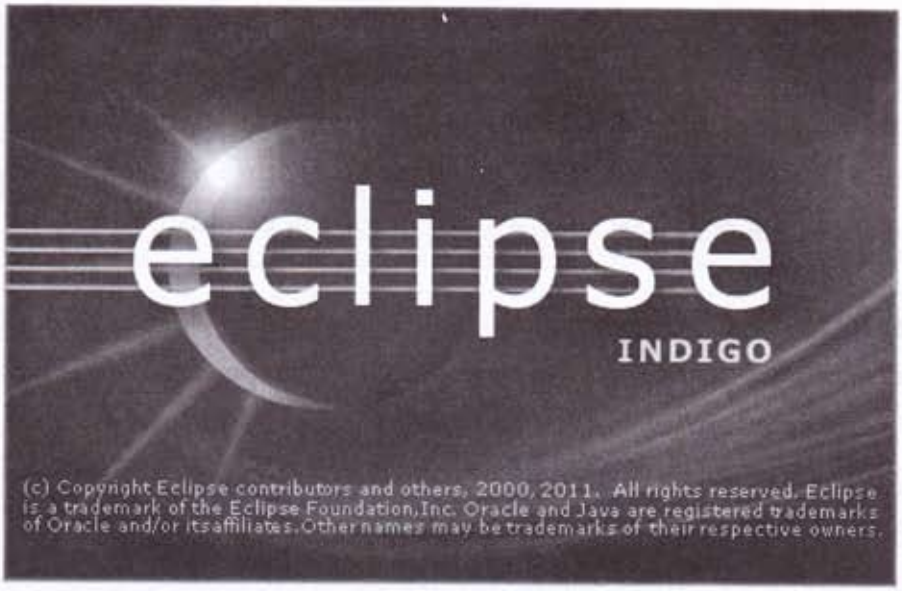
C:\android-sdk\platform-tools\

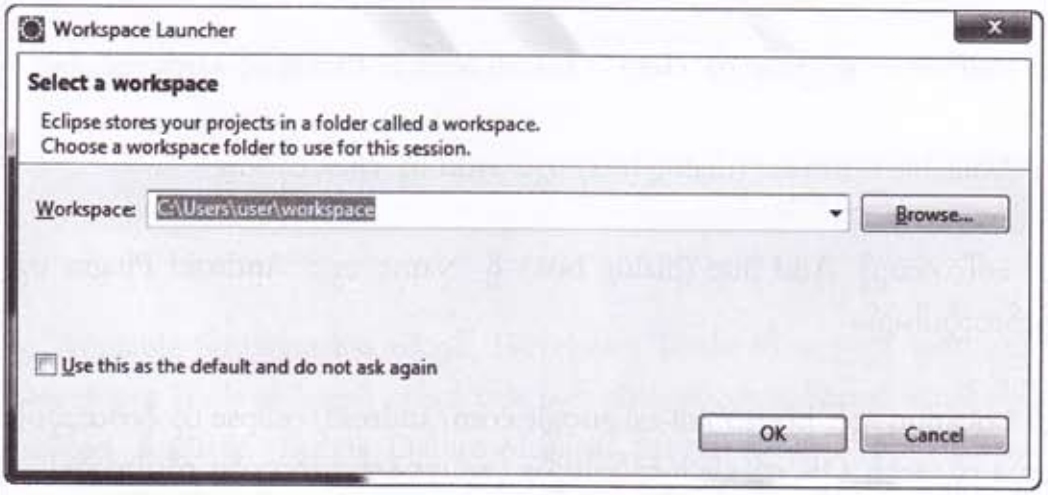
တို့ကို ဖြည့်စွက်ထားခြင်းဖြင့် များစွာ အဆင်ပြေစေပါသည်။ ဤနည်းဖြင့် tool များကို access လုပ်လိုတိုင်း သက်ဆိုင်ရာ folder များကို သွားကာဖွင့်နေရသည့် အလုပ် သက်သာ သွားစေပါမည်။

၃.၂ Eclipse ကို အသုံးပြုခြင်း

Eclipseသည်မူလအားဖြင့် Javaအတွက် IBMမှ ဖန်တီးသောမည်သူမဆိုရယူသုံးစွဲနိုင်သည့် ပရိုဂရမ်သုံး ကိရိယာများ open source programming tools အစုအဝေး ဖြစ်သည်။ ယနေ့ခေတ်တွင် Java community အတွင်းရှိ developer အားလုံးနီးပါးသည် Eclipse ကိုသာ စိတ်ကြိုက် ပရိုဂရမ်ဖန်တီးမှု လုပ်ငန်းခွင်စု Integrated Development Environment (IDE) အဖြစ် ရွေးချယ်ကြလေ့ရှိသည်။

CD ရှိ eclipse (folder) ကို C:\ အောက်ထဲသို့ ကူးထည့်လိုက်ပြီး၊ eclipse (folder) ထဲရှိ eclipse (application) ကို double-click လုပ်လျှင် တစ်ဖက်စာမျက်နှာပါ ပုံ အတိုင်း အဆင့်ဆင့် တက်လာပါမည်။





Workspace Launcher တွင်လည်း ရှေ့အဆင့်က တည်ဆောက်ခဲ့သော C:\ အောက်ရှိ apps (folder) ကိုသာ path ပေးရန် အကြံပြုလိုပါသည်။

မင်ဦးဟန်

B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Eclipse တွင် အချိန်ကုန်သက်သာစေသော feature များ အများအပြား ပါရှိသည်ကို ရှေ့လျှောက်တွင် တွေ့ရပါမည်။ သတိထားရန်တစ်ချက်မှာ Eclipse သည် powerful ဖြစ်သည်နှင့်အမျှ ကွန်ပျူတာ၏ system resource ကိုလည်း အလွန်သုံးသောကြောင့် Eclipse ကိုသုံးရသည်မှာ နေးကွေးလာပြီဟု ယူဆလျှင် Eclipse ကို ပိတ်ပြီး တစ်ခေါက် ပြန်ဖွင့်သင့်ပါသည်။ ဖြစ်နိုင်လျှင် ကွန်ပျူတာကို restart ချသင့်ပါသည်။

Eclipse သည် ကိုယ့်လုပ်ငန်းအားလုံးကို project များအဖြစ် ခွဲထုတ်စုစည်းပေးသည်။ Projectများကို Workspaceများအဖြစ် ထားရှိသည်။ Workspace သည် ရှေ့စာမျက်နှာရှိ ပုံတွင် ပြထားခဲ့သည့်အတိုင်း folder အဖြစ် တည်ရှိသည်။ Windows မောင်းနှင်စနစ်တွင် workspace များ၏ folder name ကို အမည်ပေးရာ၌ space များ မပါလျှင် ပြဿနာ ကင်းပြီး အလုပ်ရှုပ်သက်သာစေပါသည်။

၃.၃ Eclipse တွင် Android အတွက် Development Tools များ တပ်ဆင်ခြင်း

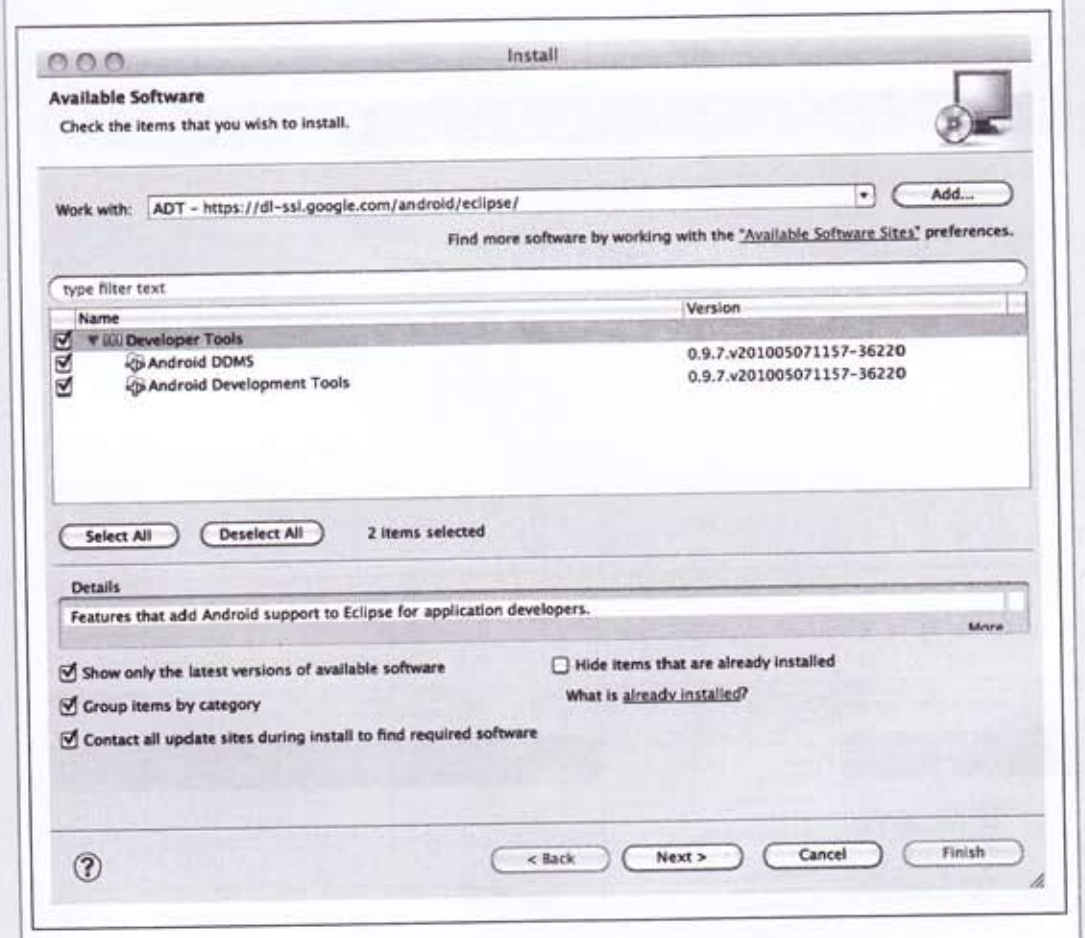
၁. Eclipse ကို ဖွင့်ပြီးနောက် Help > Install New Software ကို click လုပ်ပါ။

၂. Available Software (dialog box) တွင် Add ကို click လုပ်ပါ။

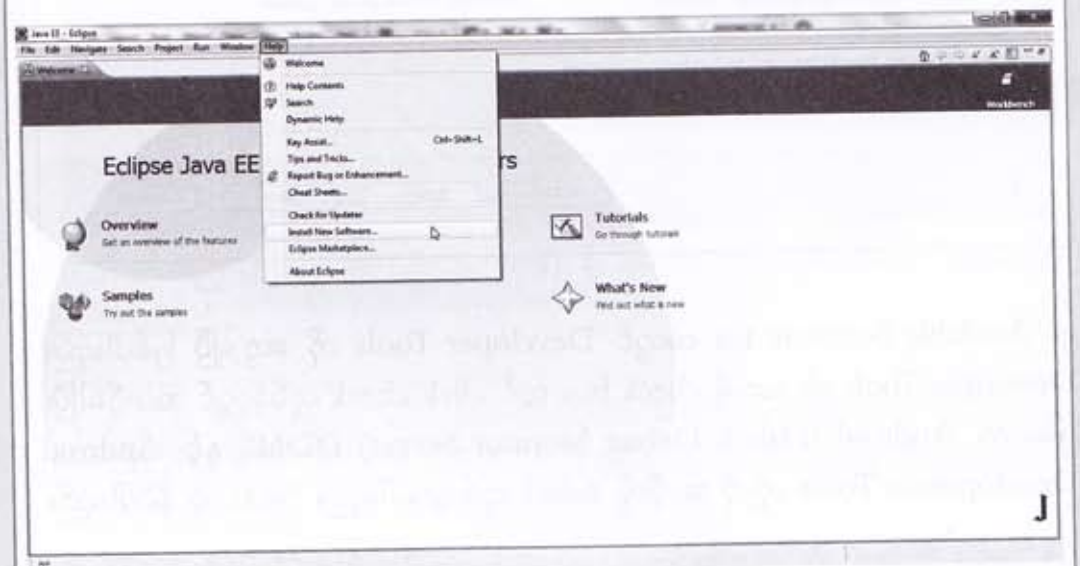
၃. ပေါ်လာသည့် Add Site (dialog box) ရှိ Name တွင် Android Plugin ဟု ရိုက်ထည့်ပါမည်။

၄. Location တွင် https://dl-ssl.google.com/android/eclipse ဟု ရိုက်ထည့်ပါမည်။ ထို့နောက် OK ကို click လုပ်ပါမည်။ (အင်တာနက်ဆက်သွယ်မှု လိုပါသည်။)

တစ်ဖက်စာမျက်နှာပါပုံအတိုင်း ဖြစ်သွားပါမည်။ (အဆင့်လေးဆင့်၏ ပုံများကို စာမျက်နှာ ၃၄ – ၃၇) တွင် ကြည့်ပါ။

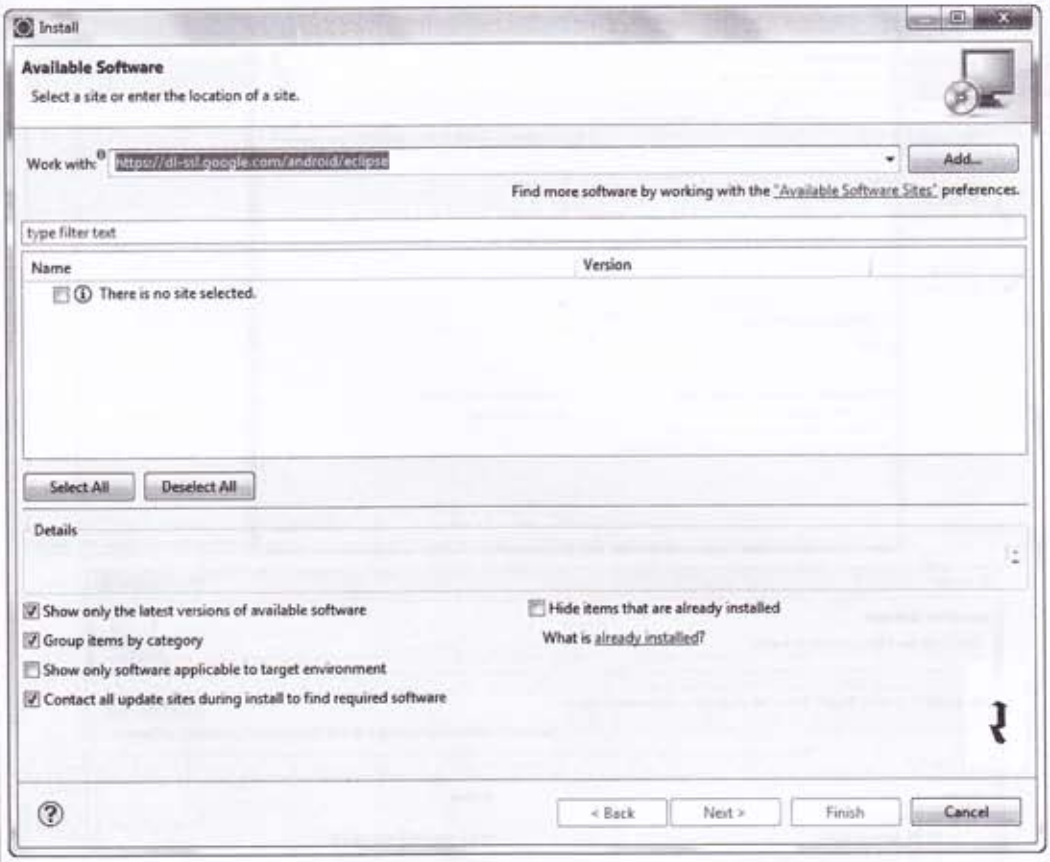


၅. Available Software list ထဲတွင် Developer Tools ကို တွေ့ရပြီး ဖြစ်ပါမည်။ Developer Tools ၏ ဘေးရှိ check box တွင် click-check လုပ်ခဲ့လျှင် အသင့်ပါပြီး ဖြစ်သော Android (Dalvik Debug Monitor Server) DDMS နှင့် Android Development Tools တို့ကို အလိုလို select လုပ်သွားပါမည်။ Next ကို နှိပ်ပါမည်။

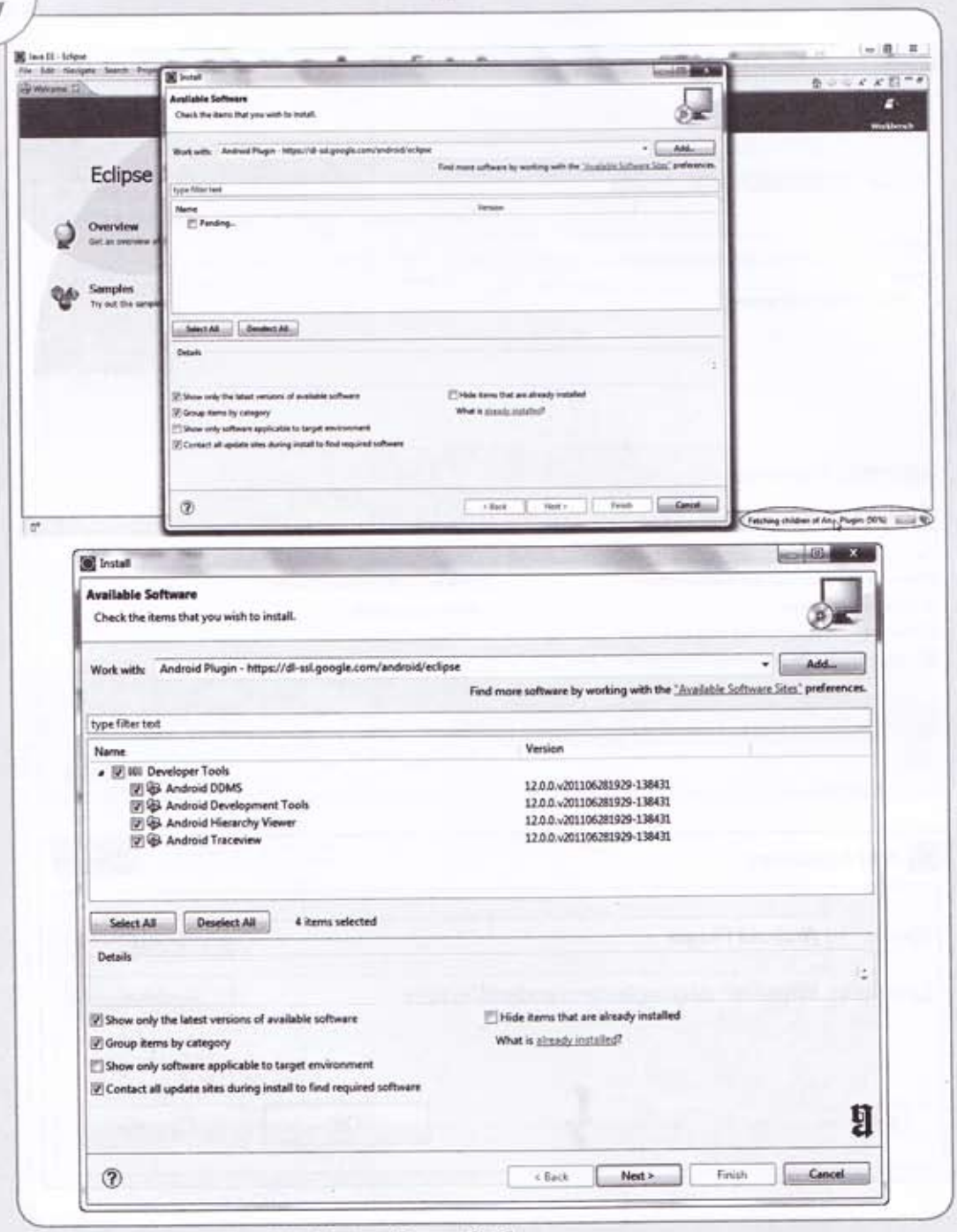


Android application

ရေးသားပန်တီးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

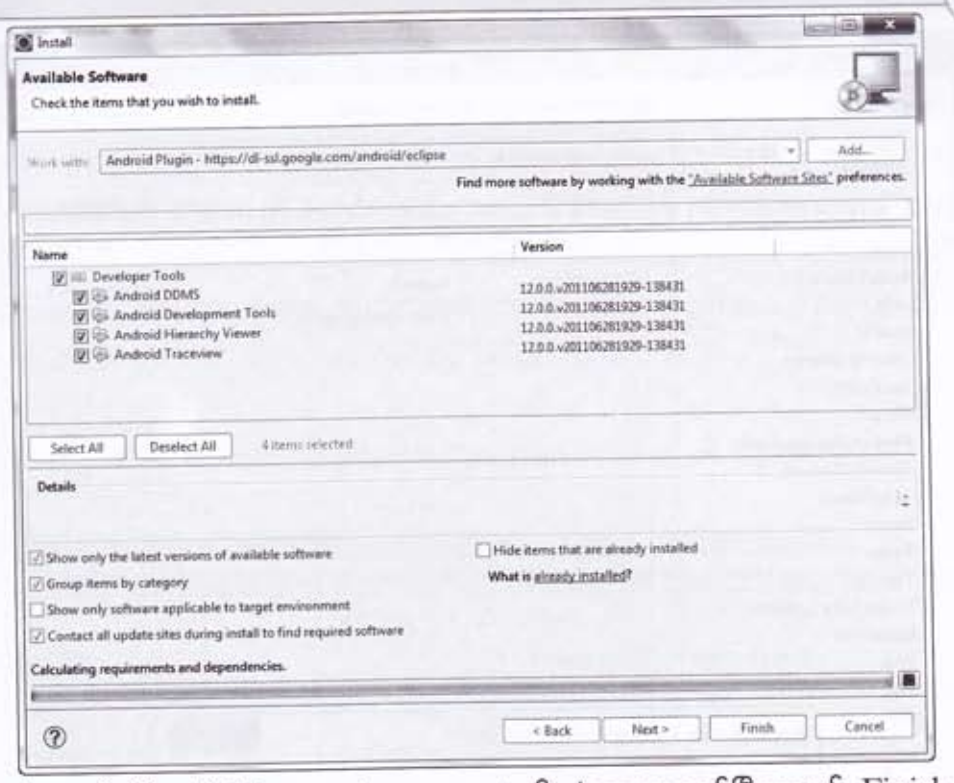

Install
Available Software
Select a site or enter the location of a site.
Work with: https://dl-ssl.google.com/android/eclipse
Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name
Version
There is no site selected.
Select All
Deselect All
Details
Show only the latest versions of available software
Group items by category
Show only software applicable to target environment
Contact all update sites during install to find required software
Hide items that are already installed
What is already installed?
< Back
Next >
Finish
Cancel


Add Repository
Name: Android Plugin
Location: https://dl-ssl.google.com/android/eclipse
Local...
Archive...
OK
Cancel

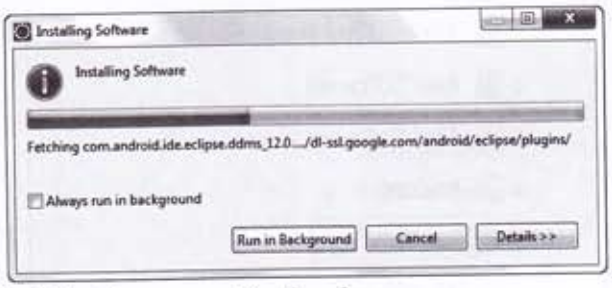


Android application

ရေးသားပန်တီးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ



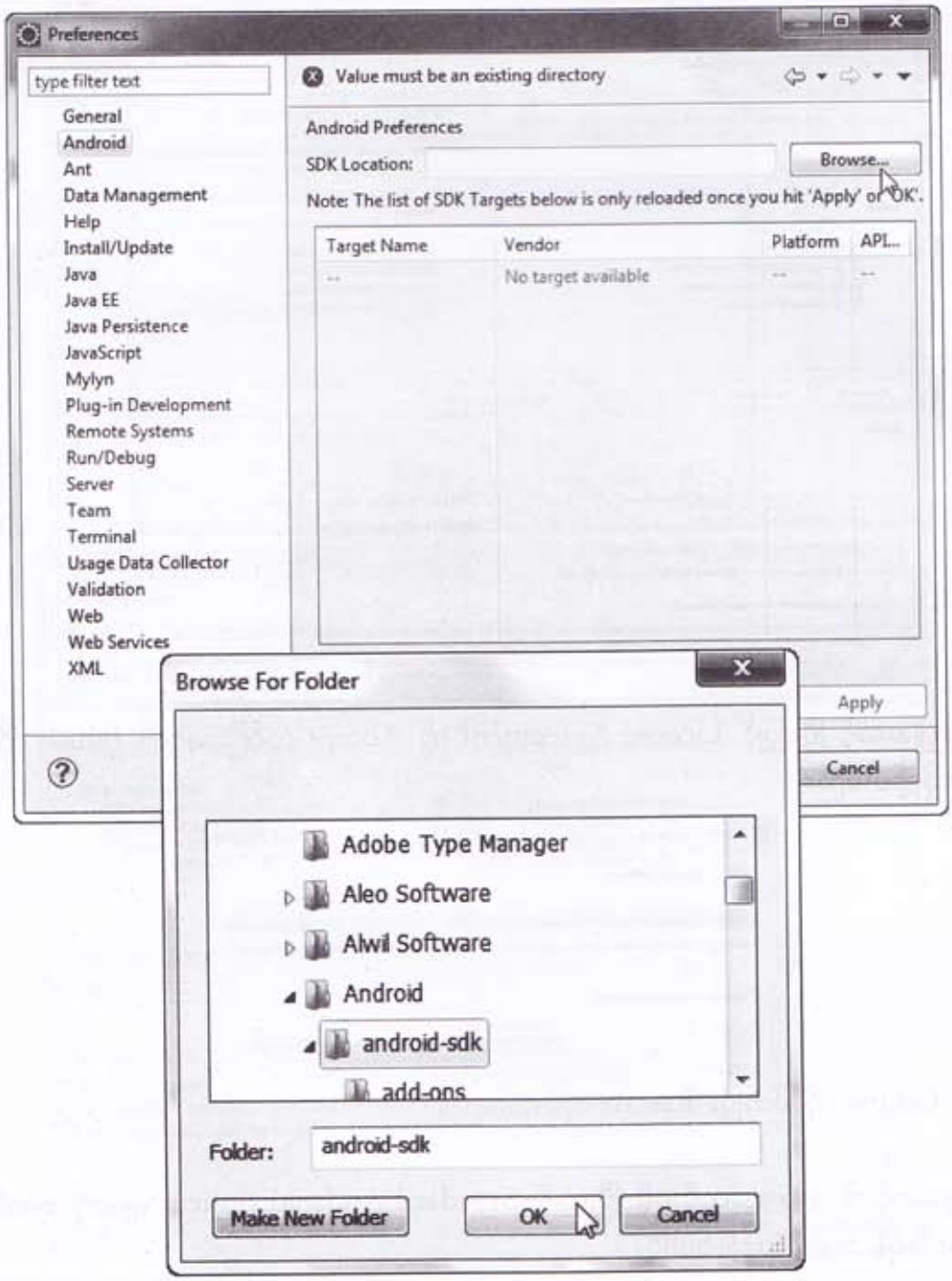
၆. ဤအဆင့် ပြီးလျှင် License Agreement ကို Accept လုပ်ပြီးနောက် Finish ကို နှိပ်လိုက်ပါမည်။



၇. Eclipse ကို ပိတ်ပြီး Restart လုပ်ရပါမည်။

ဤအဆင့်ထိ အမှားအယွင်းမရှိ ပြီးလျှင် Standard Android Project များကို စတင် ဖန်တီးရန် အဆင်သင့်ဖြစ်ပါပြီ။

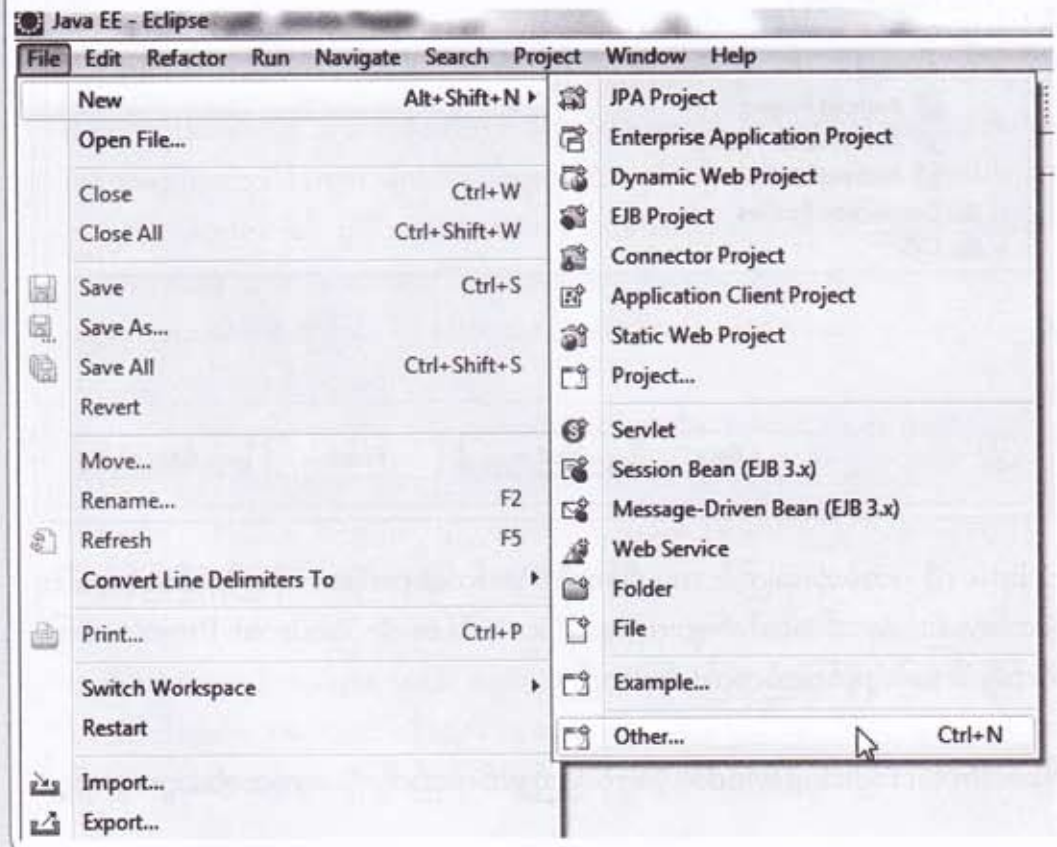
ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

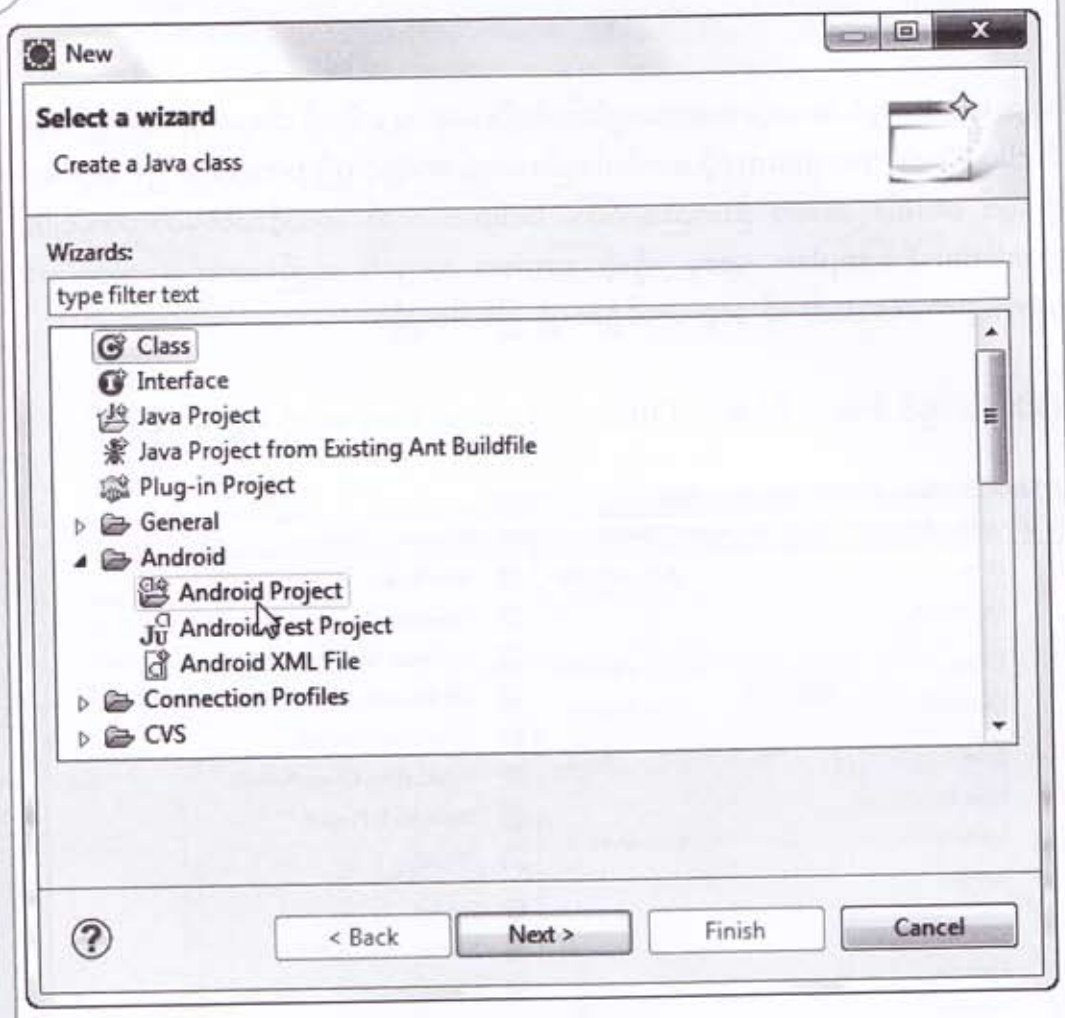

Preferences
type filter text
General
Android
Ant
Data Management
Help
Install/Update
Java
Java EE
Java Persistence
JavaScript
Mylyn
Plug-in Development
Remote Systems
Run/Debug
Server
Team
Terminal
Usage Data Collector
Validation
Web
Web Services
XML
Value must be an existing directory
Android Preferences
SDK Location:
Browse...
Note: The list of SDK Targets below is only reloaded once you hit 'Apply' or 'OK'.
Target Name
Vendor
Platform
API...
No target available
Apply
Cancel
Browse For Folder
Adobe Type Manager
Aleo Software
Alwil Software
Android
android-sdk
add-ons
Folder:
android-sdk
Make New Folder
OK
Cancel

၃.၄ Hello World သို့မဟုတ် ပထမဆုံး Android Project

ရှေ့အဆင့်များတွင် အားလုံး အမှားအယွင်းကင်းပြီး ချောမွေ့မှု ရှိမရှိ သိရစေရန် အရိုးရှင်းဆုံး Hello World (program) ကို စတင်ပါမည်။ အမှန်အားဖြင့် ဤ program တွင် ရေးစရာများစွာ မပါပါ။ အဓိက ဦးတည်ချက်မှာ Eclipse သည် အသင့်သတ်မှတ်ထားနှင့်ပြီး predefined template များမှ ကိုယ့် project အတွက် တည်ဆောက်ပေးသွားသော အကာ project shell ကို လေ့လာသိရှိစေရန် ဖြစ်ပါသည်။

Eclipse တွင် File > New > Other > Android Project ကို ရွေးပါမည်။





Eclipse ကို ပထမဆုံးအကြိမ် run ချိန်တွင် Android project ကို ဤနည်းဖြင့် ခေါ်ယူ ရသော်လည်း နောက်အခေါက်များတွင်မူ File > New > Android Project အဖြစ် တိုက်ရိုက် ခေါ်ယူနိုင်သည်လည်း ရှိပါသည်။

New Project (dialog window) တွင် ဖြည့်စွက်ရမည်တို့ကို ဆက်လက်ဆွေးနွေးရလျှင်

၁. Project Name သည် Eclipse ၏ တည်ဆောက်မှု Eclipse construct ဖြစ်သည်။ Eclipse သည် အရာအားလုံးကို project များအဖြစ် သီးခြားစီခွဲထုတ်စုစည်းထားသည်။ project name တစ်ခုသည် စကားလုံးတစ်လုံးသာ ဖြစ်သင့်သည်။ CamelCase အမည်ပေးစနစ်သည် အသင့်လျော်ဆုံး ဖြစ်သည်။ (CamelCase ဆိုသည်မှာ McDonald၊ iPod၊ McGuire စသည့် စကားလုံးများ ဖွဲ့စည်းထားပုံကဲ့သို့ စကားလုံး တစ်လုံးနှင့် တစ်လုံးကြား ကွက်လပ်မခြားဘဲ တစ်စုတည်းစကားလုံး တစ်လုံးအဖြစ် ဖွဲ့စည်းရေးသားသည့် ရေးထုံး ဖြစ်သည်။) ယခု project ကို HelloWorld ဟု အမည်ပေးပါမည်။

၂. ထို့နောက် *build target* ကို ရွေးရမည်။ build target က ဤ project သည် မည်သည့် Android platform version အတွက် တည်ဆောက်မှာလဲ ဆိုသည်ကို build tools များအား ပြောပြမည်။ Build Target (panel) ထဲတွင် SDK ၌ ကိုယ် ထပ်ဖြည့် install လုပ်ထားသည့် platform များနှင့် add-on များကို တန်းစီကာ ပြမည်။ နောက်ဆုံးအဖြစ်ဆုံး အသစ်ဆုံးကိုသာ ရွေးလိုက်ပါမည်။ (သို့သော် Google APIs ဟု အမည်ရှိသော target များကို မရွေးပါနှင့်။ ၄င်းတို့သည် Android platform အတွက် Google ၏ မူပိုင် proprietary extension များ ဖြစ်သည်။) Android platform အတွက် Android Open Source version များကိုသာ အသုံးပြုခြင်းက ရှေ့ကြောင်း နောက်ကြောင်း ရှင်းပါသည်။ (လက်ရှိအခြေအနေများအရ မူပိုင်ခွင့်ကိစ္စကို ထည့်သွင်းစဉ်းစားဖို့ လိုအပ်လာပေပြီ။)

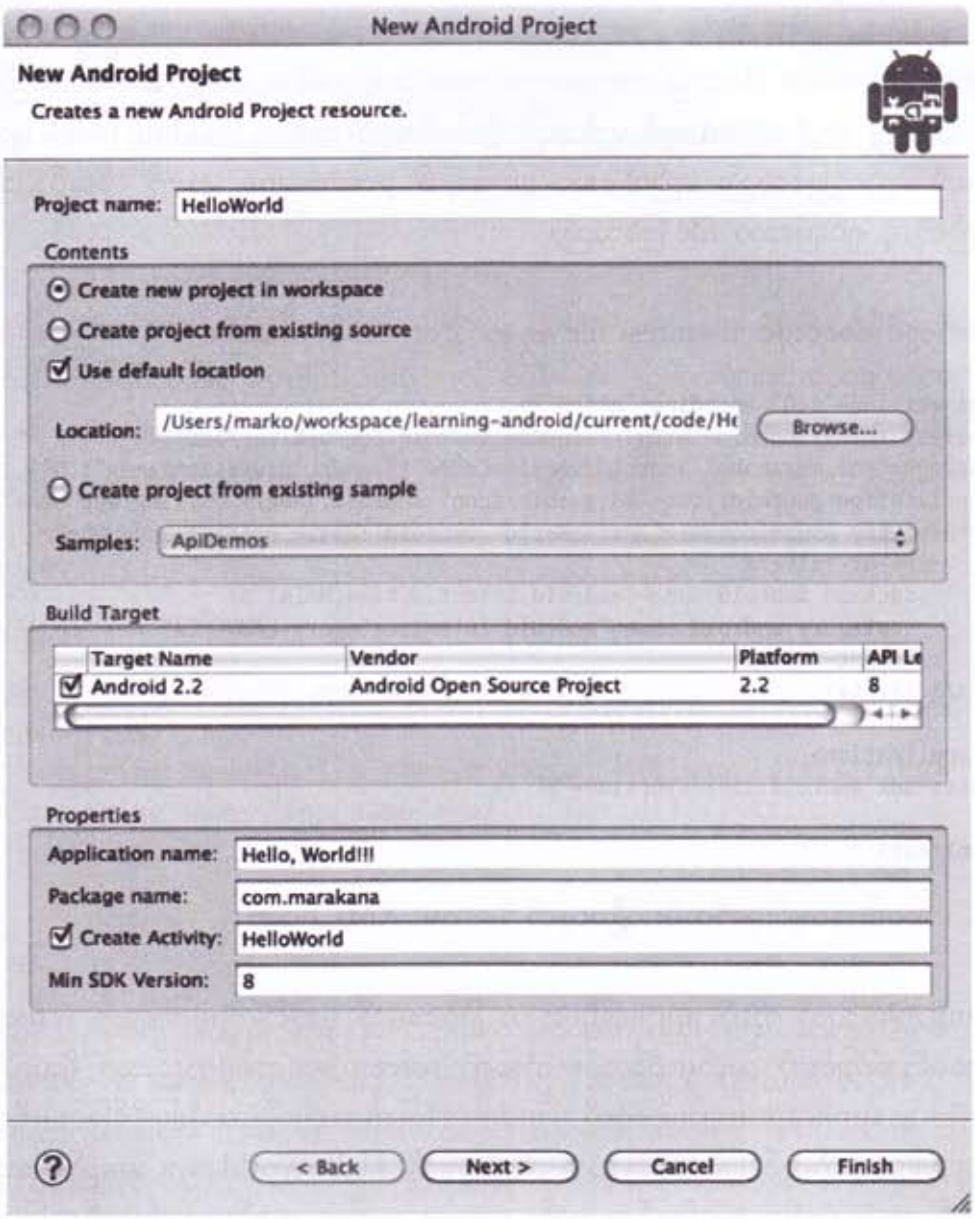
၃. ယခု project ၏ properties များကို ဖြည့်ရမည်။ application name သည် system နှင့် ဘာမျှမဆိုင်သော ဤ application ၏ အမည်နာမ သက်သက်သာ ဖြစ်ပါသည်။ Hello, World!!! ဟု ပေးလိုက်ပါမည်။ (မိမိစိတ်ကြိုက် စာလုံးများနှင့် အထားအသိုတို့ဖြင့် အမည်ပေးနိုင်ပါသည်။)

၄. Package Name သည် Java ၏ တည်ဆောက်မှု Java Construct တစ်ခု ဖြစ်ပါသည်။ Java တွင် source code အားလုံးကို package များအဖြစ် သီးခြားစီ ခွဲထုတ်စုစည်းပါသည်။ package တို့သည် အရေးပါ၏။ အကြောင်းမှာ ကိုယ့် project ထဲရှိ java class မျိုးစုံ အကြားဝယ် object များကို class အချင်းချင်း တွေ့ရစေရန်

သတ်မှတ်ပေးသောကြောင့် ဖြစ်သည်။ Android တွင်လည်း package များသည် application ကို sign လုပ်ရာတွင် အရေးပါသည်ပင်။ Android application များကို အင်တာနက်မှ ဖြန့်ချိသူများ၏ အမည်ပေးစနစ်မှာ packge name ကို ကိုယ်ပိုင် ဒိုမိန်းအမည် domain name ၏ ပြောင်းပြန် အမည်ပေးကြလေ့ရှိသည်။ ဥပမာ – example.com အင်တာနက်စာမျက်နှာကို ပိုင်သူ Android developer သည် သူ၏ package များကို com.example.calculator ၊ com.example.boardgame စသည်ဖြင့် အမည်ပေးလေ့ရှိသည်။ ဤစာအုပ်တွင် package name ကို com.marakana ဟု အမည်ပေးပါမည်။

၅. အနိမ့်ဆုံး minimum SDK version ဆိုသည်မှာ အနိမ့်ဆုံး minimum Android version ဖြစ်သည်။ API level ဖြင့် ကိုယ်စားပြုဖော်ပြလေ့ရှိသည်။ ၄င်းသည် ဤ application ကို run မည့် ကိရိယာတွင် လိုအပ်ချက် ဖြစ်သည်။ API level သည် နိမ့်နိုင်သမျှ နိမ့်ဖို့ လိုအပ်သည်။ သို့မှသာ ကိုယ့် application ကို ထည့်သွင်းနိုင်မည့် မိုဘိုင်းကြိုးမဲ့ကိရိယာ အရေအတွက် များမည် ဖြစ်သည်။

ယခုအဆင့်များအတိုင်း ဆိုလျှင် New Project (dialog window) သည် တစ်ဖက်စာမျက်နှာပါ ပုံအတိုင်း ဖြစ်နေပါမည်။


New Android Project
New Android Project
Creates a new Android Project resource.
Project name: HelloWorld
Contents
Create new project in workspace
Create project from existing source
Use default location
Location: /Users/marko/workspace/learning-android/current/code/He
Browse...
Create project from existing sample
Samples: ApiDemos
Build Target
Target Name
Vendor
Platform
API Le
Android 2.2
Android Open Source Project
2.2
8
Properties
Application name: Hello, World!!!
Package name: com.marakana
Create Activity: HelloWorld
Min SDK Version: 8
< Back
Next >
Cancel
Finish

၃.၅ စာရင်းဖိုင် သို့မဟုတ် manifest file

စာရင်းဖိုင် manifest file သည် component အားလုံးကို ပေါင်းစည်းချိတ်ဆက်ပေးသည်။ ၄င်းသည် ဤ application တွင် မည်သည်တို့ပါဝင်သည်၊ အဓိက building block များမှာ မည်သည်တို့ဖြစ်သည်၊ မည်သို့သော ခွင့်ပြုချက် permission များကို လိုအပ်သည် စသည်တို့ကို ရှင်းပြသော file ဖြစ်သည်။

အောက်တွင် စာရင်းဖိုင် manifest file နမူနာကို လေ့လာနိုင်ပါသည်။

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  package="com.marakana" android:versionCode="1" android:versionName="1.0">
  <application android:icon="@drawable/icon" android:label="@string/app_name">
    <activity android:name=".HelloWorld" android:label="@string/app_name">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>

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

</manifest>
```

၃.၆ အခင်းအကျင်းမှတ်တမ်း သို့မဟုတ် Layout XML code

Layout file သည် ကိုယ့် application က ဖြစ်စေမည့် မိုဘိုင်းကြိုးမဲ့ကိရိယာ screen ၏ အခင်းအကျင်းကို သတ်မှတ်သည်။ သုံးစရာ screen မှာ တစ်ခုတည်းသာ ရှိသည်။ (Desktop application များမှာကဲ့သို့ windows form အမျိုးမျိုး အသုံးပြုနိုင်ခြင်း မရှိ။) ထို screen ထက်တွင် ဖြစ်ပေါ်နေသော အရာများကို HelloWorld.java အမည်ရှိသော ပရိုဂရမ် code များက မောင်းနှင်နေသည်။ screen ထက်က အခင်းအကျင်းမှတ်တမ်းမှာ တစ်ဖက်စာမျက်နှာက code များကြောင့် ဖြစ်ပေါ်လာရပါသည်။

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

၃.၇ စာသားများ၏ဖိုင် သို့မဟုတ် Strings

ဤဖိုင်သည်ကား application တွင် ပါရှိသမျှ စာသား text အားလုံးကို သိမှီးထားသော XML ဖိုင်ဖြစ်သည်။ application တွင် ပါရှိသော စာသားများဆိုသည်မှာ button များ၏ အမည်များ၊ label များ၊ အလိုလျောက်အဆင်သင့်ရှိထားပြီးစာသား default text များနှင့် အလားတူ အခြားစာသားများဖြစ်သည်။ သဘောတရားမှာ layout XML သည် widget တို့၏ အခင်းအကျင်းများသက်သက်နှင့် ဆိုင်ပြီး၊ strings XML သည် widget တို့တွင် ပါဝင်သော စာသားများနှင့် သက်ဆိုင်ပေသည်။ နမူနာ စာသားဖိုင်ကို အောက်တွင် ကြည့်ပါ။

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

၃.၈ R ဖိုင်

ဤဖိုင်သည်ကား java လောကနှင့် resource လောကကို ပေါင်းကူးပေးသည့် ဖိုင်တည်း။ အလိုလျောက် ဖန်တီးသောဖိုင်ဖြစ်သည့်အတွက် ဤ R file ကို လုံးဝ မပြင်ရပါ။ ပုံတစ်ပုံ ထပ်တိုးလိုက်ခြင်း၊ XML ဖိုင်တစ်ခု ထပ်တိုးလိုက်ခြင်း စသဖြင့် res (folder) တွင်း၌ အပြောင်းအလဲ လုပ်လိုက်တိုင်းတွင် R file ကို အသစ် ပြန်လည်တည်ဆောက်သည်။ ဤ ဖိုင်နှင့် ပတ်သက်ပြီး ပြောရန်မရှိလှပါ။ ရံဖန်ရံခါ ဤ R file တွင်းမှ data ကို ယူသုံးပါ မည်။ သို့သော် Eclipse က အလိုလျောက် ကူညီသွားသည်သာ များပါမည်။

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

package com.marakana;

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

၃.၉ Java Source Code ဖိုင်

ဤ Java Source Code သည်ကား အရာရာကို မောင်းနှင်သည့်အရာ ဖြစ်ပေသည်။ ၄င်းအား နောက်ဆုံးတွင် Dalvik Executable အဖြစ် ပြောင်းလဲကာ application ကို run ၏။ အောက်ပါအတိုင်း ဖြစ်ပါသည်။

```
package com.marakana;

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

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

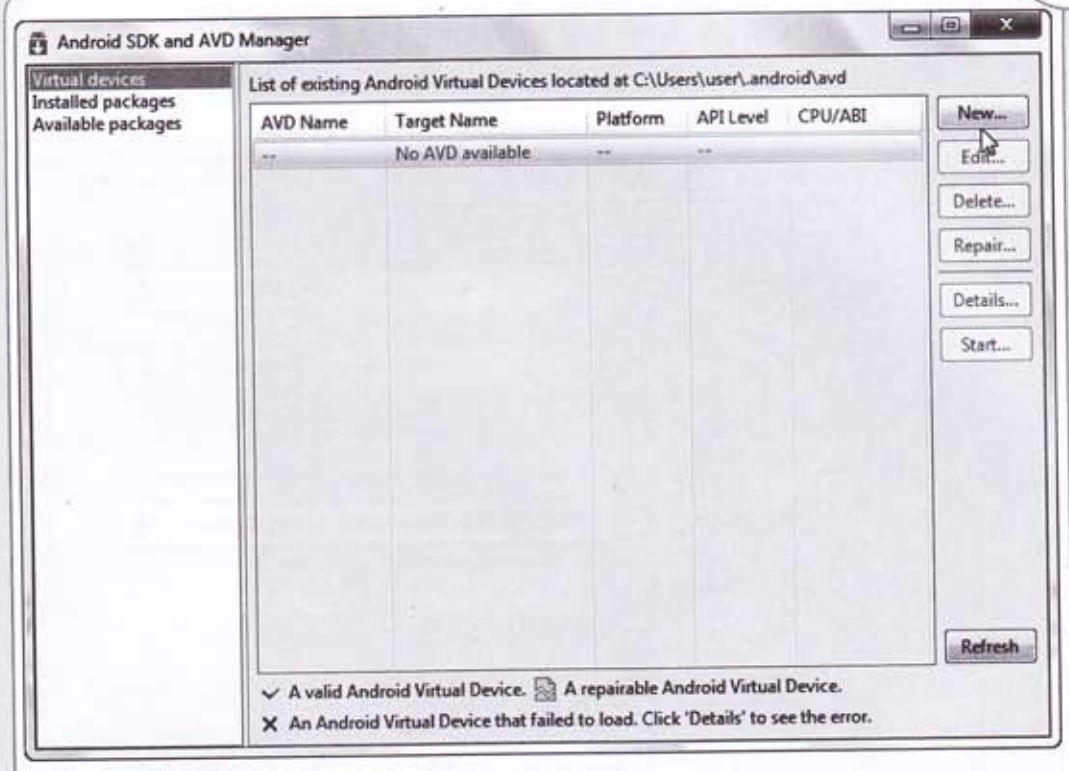
၃.၁၀ Emulator

မိမိဖန်တီးသော ဤ application ကို ကြိုးမဲ့မိုဘိုင်းကိရိယာ တစ်ခုပေါ်တွင် တကယ် run သည်နှင့် Emulator ပေါ်တွင် run သည်မှာ မည်သို့မျှ မထူးခြားပါ။ အကြောင်းမူ Emulator ဆိုသည်မှာ ကြိုးမဲ့မိုဘိုင်းကိရိယာအဖြစ် အလုပ်လုပ်ပေးသော ကိရိယာယောင် ဖြစ်သောကြောင့်ပင်။ ကိရိယာယောင် ဆိုသော်ငြား ကိရိယာအစစ်ပေါ်တွင် machine အလွှာ တိုင်သည်အထိ run သော code များကိုပင် run ပေးသည်သာဖြစ်သည်။

ဤတွင် ဆွေးနွေးရန် တစ်ချက် ရှိသည်မှာ simulator နှင့် emulator သည် အခေါ်အဝေါ် အားဖြင့် နီးစပ်သော်လည်း အလုပ်လုပ်ပုံခြင်းမှာ အခြေခံမှစ၍ လုံးဝကွာခြားပါသည်။ emulate လုပ်သည် ဆိုသည်မှာ binary code ကို execute လုပ်ပေးခြင်း ဖြစ်၍ emulator သည် virtual machine / virtual device ပင် ဖြစ်သည်။ simulator ဆိုသည်မှာကား binary code ထိအောင် မဆင်းဘဲ higher level မှာတွင်သာလျှင် code တစ်ခု၏ လုပ်ဆောင်ချက်ကို တုပဖန်တီးပေးခြင်း ဖြစ်သည်။

Emulator ကို အသုံးပြုဖို့ရာအတွက် Android Virtual Device (AVD) တစ်ခုကို ဖန်တီးရပေမည်။ AVD ကို ဖန်တီးရန် အလွယ်ဆုံးနည်းမှာ Eclipse မှတဆင့် android (tool) ကို start လုပ်ခြင်း ဖြစ်သည်။

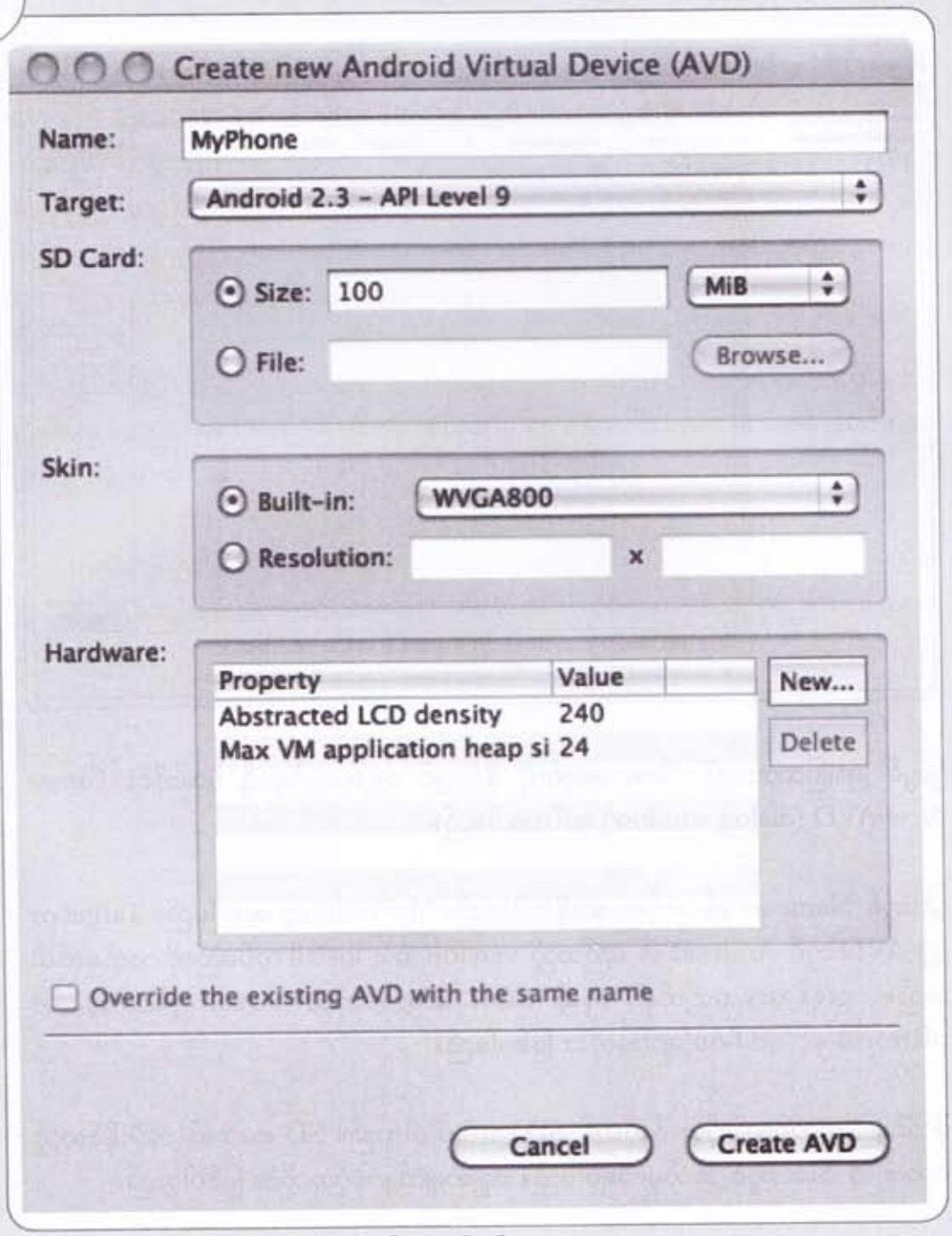
AVD သစ်တစ်ခုကို ဖန်တီးရန်အတွက် Eclipse ပေါ်မှ ပုံ ခလုပ်ကို နှိပ်လိုက်ပါမည်။ ထိုအခါ ရှေ့စာမျက်နှာမှပုံအတိုင်း ပေါ်လာပါမည်။



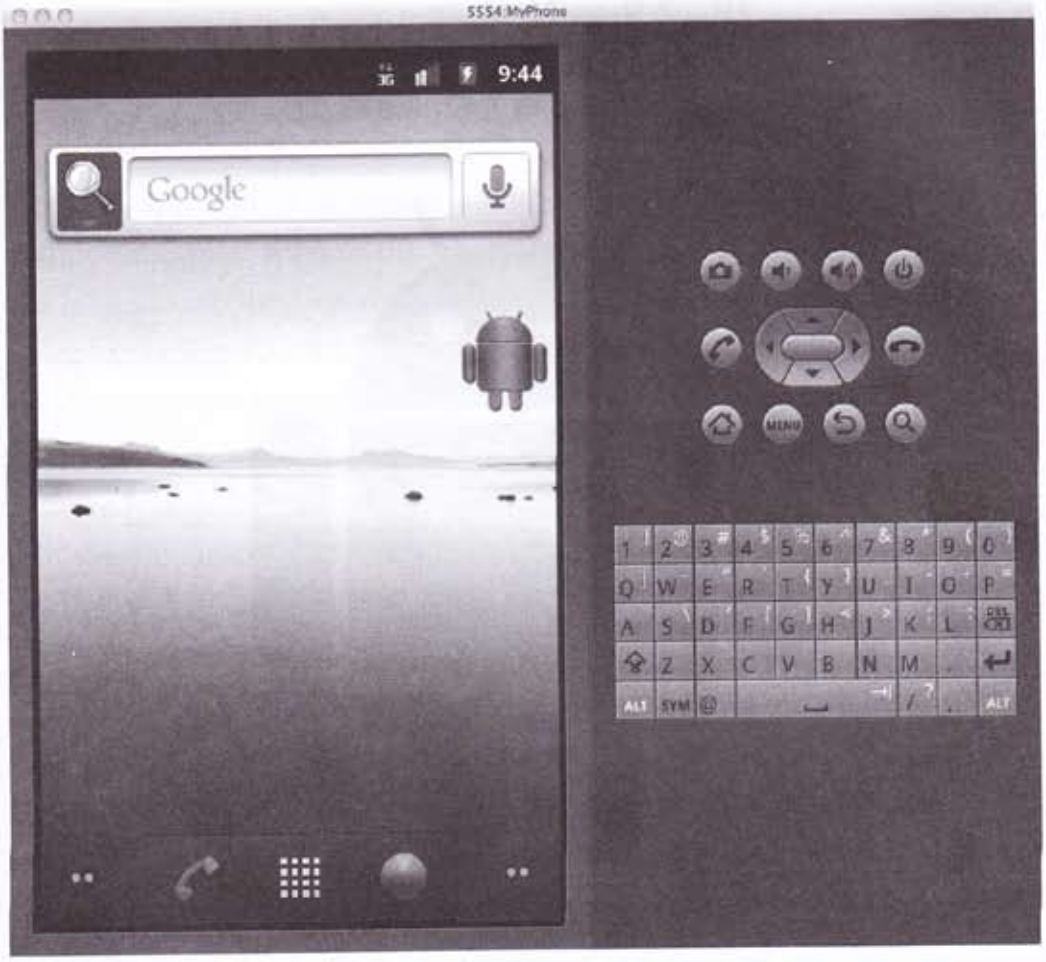
ပုံတွင် မြားပြထားသည့် New ခလုပ်ကို နှိပ်လျှင် ရှေ့စာမျက်နှာရှိ ပုံအတိုင်း Create New AVD (dialog window) ပေါ်လာပါမည်။

ဌင်းတွင် Name မှာ နှစ်သက်ရာ ပေးနိုင်ပါသည်။ MyPhone ဟု ပေးပါမည်။ Target ကဏ္ဍ AVD တွင် Android ၏ မည်သည့် version အား install လုပ်မည်ကို သတ်မှတ်ပါသည်။ ရွေးရန်အရေအတွက်မှာ ကိုယ့် SDK ထဲတွင် ကိုယ် install လုပ်ထားခဲ့သော platform နှင့် add-on များအတိုင်း ဖြစ်ပါမည်။

မိုဘိုင်းကြိုးမဲ့ကိရိယာတိုင်းလိုလိုတွင် SD card ပါပါသည်။ SD card ၏ သိုမှီးနိုင်သည့် ပမာဏကို Size တွင် သတ်မှတ်ရပါမည်။ ရှေ့စာမျက်နှာမှပုံအတိုင်း ဖြစ်ပါသည်။


Create new Android Virtual Device (AVD)
Name:
MyPhone
Target:
Android 2.3 - API Level 9
SD Card:
Size:
100
MiB
File:
Browse...
Skin:
Built-in:
WVGA800
Resolution:
x
Hardware:
Property
Value
Abstracted LCD density
240
Max VM application heap si
24
New...
Delete
Override the existing AVD with the same name
Cancel
Create AVD

ရှေ့တွင် ဆွေးနွေးခဲ့သောအဆင့်များ အားလုံးပြီး၍ Create AVD ခလုပ်ကို နှိပ်လိုက်သည် နှင့် အောက်ပါ အတိုင်း Emulator ပေါ်လာပါမည်။



မိမိဖန်တီးသော application ၌ AVD ပေါ်တွင် ကြိုးမဲ့မိုဘိုင်းကိရိယာ အစစ်နှင့် မတူနိုင်သော အချက်မှာ တစ်ချက်တည်းသာ ရှိသည်။ ယင်းမှာ Sensor များ ဖြစ်သည်။ ကျန်သော feature အားလုံးနီးပါးမှာ ကြိုးမဲ့မိုဘိုင်းကိရိယာ အစစ်အတိုင်းသာ ဖြစ်ပါသည်။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Android application

ရေးသားပန်တီးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

အခန်း – ၄
Android ၏ ပင်မအစိတ်အပိုင်းများ

Android application များကိုတည်ဆောက်ရာတွင် ပင်မအစိတ်အပိုင်းများ ဖြစ်သည့် component များမှာ application တစ်ခုလုံးကြီးကို ဆက်စပ်တည်ဆောက်ယူရမည့် item များ ဖြစ်သည်။ application တစ်ခုကို စတင်စဉ်းစားလျှင် အထက်မှ ဖြာဆင်းသွားသည့် ချဉ်းကပ်နည်း Top-Down approach ကို အသုံးပြုသင့်သည်။ application ကို design ချလျှင် screen များ၊ feature များနှင့် ၄င်းတို့အချင်းချင်း အပြန်အလှန်အလုပ်လုပ်ပုံများကို စဉ်းစားရပေမည်။ သဘောတရားကို စက်ဝိုင်းများ လေးထောင့်များဖြင့် ပုံဆွဲကာ အစပြုသင့်ပေသည်။ ဤနည်းသည် component များအချင်းချင်း မည်သို့မည်ပုံ အံဝင်ခွင်ကျဖြစ်စေမည်၊ လိုချင်သည့်အလုပ်ကို မည်သို့မည်ပုံ ပြီးမြောက်စေမည်ကို ရှင်းလင်းစွာ မြင်သာစေ၏။

၄.၁ စဉ်းစားနည်း အဆင့်ဆင့် လက်တွေ့ဥပမာ

Twitter ကဲ့သို့သော application တစ်ခုကို တည်ဆောက်လိုသည် ဆိုပါစို့။ user သည် status update များကို post လုပ်နိုင်ရန်လိုသည်။ ထိုနည်းတူ user သည် သူ၏ friend များ တင်သော post များကို သိနိုင်ရန်လိုသည်။ အထက်ပါသဘောတရားနှစ်ခုသည် twitter app တစ်ခုအတွက် အခြေခံအကျဆုံး လုပ်ဆောင်ချက်များ ဖြစ်သည်။ ထိုထက်ပိုလျှင် user သည် မိမိ၏ twitter account ထဲသို့ log in ဝင်ရန် username နှင့် pass word ကို သတ်မှတ်နိုင်ရန်လိုသည်။ ထို့ကြောင့် အထက်ပါ လုပ်ဆောင်ချက် သုံးခုအတွက် screen သုံးခုကို လိုအပ်သည်။

နောက်ထပ် အဆင့်မြှင့်လျှင် ဤ application ကို ကွန်ယက်ဆက်သွယ်အား network

connectivity နေ့သည့်အခြေအနေ၌ပင် အလုပ်လုပ်စေလိုသည်။ ဤလိုအပ်မှုကို အဆင်ပြေစေရန်အတွက် လုပ်နိုင်သည်မှာ ကြိုးမဲ့မိုဘိုင်းကိရိယာ online ဖြစ်နေချိန်တွင် twitter မှ data များကို ကိရိယာတွင်းသိုမှီးမှ local cache ထဲသို့ ဆွဲချထားရမည်။ဤကိစ္စသည် နောက်ကွယ်တွင် အလုပ်လုပ်နေရမည့် ဝန်ဆောင်မှု background service တစ်ခုနှင့် database တစ်ခုကို လိုအပ်စေပေမည်။

ဤ background service ကို ကြိုးမဲ့မိုဘိုင်းကိရိယာ ပါဝါဖွင့်လိုက်သည်နှင့် စ run စေရမည်။ သို့မှသာလျှင် user သည် ယခု application ကို စဖွင့်လိုက်သည်နှင့် သူ့ friends တို့၏ status update များကို တန်းပြီး အချိန်နှင့် တပြေးညီ တွေ့ရပေမည်။

အထက်တွင် ဆွေးနွေးခဲ့သော အချက်များသည် ဥပမာထားသည့် application အတွက် ရိုးစင်းသော အခြေခံလိုအပ်ချက်များ ဖြစ်သည်။ Android ၏ component များသည် ဤလိုအပ်ချက်များကို အကောင်အထည်ဖော်ရာတွင် တစ်ခုချင်း သပ်သပ်စီ လုပ်နိုင်ရန်အတွက် လည်းကောင်း၊ တစ်ခုချင်းသပ်သပ်စီ ဖန်တီးထားသည့် အိုင်းများကို ပြည့်စုံသော package တစ်ခုအဖြစ် ပြန်လည်ပေါင်းစည်းနိုင်ရန်အတွက် လည်းကောင်း အဆင်ပြေလွယ်ကူစေပါသည်။

၄.၂ Activities များ

Activity ဆိုသည်မှာ အချိန်တစ်ချိန်တွင် မိုဘိုင်းကြိုးမဲ့ကိရိယာပေါ်၌ user မြင်ရသည့် screen တစ်ခု ဖြစ်လေ့ရှိသည်။ Application တစ်ခုတွင် activity များစွာ ရှိနိုင်ပြီး user သည် ၄င်းတို့ကို အပြန်အလှန် ကူးချည်သန်းချည် အလုပ်လုပ်လေ့ ရှိသည်။

Activity ကို မြင်သာစေရန်အတွက် website ဖြင့် ဥပမာ ပြုနိုင်သည်။ website တစ်ခုတွင်လည်း web page များစွာ ပါရှိပြီး၊ user သည် ထို web page များကို ကူးချည်သန်းချည် သွားလေ့ရှိသကဲ့သို့ပင် Android application များတွင်လည်း activity

များစွာ ရှိသည်။ website တစ်ခုတွင် home page ရှိသကဲ့သို့ပင် Android app တစ်ခုတွင်လည်း main activity ရှိသည်။ main activity သည် application ကို စတင် launch လုပ်လိုက်လျှင် ပေါ်လာလေ့ ရှိသည်။ ထိုမျှမက website တွင် web page များ အချင်းချင်း ဟိုမှသည်မှ ညွှန်းတတ်သကဲ့သို့ပင် Android application တွင်လည်း activity တစ်ခုမှတစ်ခု ဟိုမှသည်မှ ညွှန်းတတ်လေသည်။ နောက်ဆုံးအနေနှင့်ကား web site တစ်ခုသည် အခြား website တစ်ခုကို လှမ်းကာ ခေါ်ယူတတ်သကဲ့သို့ပင် Android application တစ်ခုသည်လည်း အခြား application တစ်ခုကို ခေါ်ယူတတ်လေသည်။ ဥပမာ ဆိုရလျှင် Contacts (application) မှ contact တစ်ခုကို ကြည့်ရင်း Messaging (application) ကို ဆက်ပြီး ခေါ်ယူသကဲ့သို့။

၄.၃ Activity Life Cycle

Activity တစ်ခုကို မိုဘိုင်းကြိုးမဲ့ဆက်သွယ်ရေးကိရိယာတွင် launch လုပ်ရခြင်းသည် Linux process တစ်ခုကို ဖန်တီးရခြင်း၊ သုံးသူမြင်ကွင်းရှိ အရာဝတ္ထု User Interface object များအတွက် memory ကို ဖွဲ့ယူသုံးစွဲ အကန့်ခွဲရခြင်း၊ screen တစ်ခုလုံးအား set up လုပ်ရခြင်း စသဖြင့် ကရိကထကြီးလုပေသည်။ ထိုမျှ ကရိကထခံကာ ဖန်တီးထားသော activity သည် ခဏကြည့်စရာမျှသာ ဖြစ်သွားပြီး user က ထပ်၍ ထပ်၍ နောက် activity များကို ဆက်လက်ဖန်တီးသွားမည်ဆိုလျှင် မိုဘိုင်း၏ စွမ်းနိုင်အားနှင့်မမျှ ဖြစ်ပေတော့မည်။ ထိုသို့ဖြစ်သောကြောင့် activity life cycle ကို Activity Manager ဖြင့် ထိန်းချုပ်ရ၏။

Activity Manager သည် activity များကို ဖန်တီးခြင်း၊ ဖျက်သိမ်းခြင်းနှင့် ထိန်းချုပ်ခြင်းများကို လုပ်ဆောင်သည်။ ဥပမာ – user က application တစ်ခုကို ပထမဆုံးအကြိမ် အဖြစ် စတင် launch လုပ်လိုက်လျှင် Activity Manager သည် ၄င်း၏ activity ကို ဖန်တီးလိုက်ပြီး screen ပေါ်သို့ တင်လိုက်သည်။ user က screen တစ်ခုသို့ကူးလျှင် Activity Manager က ယခင် activity ကို ယာယီမိုးသိုရာ holding place ပေါ်သို့ တင်လိုက်သည်။

ဤနည်းအားဖြင့် user သည် ယခင် activity သို့ ပြန်သွားပါက ပိုမိုလျှင်မြန်စွာ launch လုပ်ပေးနိုင်သည်။ အချိန် အတော်ကြာသည်အထိ အသုံးမပြုဖြစ်သော activity ဟောင်းများ ကိုမူ destroy လုပ်ပစ်သည်။ သို့မှသာ လက်ရှိ activity အတွက် memory ထွက်လာမည်။ ဤ ယန္တရားစနစ်သည် user interface ၏ မြန်နှုန်း speed ကို ပိုမိုတိုးတက်လာစေသည်။

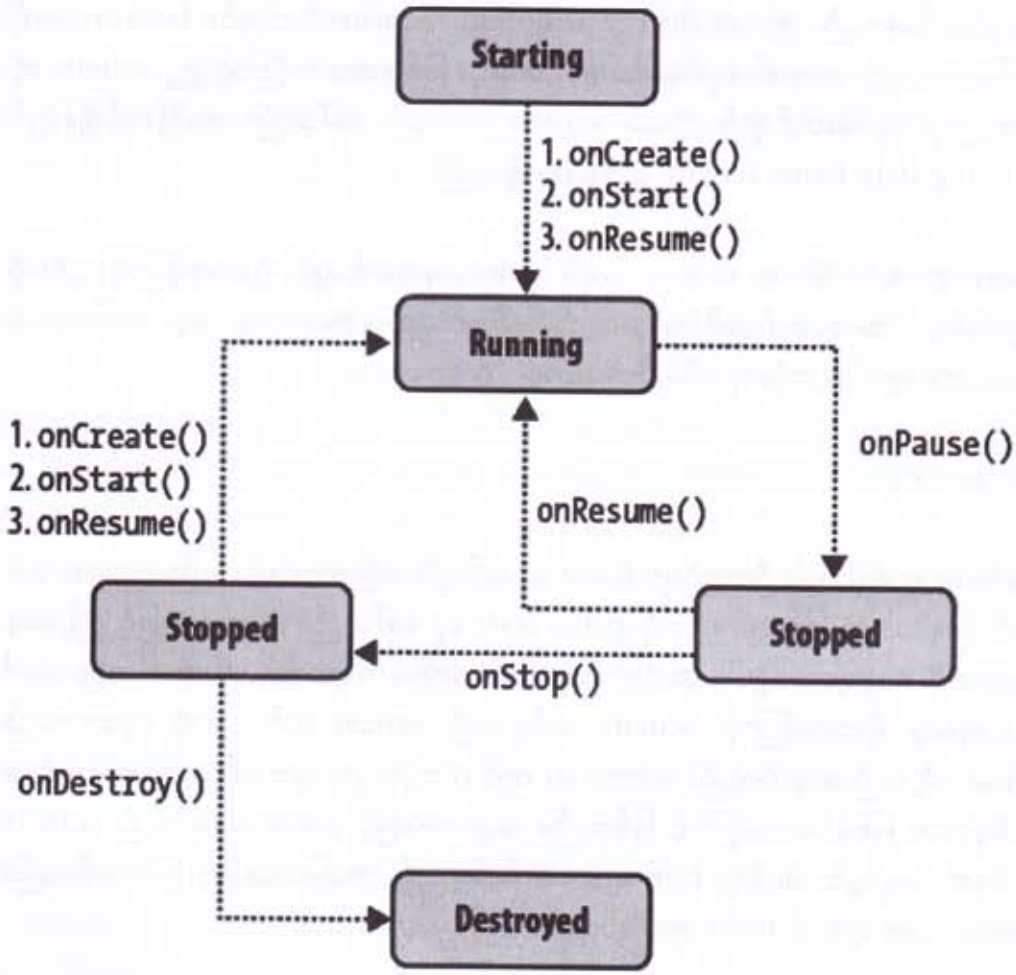
Android အတွက် ပရိုဂရမ်ရေးသားခြင်းသည် အခြားလုပ်ငန်းခွင်များအတွက် ပရိုဂရမ်ရေးသားခြင်းနှင့် သဘောတရားချင်း မတူလှချေ။ Android တွင် application ၏ ကူးပြောင်းမှု လားရာကို ကိုယ်က လမ်းကြောင်းချပေးသည်ထက် ပြောင်းလဲသွားသည်များကို တုံ့ပြန်ပေးရခြင်းမျိုးကို ပိုမိုကြိုတွေ့ရသည်။

Activity တစ်ခု ဖြတ်သန်းသွားသော state များကို တစ်ဖက်စာမျက်နှာရှိ သရုပ်ပြပုံတွင် တွေ့ရှိနိုင် သည်။

Starting state

Activity တစ်ခုသည် memory ပေါ်တွင် မရှိသေးလျှင် ၄င်းသည် starting state ဖြစ်သည်။ ၄င်း အစပြုနေချိန်တွင် ခေါ်ယူစနစ် callback method များကို အစုလိုက် run ပါမည်။ developer တစ်ဦးအနေဖြင့် ထိုအခြေအနေတွင် callback method များကို ဆင်ထားရန် ကြိုတင်စီစဉ်နိုင်ခွင့် အပြည့်အဝ ရရှိမည်။ ဤအဆင့်ပြီးက running state သို့ ရောက်ပေမည်။

startingstateမှ runningstateသို့အကူးသည်တွက်ချက်ချိန် computingtimeရှုထောင့်မှ ကြည့်လျှင် အလွန်တန်ကြေးကြီးသည်။ ဆိုလိုသည်မှာ တွက်ချက်စွမ်းအား computing power ကို အထူးစားသုံးသည်။ အကျိုးဆက်အားဖြင့် မိုဘိုင်းဘက်ထရီကိုလည်း အထူးဆွဲငင်အသုံးပြုသည်။ နောက်ကွယ်သို့ရောက်သော activity များကို ချက်ချင်း destroy မလုပ်ပစ်ခြင်းမှာ အဓိက ဤအချက်ကြောင့်ပင်။ user က ပြန်သုံးကောင်း သုံးပေဦးမည်။


Starting
1. onCreate()
2. onStart()
3. onResume()
Running
onPause()
onResume()
1. onCreate()
2. onStart()
3. onResume()
Stopped
Stopped
onStop()
onDestroy()
Destroyed

Running state

running state ရှိ activity မှာ လက်ရှိ screen ပေါ်တွင် ရှိနေပြီး user နှင့် အပြန်အလှန် အလုပ်လုပ်နေသည်။ ဤ activity တွင် စာရိုက်ခြင်း၊ screen ကို ထိခြင်း၊ button များကို နှိပ်ခြင်း စသည့် user ၏ လုပ်ဆောင်ချက်အားလုံး ဖြစ်ပွားသောကြောင့် ဤ activity ကို ဦးတည်ရာ လုပ်ဆောင်ချက် focus activity ဟုလည်း ခေါ်သည်။ အချိန်တစ်ချိန်တွင် running state focus activity တစ်ခုသာ ရှိသည်။

running state focus activity သည် ဦးစားပေးအဆင့်တွင် ရှိသောကြောင့် ၄င်းကို လျှင်မြန်စွာ အလုပ်လုပ်စေနိုင်ရန်အတွက် လိုအပ်သည့် memory နှင့် အခြားသော resource များ အားလုံးကို သိမ်းပိုက်သည်။

Pause state

Activity တစ်ခုသည် ဦးတည်ရာ focus အဆင့်တွင် မရှိတော့သော်လည်း screen ပေါ်တွင် ရှိနေသေးဆဲ ဖြစ်လျှင် ၄င်းကို pause state ဟု ခေါ်သည်။ အများအားဖြင့် ဤအနေအထားကို တွေ့ရလေ့မရှိပါ။ အကြောင်းမူ မိုဘိုင်းကြိုးမဲ့ကိရိယာ၏ screen သည် သေးငယ်နေလေ့ ရှိသောကြောင့် activity တစ်ခုသည် screen တစ်ခုလုံးကို ယူထားသည် ဖြစ်စေ၊ သို့တည်းမဟုတ်လည်း screen ပေါ်တွင် လုံးလုံးလျားလျား မရှိသည်လည်း ဖြစ်စေ တစ်ခုခုသာ ဖြစ်နိုင်သောကြောင့် ဖြစ်သည်။ အများအားဖြင့် pause state သည် activity တစ်ခု၏ ရှေ့တွင် dialog box များ တက်လာသည့် အခြေအနေမျိုး ဖြစ်တတ်သည်။ pause state တွင် activity များ ရပ်သွားသည်။

pause state တွင် ရှိနေသော activity များသည်လည်း memory နှင့် အခြားသော resource များအားလုံး အပေါ်တွင် ဦးစားပေး အနေအထား၌ ရှိဆဲဖြစ်သည်။ အကြောင်းမူ ၄င်းတို့ကို ဖြုတ်ချ ဖျက်သိမ်းပစ်လိုက်၍ မရသေးသောကြောင့် ဖြစ်သည်။

Stopped state

Activity တစ်ခုသည် မြင်ကွင်းတွင် မရှိတော့ပဲ memory ပေါ်တွင်သာ ရှိနေတော့လျှင် ၄င်းကို stopped state ဟု ခေါ်သည်။ stopped state သို့ ရောက်သွားသော activity တစ်ခုသည် လက်ရှိအလုပ်လုပ်ဆဲ running activity တစ်ခုက ခေါ်ယူမှုကြောင့် မြင်ကွင်းပေါ်သို့ ပြန်လည်ရောက်ရှိလာနိုင်ပါသေးသည်။ ထိုသို့မဟုတ်ခဲ့လျှင် ၄င်းကို ဖျက်သိမ်းပြီး memory ပေါ်မှ ဖယ်ရှားပစ်လိုက်နိုင်ပြီး ဖြစ်ပေမည်။

Android ကြိုးမဲ့မိုဘိုင်းကိရိယာ၏ စနစ် system သည် activity များကို stopped state တွင် ထား ထားလေ့ရှိသည်။ မကြာခင် တချိန်ချိန်တွင် user သည် ထို activity များကို ပြန်သုံးနိုင်လောက်သည် ဟု ယူဆကာ ထိုသို့ ဖန်တီးထားခြင်း ဖြစ်သည်။ (Android တွင် application များအတွက် close button အလိုလျောက် မပါပါ။) stopped state ရှိ activity တစ်ခုကို ပြန်လည်ကာ restart လုပ်ခြင်းသည် memory ပေါ်တွင် object အားလုံးကို load လုပ်ထားနှင့်ပြီး ဖြစ်သောကြောင့် ရှေ့သို့ ဆွဲတင်လိုက်ရုံမျှသာ ဖြစ်သဖြင့်၊ ယင်း activity ကို အသစ် စတင်ဖန်တီးခြင်းထက် များစွာ resource သက်သာပေသည်။

Stopped activity များကို memory မှ အချိန်မရွေး ဖြုတ်ချဖျက်သိမ်းလိုက်နိုင်ပါသည်။

Destroyed state

Destroy လုပ်လိုက်ပြီးသော activity သည် memory ပေါ်တွင် မရှိနေတော့ပါ။ Activity Manager က ဤ activity သည် မလိုအပ်တော့ ဟု ဆုံးဖြတ်လိုက်ပြီး memory ပေါ်မှ ဖျက်သိမ်းလိုက်လေပြီ။ activity တစ်ခုကို destroy မလုပ်မီတွင် (save မလုပ်ရသေးသော data များကို save လုပ်ခြင်းကဲ့သို့) လုပ်ဆောင်ချက် အချို့ကို လုပ်နိုင်ပေသည်။ သို့သော် pause state တွင်ရှိသော activity တစ်ခုသည်လည်း destroy လုပ်ခြင်း ခံရနိုင်ပေသည်။ သို့ဖြစ်သောကြောင့် save လုပ်ခြင်းကဲ့သို့ အရေးကြီးသည့် လုပ်ဆောင်ချက်မျိုးကို destroy

state၏ ရှေ့တွင် လုပ်ဆောင်သည်ထက် paused state၏ ရှေ့တွင် လုပ်ဆောင်သင့်ပါသည်။

သိသင့်သော အချက်တစ်ချက်မှာ – activity တစ်ခုသည် running state တွင် ရှိနေသည် ဆိုခြင်းမှာ အသည်းအသန် အလုပ်ရှုပ်နေခြင်းမျိုး ဖြစ်ရမည်ဟု ပုံသေ အဓိပ္ပာယ်ကောက်၍ မရပါ။ running state တွင် ရှိနေသည် မှန်သော်လည်း user ၏ input ကို စောင့်နေခြင်း မျိုးလည်း ဖြစ်နိုင်ပါသည်။ အလားတူပင် activity တစ်ခုသည် stopped state တွင် ရှိနေသည် ဆိုခြင်းကို ဘာအလုပ်မျှ မလုပ်တော့ ဟု ပုံသေ သတ်မှတ်၍ မရပါ။ ဤ အဆင့်များကို သတ်မှတ်မှုမှာ activity တစ်ခုသည် ဦးတည်ရာ focus အဆင့်တွင် ရှိနေခြင်း မရှိနေခြင်း၊ မြင်ကွင်းတွင် ရှိနေခြင်း မရှိနေခြင်းတို့ ပေါ်တွင်သာ အခြေခံခြင်း ဖြစ်သည်။

၄.၄ Intent များ

Intent ဆိုသည်မှာ Android platform ၏ ပင်မအခြေခံအစိတ်အပိုင်း building block များ အတွင်း အချင်းချင်း ပေးပို့နေသော message များ ဖြစ်သည်။ ၄င်းတို့သည် activity တစ်ခုကို စတင်စေရန် ခလုပ်ဖြုတ်သည်၊ service တစ်ခုကို စတင်ရန် သို့မဟုတ် အဆုံးသတ်ရန် ညွှန်ကြားသည်၊ သို့တည်းမဟုတ် ထုတ်လွှင့်မှုသက်သက်များလည်း ဖြစ်နိုင်သည်။

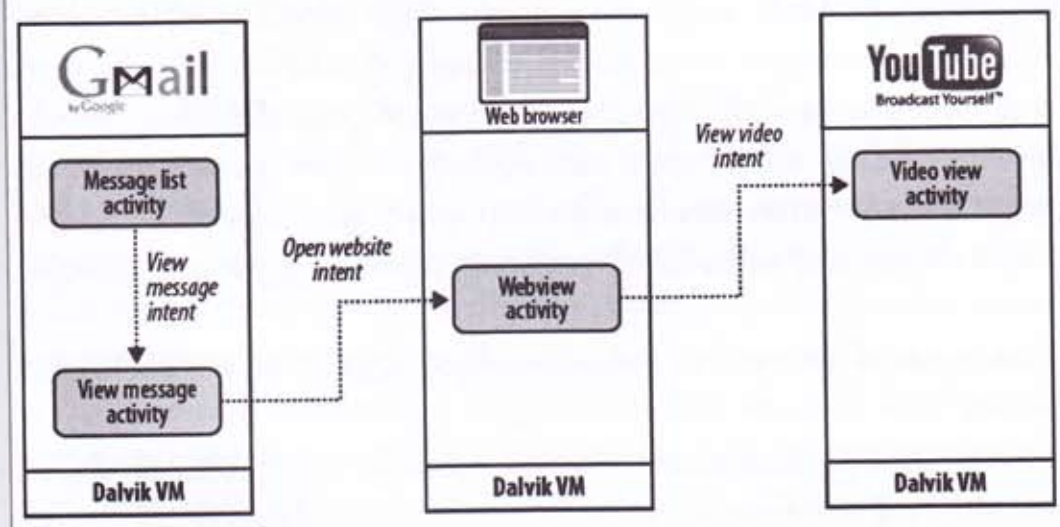
Intent များသည် asynchronous သဘာဝ ရှိသည်။ အဓိပ္ပာယ်မှာ intent ကို ထုတ်လွှင့်သော code သည် intent ဆုံးသည် ထိ စောင့်ရန် မလိုပါ။

Intent တစ်ခုသည် သီးခြားရှင်းလင်းသည် explicit လည်း ဖြစ်နိုင်သည်၊ ရောယှက်ရှုပ်ထွေးသည် implicit လည်း ဖြစ်နိုင်သည်။ explicit intent မျိုးတွင် ပို့လွှတ်သော sender code သည် လက်ခံဘက်ခြမ်း receiving end တွင် မည်သည့် component ဖြစ် / ရှိ မည်ကို အတိအကျ ဖော်ပြသည်။ ဥပမာ ပြရလျှင် ကိုယ့် application မှ activity သည် user အား web page တစ်ခုကို ဖွင့်စေသည့် intent ကို ပို့သည် ဆိုပါစို့။ ဤသို့ဆိုလျှင်

ထို web page ကို ဖွင့်ရာတွင် အတူပူးတွဲ အလုပ်လုပ်ရမည့် application တစ်ခုခု သည် ဤလုပ်ဆောင်ချက် ပြီးမြောက်ရေးတွင် အပြိုင်ဝင်ရောက်လာပေမည်။ (web page ကို ဖွင့်ရန် Browser application ပါ memory ပေါ်သို့ တက်လာမည်။)

ကြိုးမဲ့မိုဘိုင်း ကိရိယာထဲတွင် Browser application တစ်ခုထက်မက ရှိနိုင်ပေသည်။ ဤသို့ application များ အပြိုင်ရှိလာလျှင် မူလပေးပို့သည် intent ၏ စည်းမျဉ်းမှုကို လုပ်ဆောင်ရန် မည်သည့် application ကို အသုံးပြုမည်နည်းဟု system က မေးလာမည်။ app တစ်ခုကို default အဖြစ်ထားခြင်းဖြင့်လည်း ဤကိစ္စကို ဖြေရှင်းနိုင်သည်။ desktop computer များတွင်လည်း ဤဖြစ်ရပ်မျိုးသည် ကြုံတွေ့ရနေကျပင်။

Intent များသည် လက်ရှိ application အတွင်းရှိ ဖြစ်စေ၊ အခြား application များရှိ ဖြစ်စေ activity များစွာ ပါဝင်ပတ်သက်လာစေရန် ခေါ်ယူနိုင်ပုံကို အောက်ပါသရုပ်ပြပုံတွင် တွေ့နိုင်ပါသည်။



၄.၅ Service များ

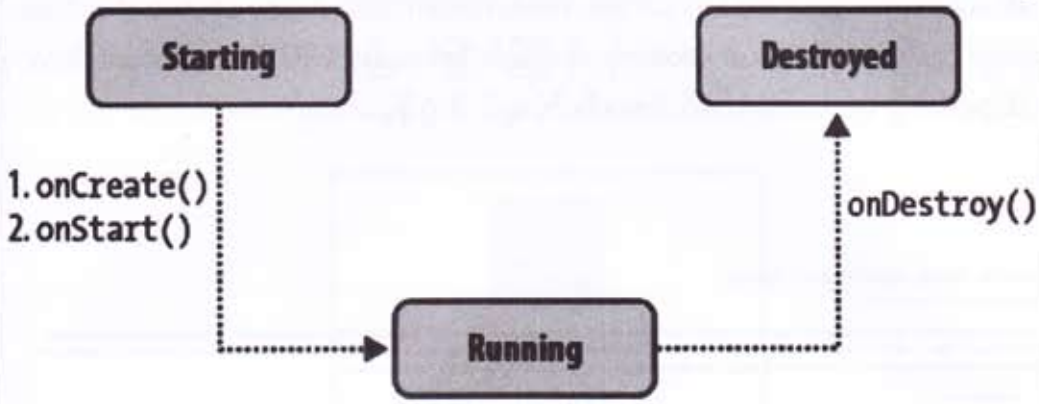
Service များသည်ကား bacground တွင် run နေခြင်းသက်သက် ဖြစ်ပြီး user interface component များ မရှိချေ။ ၄င်းတို့သည် activity များအတိုင်းပင် အလုပ်လုပ်ကြပြီး user interface မရှိခြင်းသာ ထူးခြားလေသည်။ screen ပေါ်တွင် မည်သည့် activity ရှိနေသည် ဖြစ်စေ run နေသေးသည် ဖြစ်သော service များသည် မိုဘိုင်းကြိုးမဲ့ကိရိယာတွင် ရှိနေတတ်သည်။ ဥပမာ ဆိုရလျှင် music player ကို play ထားလျှက်ဖြင့် အခြား application များကို သုံးနေခြင်းမျိုး။

ဤနေရာတွင် ရှင်းလင်းရန် တစ်ချက် ရှိသည်မှာ ယခု ဆွေးနွေးလျက်ရှိသော Android Service များသည် မောင်းနှင်စနစ် operating system ၏ လိုအပ်ချက်ဖြစ်သော Native Linux service များ၊ server များနှင့် lower-level component များ မဟုတ်ပါ။

Service များတွင် activity များထက် ပိုမိုရိုးစင်းသော life cycle ရှိသည်။ စတင်ခြင်း အခြေအနေနှင့် ပြီးဆုံးခြင်း အခြေအနေသာ ရှိသည်။ ထို့ပြင် service များကို program တည်ဆောက်စဉ်ကတည်းက developer က ထိန်းချုပ်မှုကို ဆင်ခဲ့ခြင်း ဖြစ်သည်။ user က သို့မဟုတ် system က တိုက်ရိုက် ထိန်းချုပ်ခြင်း မဟုတ်လှချေ။ ထိုသို့ဖြစ်သောကြောင့် developer အနေနှင့် service များကို စတင်ဖန်တီးစဉ်ကတည်းက ဤ service ကို run လျှင် (CPU နှင့် battery စသော) အခြား app များနှင့် မျှဝေသုံးစွဲရမည့် ရင်းမြစ်များ shared resources ကို မလိုအပ်ဘဲ သုံးစွဲနေခြင်းမျိုး မရှိစေရန် ကြိုး၍ ဆင်ယူခဲ့ရပေမည်။

Service တစ်ခု၏ life cycle ကို တစ်ဖက်စာမျက်နှာရှိ သရုပ်ပြပုံတွင် တွေ့နိုင်ပါသည်။

Service တစ်ခုသည် background တွင် run သည် ဆိုခြင်းကြောင့် ၄င်းသည် သီးခြား thread တစ်ခုကို အသုံးပြုသည်ဟု အမြဲ မဆိုလိုချေ။ ပြီးဆုံးရန် အချိန်အတန်ငယ်ကြာသော service တစ်ခုကို လုပ်ဆောင်နေစဉ် အခြေအနေမျိုးတွင် သီးခြား thread တစ်ခုကို အသုံး



ပြုရမည် မှန်ပေသည်။ ထိုသို့မဟုတ်ပါက user interface သည် သိသိသာသာ နှေးဆင်းသွားပေမည်။ ထိုအခြေအနေမျိုးမှတပါးတွင်မူ UI (user interface) thread ဟုခေါ်သော ပင်မ application ၏ thread တွင်သာလျှင် service များရော activity များပါ အတူတူ run နိုင်ပေသည်။

၄.၆ Content Provider များ

Content Provider များ ဆိုသည်မှာ application များ အချင်းချင်း data များ မျှဝေ share လုပ်ပေးသည့် interface များ ဖြစ်သည်။ ပုံမှန် default အားဖြင့် Android သည် system ပေါ်တွင် application တစ်ခုနှင့် ဆိုင်သော data အားလုံးကို အခြား application များနှင့် ရောနှောမသွားစေပဲ လုံးဝ သီးသန့် ဖြစ်နေစေရန် အတွက် application တစ်ခုစီတိုင်းကို ကိုယ်ပိုင် sandbox များအတွင်း၌ run ပါသည်။ data ပမာဏ နည်းပါက intent များအနေဖြင့် application အချင်းချင်း ဖလှယ်နိုင်သည် မှန်သော်လည်း ကြီးမားသော အစုအဝေးလိုက် အချက်အလက် data set တို့ကို ဖလှယ်မျှဝေရန် ဆိုလျှင်ကား content provider များက ပိုမိုသင့်လျော်ပေသည်။ content provider ၏ API (application interface) သည် CRUD principle ကို ကောင်းစွာ လိုက်နာပေသည်။

CRUD principle ဆိုသည်မှာ create, read, update and delete ဟူသော database ဆိုင်ရာ အခြေခံ operation လေးခုကို ဆိုလိုခြင်း ဖြစ်သည်။ CRUD နှင့်ပတ်သက်သော wikipedia ရှိ ရှင်းလင်းချက်ကို အောက်ပါပုံတွင် တွေ့နိုင်ပါသည်။

Create, read, update and delete

From Wikipedia, the free encyclopedia

In computer programming, **create, read, update and delete** (CRUD) are the four basic functions of persistent storage.[1] Sometimes CRUD is expanded with the words **retrieve** instead of **read** or **destroy** instead of **delete**. It is also sometimes used to describe user interface conventions that facilitate viewing, searching, and changing information; often using computer-based forms and reports. The term was likely first popularized by James Martin in a 1983 book titled "Managing the Data-base Environment".[2]

Contents [hide]

Database applications [edit]

The acronym CRUD refers to all of the major functions that are implemented in relational database applications, that are considered as complete application. Each letter in the acronym is mapped to a standard SQL statement:

Operation	SQL	HTTP/REST
Create	INSERT	POST
Read (Retrieve)	SELECT	GET
Update	UPDATE	PUT
Delete (Destroy)	DELETE	DELETE

Although a relational database is a common persistence layer in software applications, there are numerous others. CRUD can be implemented with an object database, an XML database, flat text files, custom file formats, tape, or card, for example.

User interface [edit]

CRUD is also relevant at the user interface level of most applications. For example, in address book software, the basic storage unit is an individual *contact* entry. As a bare minimum, the software must allow the user to:

- Create or add new entries
- Read, retrieve, search, or view existing entries
- Update or edit existing entries
- Delete existing entries

Without at least these four operations, the software cannot be considered complete. Because these operations are so fundamental, they are often documented and described under one comprehensive heading, such as "contact management" or "contact maintenance" (or "document management" in general, depending on the basic storage unit for the particular application).

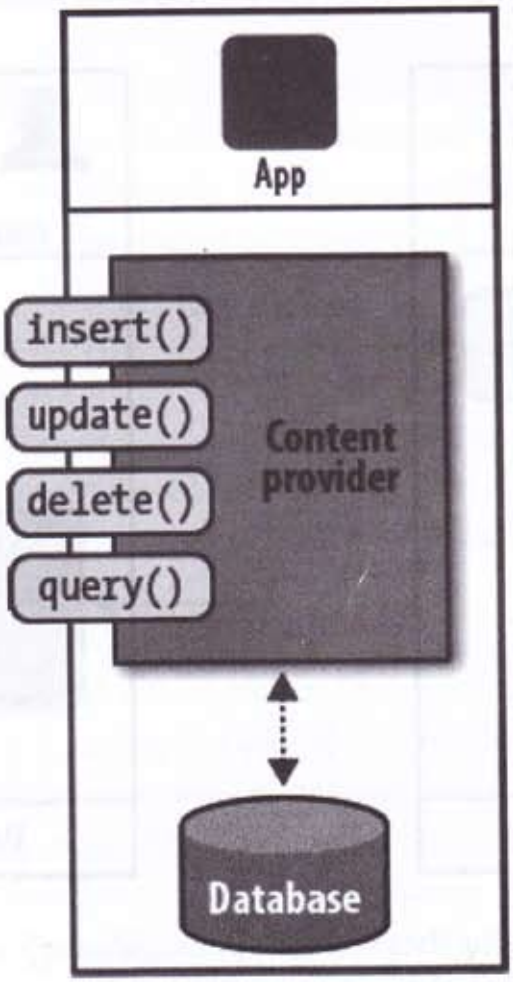
See also [edit]

- Representational state transfer (REST)
- Active record pattern
- Data manipulation language

Notes [edit]

1. ^ InfoWorld | Developer World | REST and CRUD: the impedance mismatch | January 29, 2007 08:00 AM | By Martin Heller
2. ^ James Martin, Managing the Data-base Environment, Prentice-Hall, 1983

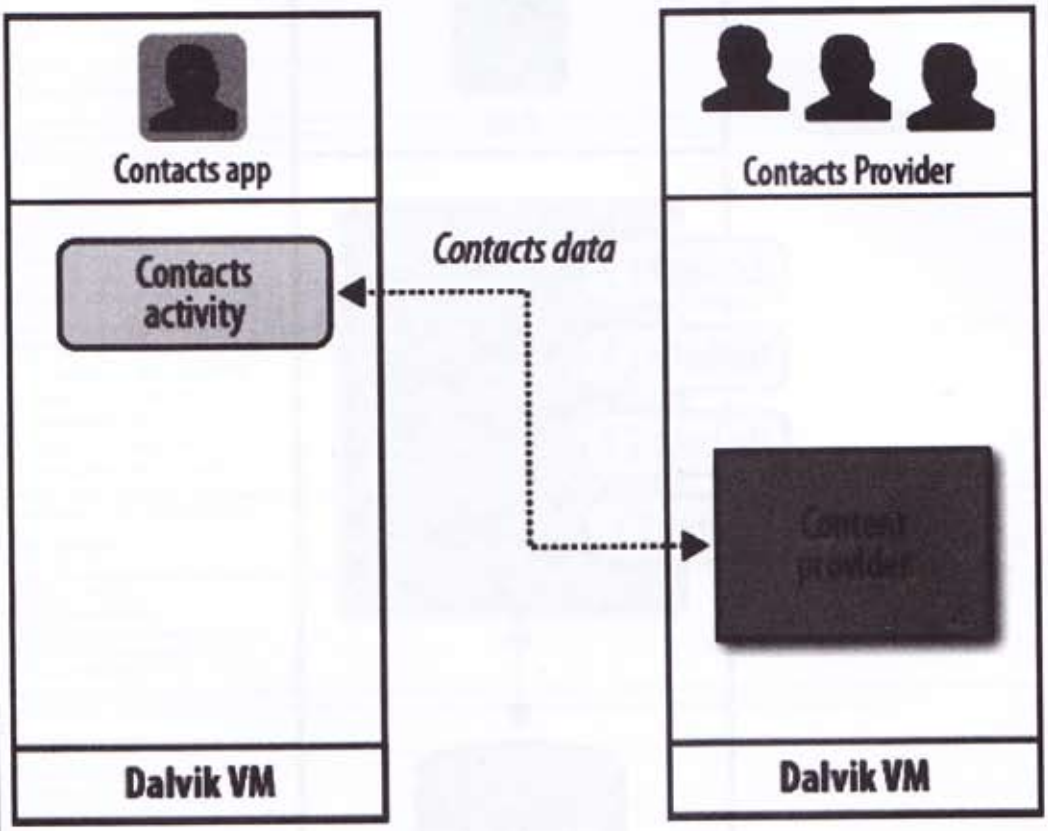
content provider ၏ CRUD interface သည် application တို့၏ sandbox နယ်နိမိတ်များကို ကျော်ဖြတ်ထိုးဖောက်ပြီး အခြား application ကို ချိတ်ဆက်ခွင့်ပြုကာ data များ ဖလှယ်မျှဝေ share လုပ်ပုံကို အောက်ပါသရုပ်ဖော်ပုံတွင် လေ့လာနိုင်ပါသည်။



Android system သည် ဤယန္တရားစနစ်ကို တောက်လျှောက် အသုံးပြုနေသည်။ ဥပမာ ဆိုရလျှင် Contacts သည် application မျိုးစုံဆီသို့ user ၏ ဆက်သွယ်ပုဂ္ဂိုလ်ဆိုင်ရာ အချက်အလက် contact data အားလုံးကို ပို့သည့် content provider တစ်ခုဖြစ်သည်။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Media Store သည် ဓာတ်ပုံနှင့် သီချင်းများ ကဲ့သို့သော မီဒီယာမျိုးစုံကို သိုမှီးမှုနှင့် မျှဝေမှု တို့ကို ဆောင်ရွက်သည်။ Contact (app) က အခြားသီးသန့် application သပ်သပ်တစ်ခု ဖြစ်သည့် Contact Provider ကို အသုံးပြုပြီး ဆက်သွယ်ပုဂ္ဂိုလ်များဆိုင်ရာ အချက်အလက် တို့အား အပြန်အလှန်ရယူပုံကို အောက်ပါသရုပ်ပြပုံတွင် လေ့လာနိုင်ပါသည်။



ဤသို့ အချက်အလက်သိုမှီးမှု data storage နှင့် သုံးသူမြင်ရသည့် ပရိုဂရမ် user interface application ကို သီးခြားစီ ခွဲထုတ်ထားလိုက်ခြင်းသည် system ပေါ်ရှိ application မျိုးစုံကို ရောသမမွှေသုံးစွဲရာတွင်လည်းကောင်း၊ အစိတ်အပိုင်းငယ်ကလေးပေါင်း များစွာ အဖြစ်သို့ ခွဲထုတ်ရာတွင်လည်းကောင်း ပိုမိုလိုက်လျောညီထွေဖြစ်စေသည်။ ဥပမာ ဆိုရလျှင်

user သည် မိုဘိုင်းကြိုးမဲ့ကိရိယာ၏ မူရင်းပါ Contacts (app) ကို မသုံးဘဲ အခြား Address Book (app) တစ်ခုကို install လုပ်ပြီး Contacts (app) ထဲရှိ data များကိုပင် အသုံးချနိုင်သကဲ့သို့။ သို့တည်းမဟုတ်လည်း Wi Fi / Bluetooth / GPS စသော feature များကို Home screen ပေါ်တွင်ပင် widget များတင်ပြီး ပိတ်နိုင်ဖွင့်နိုင်သကဲ့သို့။

မိုဘိုင်းဖုန်းထုတ်လုပ်သည့် လုပ်ငန်း များစွာသည် content provider များ၏ ဤအားသာချက်ကို အသုံးပြုကာ standard Android platform ပေါ်တွင် သူတို့၏ ကိုယ်ပိုင် application များကို ထပ်ပေါင်းထည့်ပြီး အသုံးပြုသူများ အဆင်ပြေရေးကို ဖန်တီးကာ အသုံးပြုသူတို့ကို ဆွဲဆောင်ကြလေသည်။ (HTC Sense ကဲ့သို့)

၄.၇ Broadcast Receiver များ

Broadcast Receiver များသည် Android ၏ system တစ်ခုလုံးဆိုင်ရာ publish/subscribe mechanism လုပ်ဆောင်မှု ဖြစ်သည်။ receiver ဆိုသည်မှာ ပုံမှန်အားဖြင့် ငြိမ်နေ၍ ၄င်းကို subscribe လုပ်ထားသော ဖြစ်ရပ် event တစ်ခု ဖြစ်ပွားပါမှ အသက်ဝင်လာသော code တစ်ခုသာ ဖြစ်သည်။

system ကိုယ်တိုင်သည် ဖြစ်ရပ် event များကို တချိန်လုံး ထုတ်လွှင့် broadcast လုပ်နေသည်သာ ဖြစ်သည်။ ထိုသို့ ထုတ်လွှတ်နေမှုကြောင့် တာဝါမှ လာလတ္တံ့ ဖြစ်သော SMS ပို့လွှတ်ခြင်း၊ အဝင်ခေါ်ဆိုမှု incoming call ဝင်လာခြင် စသည့် ဖြစ်ရပ်တို့ကို ကြိုးမဲ့ဆက်သွယ်ရေး ကိရိယာမှ အချိန်မဆိုင်း သိနေခြင်း ဖြစ်သည်။ ထို့ပြင် ကြိုးမဲ့ဆက်သွယ်ရေး ကိရိယာ system မှ broadcast လုပ်သည့် ဘက်ထရီလျော့ အချက်ပေးမှု၊ သို့မဟုတ် မိုဘိုင်းဖုန်း ပါဝါစဖွင့် boot လုပ်ခြင်း စသည့် အခြား system ဆိုင်ရာ event များလည်း ရှိပေသည်။ ထို broadcast event များကြောင့် receiver (များစွာ)ကို trigger ဖြစ်သွားစေနိုင်ဖို့ လိုအပ်တတ်ပေသည်။

public/subscribe mechanism ဆိုသည်မှာ message တစ်ခု၏ sender (pulisher) က ဦးတည်ရာ receiver (subscriber) (များ) ဆီသို့ message ကို တိုက်ရိုက်ပို့လွှတ်လိုက်စေရန် ပရိုဂရမ်ရေးဆွဲထားခြင်း မရှိသော ပို့လွှတ်မှုပုံစံ messaging pattern ဖြစ်သည်။ (message ဆိုရာတွင် မိုဘိုင်းဖုန်းတို့၏ စာတိုပို့စနစ် SMS မှ message များကို ဆိုလိုခြင်း မဟုတ်ပါ။ application များအချင်းချင်းကြား အပြန်အလှန်သွားနေသည့် စက်နားလည်သော ဘာသာစကားဖြင့် သတင်းပို့မှုများ machine code message များကို ဆိုလိုခြင်း ဖြစ်ပါသည်။) pub/sub mechanism အကြောင်း wikipedia မှ ရှင်းလင်းချက်များနှင့် နမူနာ code များကို အောက်ပါအတိုင်း တွေ့နိုင်ပါသည်။

Publish/subscribe

From Wikipedia, the free encyclopedia

For document linking used in Apple computers, see Publish and Subscribe.

"PubSub" redirects here. For the defunct search website, see PubSub (website).

This article **does not cite any references or sources. Please help** improve this article **by adding citations to** reliable sources. Unsourced material may be challenged and removed. *(March 2010)*

Pub/sub is a messaging pattern where senders(publishers) of messages do not program the messages to be sent directly to specific receivers(subscribers). Published messages are characterized into classes, without knowledge of what, if any, subscribers there may be. Subscribers express interest in one or more classes, and only receives messages that are of interest, without knowledge of what, if any, publishers there are.

Pub/sub is a sibling of the message queue paradigm, and is typically one part of a larger message-oriented middleware system. Most messaging systems support both the pub/sub and message queue models in their API, e.g. Java Message Service (JMS).

This pattern provides greater network scalability and a more dynamic network topology.

Contents

Message filtering [edit]

In the pub/sub model, subscribers typically receive only a subset of the total messages published. The process of selecting messages for reception and processing is called filtering. There are two common forms of filtering: topic-based and content-based.

In a **topic-based** system, messages are published to "topics" or named logical channels. Subscribers in a topic-based system will receive all messages published to the topics to which they subscribe, and all subscribers to a topic will receive the same messages. The publisher is responsible for defining the classes of messages to which subscribers can subscribe.

In a **content-based** system, messages are only delivered to a subscriber if the attributes or content of those messages match constraints defined by the subscriber. The subscriber is responsible for classifying the messages.

Some systems support a hybrid of the two; publishers post messages to a topic while subscribers register content-based subscriptions to one or more topics.

Topologies [edit]

In many pub/sub systems, publishers post messages to an intermediary message broker and subscribers register subscriptions with that broker, letting the broker perform the filtering. The broker normally performs a store and forward function to route messages from publishers to subscribers.

History [edit]

One of the earliest publicly described pub/sub systems was the "news" subsystem of the Isis Toolkit, described at the 1987 Association for Computing Machinery (ACM) Symposium on Operating Systems Principles conference (SOSP '87), in a paper "Exploiting Virtual Synchrony in Distributed Systems. 123-138"[1]. The pub/sub technology described therein was invented by Frank Schmuck.

Advantages [edit]

Loosely coupled: Publishers are loosely coupled to subscribers, and need not even know of their existence. With the topic being the focus, publishers and subscribers are allowed to remain ignorant of system topology. Each can continue to operate normally regardless of the other. In the traditional tightly-coupled client–server paradigm, the client cannot post messages to the server while the server process is not running, nor can the server receive messages unless the client is running. Many pub/sub systems decouple not only the locations of the publishers and subscribers, but also decouple them temporally. A common strategy used by middleware analysts with such pub/sub systems is to take down a publisher to allow the subscriber to work through the backlog (a form of bandwidth throttling).

Scalable: For relatively small installations, pub/sub provides the opportunity for better scalability than traditional client-server, through parallel operation, message caching, tree-based or network-based routing, etc. However, as systems scale up to become data centers with thousands of servers sharing the pub/sub infrastructure, this benefit is often lost, scalability for pub/sub products under high load in large deployments is a research challenge.

Disadvantages [edit]

The most serious problems with pub/sub systems are a side-effect of their main advantage: the decoupling of publisher from subscriber. The problem is that it can be hard to specify stronger properties that the application might need on an end-to-end basis:

- As a first example, many pub/sub systems will try to deliver messages for a little while, but then give up. If an application actually needs a stronger guarantee (such as: messages will always be delivered or, if delivery cannot be confirmed, the publisher will be informed), the pub/sub system probably won't have a way to provide that property.
- Another example arises when a publisher "assumes" that a subscriber is listening. Suppose that we use a pub/sub system to log problems in a factory: any application that senses an error publishes an appropriate message, and the messages are displayed on a console by the logger daemon, which subscribes to the "errors" topic. If the logger happens to crash, publishers won't have any way to see this, and all the error messages will vanish.

As noted above, while pub/sub scales very well with small installations, a major difficulty is that the technology often scales poorly in larger ones. These manifest themselves as instabilities in throughput (load surges followed by long silence periods), slowdowns as more and more applications use the system (even if they are communicating on disjoint topics), and so-called IP broadcast storms, which can shut down a local area network by saturating it with overhead messages that choke out all normal traffic, even traffic unrelated to pub/sub.

For pub/sub systems that use brokers (servers), the agreement for a broker to send messages to a subscriber is in-band, and can be subject to security problems. Brokers might be fooled into sending notifications to the wrong client, amplifying denial of service requests against the client. Brokers themselves could be overloaded as they allocate resources to track created subscriptions.

Even with systems that do not rely on brokers, a subscriber might be able to receive data that it is not authorized to receive. An unauthorized publisher may be able to introduce incorrect or damaging messages into the pub/sub system. This is especially true with systems that broadcast or multicast their messages. Encryption (e.g. Transport Layer Security (SSL/TLS)) can be the only strong defense against unauthorized access.

See also [edit]

- PubSubHubbub (an implementation of pub/sub)
- Event-driven programming
- Observer Pattern
- DDS
- Push technology
- Usenet

References [edit]

1. ^ Birman, K. and Joseph, T. "Exploiting virtual synchrony in distributed systems" in *Proceedings of the eleventh ACM Symposium on Operating systems principles (SOSP '87)*, 1987. pp. 123-138.

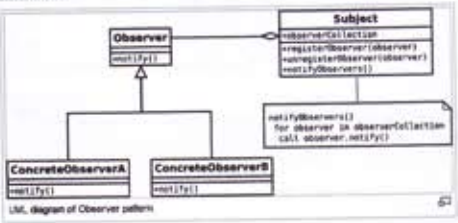
External links [edit]

- XMPP XEP-0060: Publish-Subscribe
- For an open source example which is in production on MSN.com and Microsoft.com, see Distributed Publish/Subscribe Event System
- Python PubSub a Python Publish-Subscribe broker for messages "within" an application (NOT network)
- A Ruby open source example: Faye

Categories: Message-oriented middleware

Transferring data from en.wikipedia.org...

Structure [edit]



List diagram of Observer pattern

Definition [edit]

The essence of the Observer Pattern is to "Define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically." [1]

Example [edit]

Below is an example that takes keyboard input and treats each input line as an event. The example is built upon the library classes `java.util.Observer` and `java.util.Observable`. When a string is supplied from System.in, the method `notifyObservers` is then called, in order to notify all observers of the event's occurrence, in the form of an invocation of their 'update' methods - in our example, `ResponseHandler.update(...)`.

The Wikibook *Computer Science Design Patterns* has a page on the topic of Observer implementations in various languages

The file `MyApp.java` contains a `main()` method that might be used in order to run the code

```
/* File Name : EventSource.java */
package obs;

import java.util.Observable;          //Observable is here
import java.io.BufferedReader;
import java.io.IOException;
```

```
import java.io.InputStreamReader;

public class EventSource extends Observable implements Runnable {
    public void run() {
        try {
            final InputStreamReader isr = new InputStreamReader( System.in );
            final BufferedReader br = new BufferedReader( isr );
            while( true ) {
                String response = br.readLine();
                setChanged();
                notifyObservers( response );
            }
        }
        catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

```
/* File Name: ResponseHandler.java */

package obs;

import java.util.Observable;
import java.util.Observer;  /* this is Event Handler */

public class ResponseHandler implements Observer {
    private String resp;
    public void update (Observable obj, Object arg) {
        if (arg instanceof String) {
            resp = (String) arg;
            System.out.println("\nReceived Response: "+ resp );
        }
    }
}
```

```
/* Filename : MyApp.java */
/* This is the main program */

package obs;

public class MyApp {
    public static void main(String args[]) {
        System.out.println("Enter Text >");

        // create an event source - reads from stdin
        final EventSource evSrc = new EventSource();

        // create an observer
        final ResponseHandler respHandler = new ResponseHandler();

        // subscribe the observer to the event source
        evSrc.addObserver( respHandler );

        // starts the event thread
        Thread thread = new Thread(evSrc);
        thread.start();
    }
}
```

Implementations [edit]

The observer pattern is implemented in numerous programming libraries and systems, including almost all GUI toolkits.

Some of the most notable implementations of this pattern:

ActionScript [edit]

- flash.events, a package in ActionScript 3.0 (following from the mx.events package in ActionScript 2.0).

C [edit]

- GObject, in GLib - an implementation of objects and signals/callbacks in C. (This library has many bindings to other programming languages.)

C++ [edit]

- libsigc++ - the C++ signalling template library.
- sigslot - C++ Signal/Slot Library
- Cpp::Events - Template-based C++ implementation that introduces separation of connection management interface of the event object from the invocation interface.
- XUObject - Template-based C++ signal/slot model patterned after Qt.
- Signals - A lightweight and non-intrusive C++ signal/slot model implementation.
- libevent - Multi-threaded Crossplatform Signal/Slot C++ Library
- Boost.Signals, an implementation of signal/slot model
- MFC's CDocument-CView-framework
- The Qt C++ framework's signal/slot model
- The MRPT robotics C++ framework's observer/observable model.

Objective C [edit]

- NSKeyValueObserving - The Objective C NSKeyValueObserving protocol

C# [edit]

- The IObserver<T> Interface - The .NET Framework supported way of implementing the observer pattern
- Exploring the Observer Design Pattern - the C# and Visual Basic .NET implementation, using delegates and the Event pattern

ColdFusion [edit]

- http://www.cfdesignpatterns.com/behavioral-patterns/observer-design-pattern-in-coldfusion/

Delphi [edit]

- Delphi Observer Pattern, a Delphi implementation

Java [edit]

- The class `java.util.Observer`[2] provides a simple observer implementation.
- Events are typically implemented in Java through the callback pattern: one piece of code provides a common interface with as many methods as many events are required[3], another piece of code provides an implementation of that interface, and another one receives that implementation, and raises events as necessary.[4]

JavaScript [edit]

- JavaScript Observer Pattern. Pocket-sized minimalist framework of common design patterns in JavaScript
- EventDispatcher singleton, a JavaScript core API based Signals and slots implementation - an observer concept different from Publish/subscribe - pretty lightweighted but still type-safety enforcing.

Lisp [edit]

- Cells, a dataflow extension to Common Lisp that uses meta-programming to hide some of the details of Observer pattern implementation.

Perl [edit]

- Class::Observable Basic observer pattern implementation
- Class::Publisher a slightly more advanced implementation

PHP [edit]

- Event_Dispatcher, a PHP implementation
- SPL, the Standard PHP Library
- Symfony Event Dispatcher, a standalone library by the Symfony team

Python [edit]

- Louie, an implementation by Patrick K. O'Brien
- PyDispatcher, the implementation on which the Django web framework's signals are based
- Pyvently, a Python (plus a little C) implementation
- Observer Pattern using Weak References implementation by Michael Kent
- PyPubSub an in-application Pub/Sub library for Observer behavior
- NotificationFramework classes directly implementing Observer patterns
- Blinker, an implementation which can be used with decorators

Ruby [edit]

- Observer, from the Ruby Standard Library

Other/Misc [edit]

- CSP - *Observer Pattern using CSP-like Rendezvous* (each actor is a process, communication is via rendezvous)
- YUI Event utility implements custom events through the observer pattern
- Publish/Subscribe with LabVIEW, Implementation example of Observer or Publish/Subscribe using G

Critics [edit]

The Observer pattern is criticized[5] for being too verbose, introducing too many bugs and violating software engineering principles, such as not promoting side-effects, encapsulation, composability, separation of concepts, scalability, uniformity, abstraction, resource management, semantic distance. The recommended approach is to gradually deprecate observers in favor of reactive programming abstractions.

See also [edit]

- Design Patterns (book), the book which gave rise to the study of design patterns in computer science
- Design pattern (computer science), a standard solution to common problems in software design
- Implicit invocation
- Model-view-controller (MVC)
- Client–server model

References [edit]

1. ^ Gang Of Four
2. ^ java.util.Observer
3. ^ Which will typically implement interface java.util.EventListener
4. ^ http://download.oracle.com/javase/tutorial/uiswing/events/generalrules.html

ရှေ့တွင်ဆွေးနွေးခဲ့သော twitter ဥပမာနှင့် ချိတ်ဆက်ရလျှင် မိုဘိုင်းကြိုးမဲ့ကိရိယာ၏ system ကို ပါဝါစဖွင့်သည်နှင့် update (service) ကို စတင်စေလိုသည်။ ထိုသို့ စတင်နိုင်ရန်အတွက် (developer အနေဖြင့်) system ၏ boot up လုပ်ဆောင်ချက် ပြီးစီးပြီ ဖြစ်ကြောင်း system မှ ထုတ်လွှင့်သည့် broadcast တွင် subscribe လုပ်ထားရပေမည်။

အလားတူ သဘောတရားအတိုင်းပင် ကိုယ်ဖန်တီးသည့် application များတွင်လည်း application ၏ အစိတ်အပိုင်းတစ်ခုမှ တစ်ခုသို့ လည်းကောင်း၊ application တစ်ခုမှ အခြား application တစ်ခုသို့ လည်းကောင်း ကိုယ်ပိုင် broadcast များ ဖန်တီးနိုင်သည်။

Broadcast receiver များသည် မြင်သာသိသာခြင်း တစုံတရာလည်း မရှိသည့်အပြင် ၄င်းတို့သည် memory ပေါ်၌လည်း active အနေအထားဖြင့် run နေခြင်းမျိုး မဟုတ်ပေ။ သို့သော် trigger လုပ်လိုက်ခြင်း ခံရသည့်အခါတွင်ကား ၄င်းတို့သည် activity တစ်ခု၊ service တစ်ခု သို့မဟုတ် အခြား တစ်ခုခုကို စတင်ခြင်း ဖြစ်စေသည့် code တစ်ခုကို ဆောင်ရွက် execute လုပ်လေသည်။

၄.၈ Application Context များ

ယခုအထိ ဆွေးနွေးခဲ့ရာတွင် activity များ၊ service များ၊ content provider များ၊ broadcast receiver များအကြောင်းကို သိခဲ့ပေပြီ။ ၄င်းတို့ အားလုံးပေါင်း၍ application တစ်ခုကို ဖြစ်စေလေသည်။ တနည်းဆိုလျှင် ၄င်းတို့သည် application context တစ်ခုတည်းအတွင်း ရှိကြလေသည်။

Application context ဆိုသည်မှာ application တစ်ခုအတွက် ပတ်သက်ရာ component များ အားလုံး ရှိနေ run နေသော နယ်ပယ်နှင့် ဖြစ်စဉ် (environment and process) ဖြစ်သည်။ ၄င်းသည် application တစ်ခုအတွက် အသီးသီးသော အခြေခံ building block component များအကြား အချက်အလက်များ data နှင့် ရင်းမြစ်များ

resources ကို မျှဝေနိုင်စေရန် ဖန်တီးပေးသည်။

application တစ်ခု၏ (activity ဖြစ်စေ၊ service ဖြစ်စေ၊ broadcast ဖြစ်စေ၊အခြား မည်သည့်အရာဖြစ်စေ) ပထမဆုံး component ကို စတင်လိုက်သည်နှင့် application context ကို Android မောင်းနှင်စနစ် operating system က ဖန်တီးလိုက်ပြီး ဖြစ်သည်။ ဤ application အသက်ရှိနေသရွေ့ ဤ application ၏ context မှာ ရှိနေမည် ဖြစ်သည်။ သို့ဖြစ်၍ application context သည် activity life cycle နှင့် မချိတ်ဆက်ချေ။ context ၏ reference ကို Context.getApplication() သို့မဟုတ် Activity.getApplicationContext() ဟု function ခေါ်ယူခြင်းအားဖြင့် အလွယ်တကူ ရနိုင်ပါသည်။ activity နှင့် service တို့သည် context ၏ subclass များ ဖြစ်နေနှင့်ပြီး သောကြောင့် context ၏ method အားလုံးကိုလည်း inherit ရရှိထားပြီး ဖြစ်ပါသည်။

Android application

ရေးသားစနစ်တီးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

အခန်း – ၅
Project မိတ်ဆက်

Android application ဖန်တီးမှုကို လေ့လာရန် အကောင်းဆုံးနည်းမှာ နမူနာ project တစ်ခု ဖန်တီးကြည့်လိုက်ခြင်းပင် ဖြစ်သည်။ ဤနမူနာ project သည်လည်း အချို့သော စံနှုန်းများကို ပြည့်မီရန်လိုပေသည်။ အကောင်းဆုံးဟုဆိုထိုက်သည့် နမူနာ project တစ်ခု သည်

၁. နားလည်လွယ်ရမည် – (Androidကို လေ့လာနေသည့်အတွက်) Androidသီးသန့်နှင့် သာ ဆိုင်သည့် Application framework နှင့် ပတ်သက်သော ရှုထောင့်နှင့် အချက် အားလုံးနီးပါးကို သရုပ်ဖော်နိုင်စွမ်း ရှိရမည်။ ထို့ထက်လွန်၍ ဆိုရလျှင် ဖြစ်လိုသည့် အလုပ် တစ်ခုကို ပြီးမြောက်သွားစေရန် (အခြားနည်းများဖြင့် လုပ်ခြင်းကို မရွေးပဲ) တစ်ခုသော feature ကို အသုံးပြုလိုက်ခြင်းတိုင်း၌ ကောင်းမွန်ခိုင်လုံသော အကြောင်း အရင်း ရှိရမည်။ Android နှင့် မရင်းနှီးသေးသူများ၊ Android platform ပေါ်တွင် application များဖန်တီးခြင်းကို ယခုမှစတင်လေ့လာသူများ အဖို့ စတင်စဉ်ကတည်းက ရှိအပ်သော သဘောထားများကို ရရှိလာစေရန် ဤအချက်သည် အရေးကြီး၏။

၂. ရင်းနှီးကျွမ်းဝင်သည် ဖြစ်ရမည် – နမူနာ application သည် ရိုးရှင်းပြီး မြင်လွယ်ရ မည်။ ယခုနမူနာသည် design ပိုင်းနှင့် ပြီးပြည့်စုံအောင် တနည်းအားဖြင့် စ၊လယ်၊ဆုံး ချိတ်ဆက်အောင် ဖန်တီးမှုပိုင်းကိုသာ အဓိကထားသည်။ feature များနှင့် အကျိုး သက်ရောက်မှုများ အပေါ် အဓိကထားခြင်း မဟုတ်ပါ။

၅.၁ နမူနာ Yamba Application

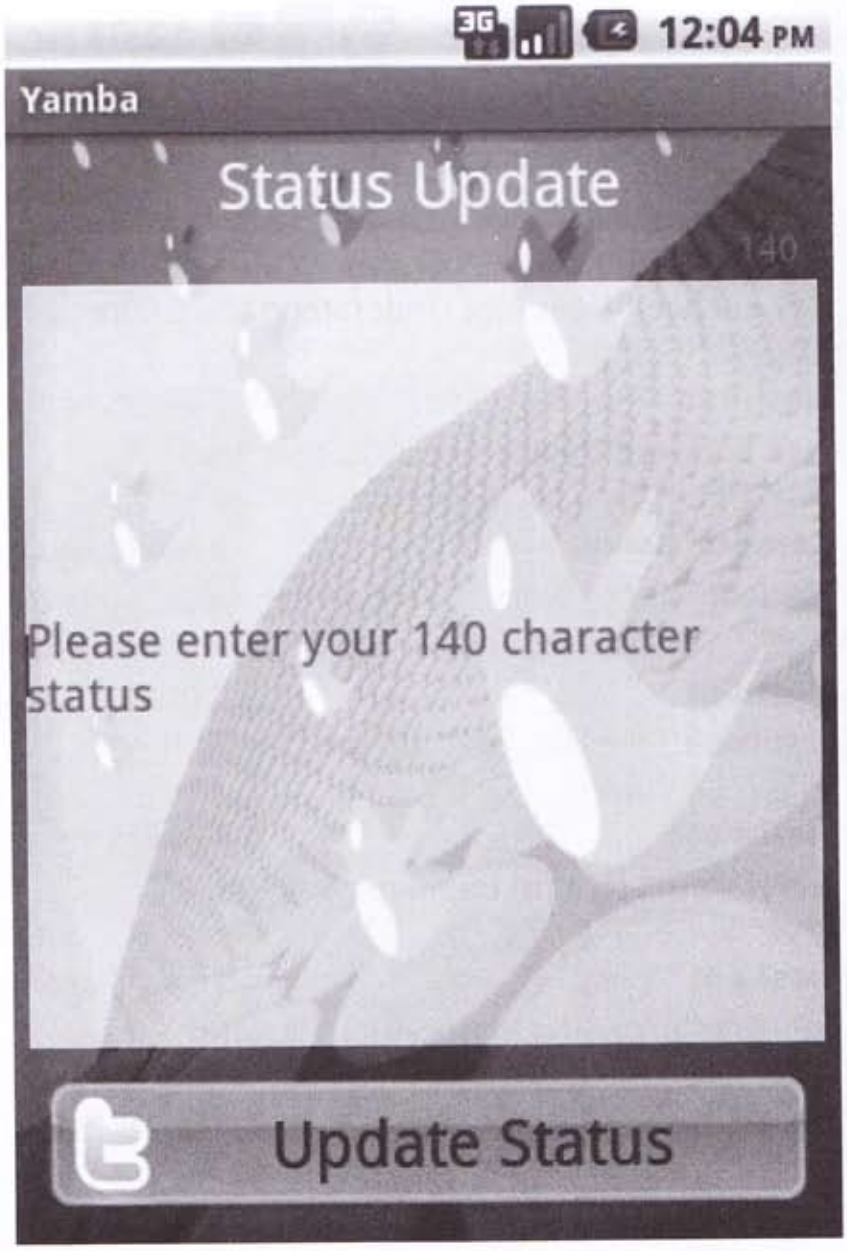
ယခုရွေးချယ်ထားသည့် နမူနာမှာ ရှေ့အခန်းမှ ဆွေးနွေးလက်စ ဖြစ်သည့် Twitter နှင့် တူသော application ဖြစ်သည်။ Yamba ဟု အမည်ပေးပါမည်။ (နှစ်သက်ရာ ပေးနိုင်ပါသည်) Yamba သည် user ကို twitter ကဲ့သို့ ဝန်ဆောင်မှု တစ်ခုနှင့် ချိတ်ဆက်ပေးမည်၊ friends တို့၏ status များကို ဆွဲချယူကာ ပြသပေးမည်၊ user ၏ ကိုယ်ပိုင် status များကို update တင်ပေးမည်။

Yamba သည် လုပ်ဆောင်ချက်သဘာဝအားဖြင့်ပင်လျှင် Android ၏ အခြေခံအုတ်မြစ် (building block) component များကို ခြိုမိလေသည်။ သို့ဖြစ်သောကြောင့် Android ၏ မျိုးစုံသော component အသီးသီးတို့သည် ကိုယ့်ဘာသာကိုယ် မည်သို့ အလုပ်လုပ်ကြသည်၊ ထိုသို့ ဖြစ်နေရင်းမှ တချိန်တည်းတွင်ပင် အချင်းချင်း မည်သို့ အံဝင်ခွင်ကျ ပြန်ဖြစ်ကြသည် စသည်တို့ကို နမူနာပြု သရုပ်ဖော်ရန် သင့်လျော်လှသော application ဖြစ်၏။ twitter ကဲ့သို့သော ဝန်ဆောင်မှုမျိုးများသည် ယနေ့ခေတ် လူအားလုံးနီးပါးနှင့် ကျွမ်းဝင်လျက် ရှိနေလေပြီ။ twitter သည် အကြမ်းအားဖြင့် ဆိုရလျှင် facebook မှ wall ၏ အလုပ်လုပ်ပုံအတိုင်းပင်။

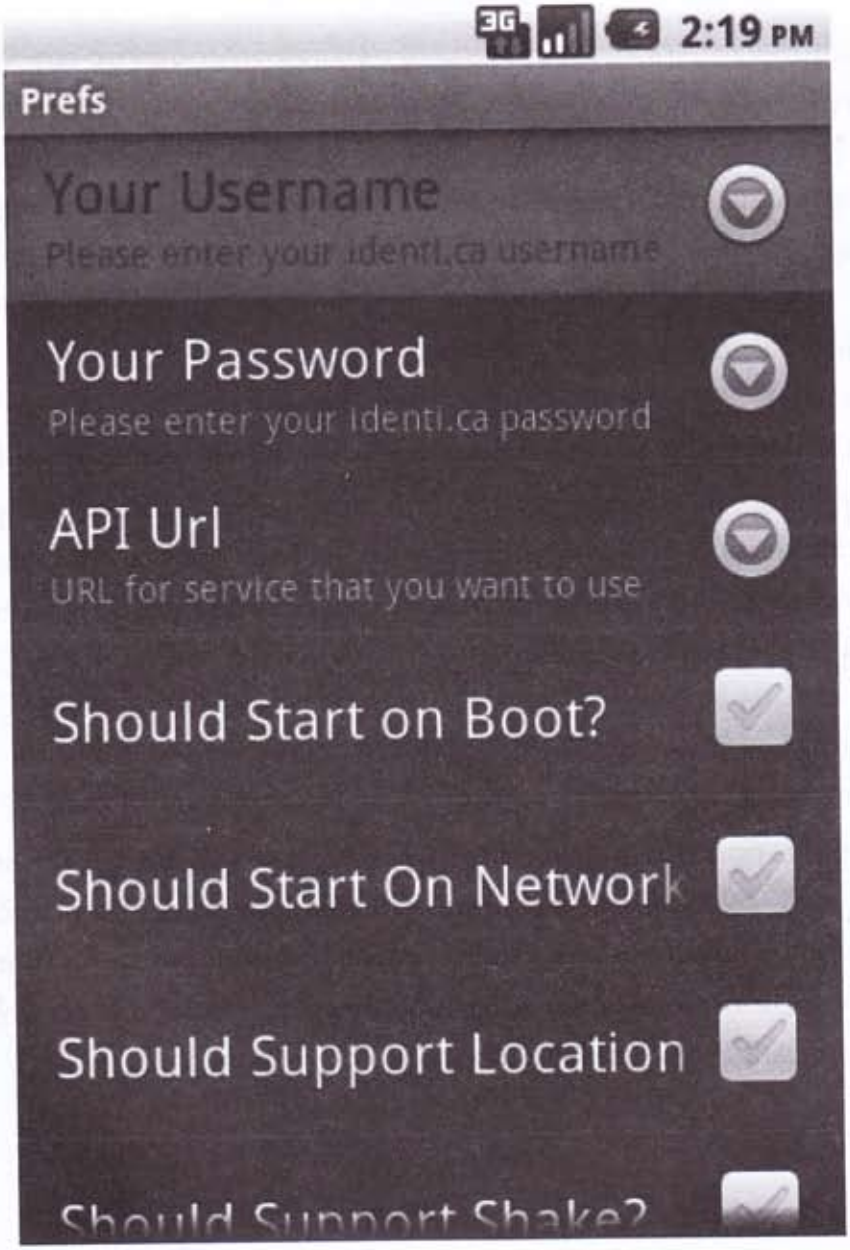
Yamba သည် အားလုံးပြီးစီးသွားလျှင် မည်သို့ ဖြစ်ရမည် ဆိုသည်ကို ရှေ့စာမျက်နှာများမှ ပုံများတွင် ကြည့်ပါ။

အခြားသူများ၏ status message များကို ဖော်ပြမည့် screen (Timeline ဟု ခေါ်သည်)

user က သူ၏ status message ကို ရိုက်ထည့်နိုင်မည့် screen (ပထမဆုံး ပေါ်ရမည်)



user က စိတ်ကြိုက် Preference များကို ပြောင်လဲနိုင်မည့် screen



၅.၂ Design သဘောတရား

ဤ design သဘောတရားသည် ဤ project ကို ဖန်တီးဆောက်လုပ်ရာတွင် ကျောရိုးအဖြစ် လမ်းညွှန်မှုပေးမည်ဖြစ်ပြီး ဘာကိုဆက်လုပ်ရမည်မသိသည့် အခြေအနေတိုင်းတွင် ပြန်လည်စဉ်းစားကိုးကားရမည့် သံလိုက်အိမ်မြှောင် ဖြစ်ပေမည်။ ထို့ပြင် design သဘောတရားကို ရှင်းလင်းစွာ ချရေးလိုက်ခြင်းအားဖြင့် ရှေ့ဆက်မည့် လုပ်ဆောင်ချက်များတွင် ရှုပ်ထွေးမှုများ ပေါ်လာမည့်အရေးကိုလည်း ရှင်းလင်းလိုက်ပြီး ဖြစ်သွားစေသည်။

တဖြည်းဖြည်းချင်း တိုးသွားမည်

Yamba application သည် သေးသေးလေးက စမည်၊ လုပ်ဆောင်ချက်ပိုင်ရော ရှုပ်ထွေးမှုပိုင်းရောတွင် တဖြည်းဖြည်းချင်း ပုံမှန် တိုး၍ တိုး၍ လာမည်။ စစချင်းတွင် Yamba application သည် ဘာမျှ ရှိဦးမည် မဟုတ်။ (facebook account တစ်ခုကို တည်ဆောက်သကဲ့သို့၊ email account တစ်ခုကို တည်ဆောက်သကဲ့သို့ပင်) တစ်ဆင့်ချင်း တစ်ဆင့်ချင်း တိုးပွားကျယ်ပြန့်သွားမည်။

အမြဲတမ်း ချိတ်ဆက်မိစွာ အလုပ်လုပ်နေရမည်

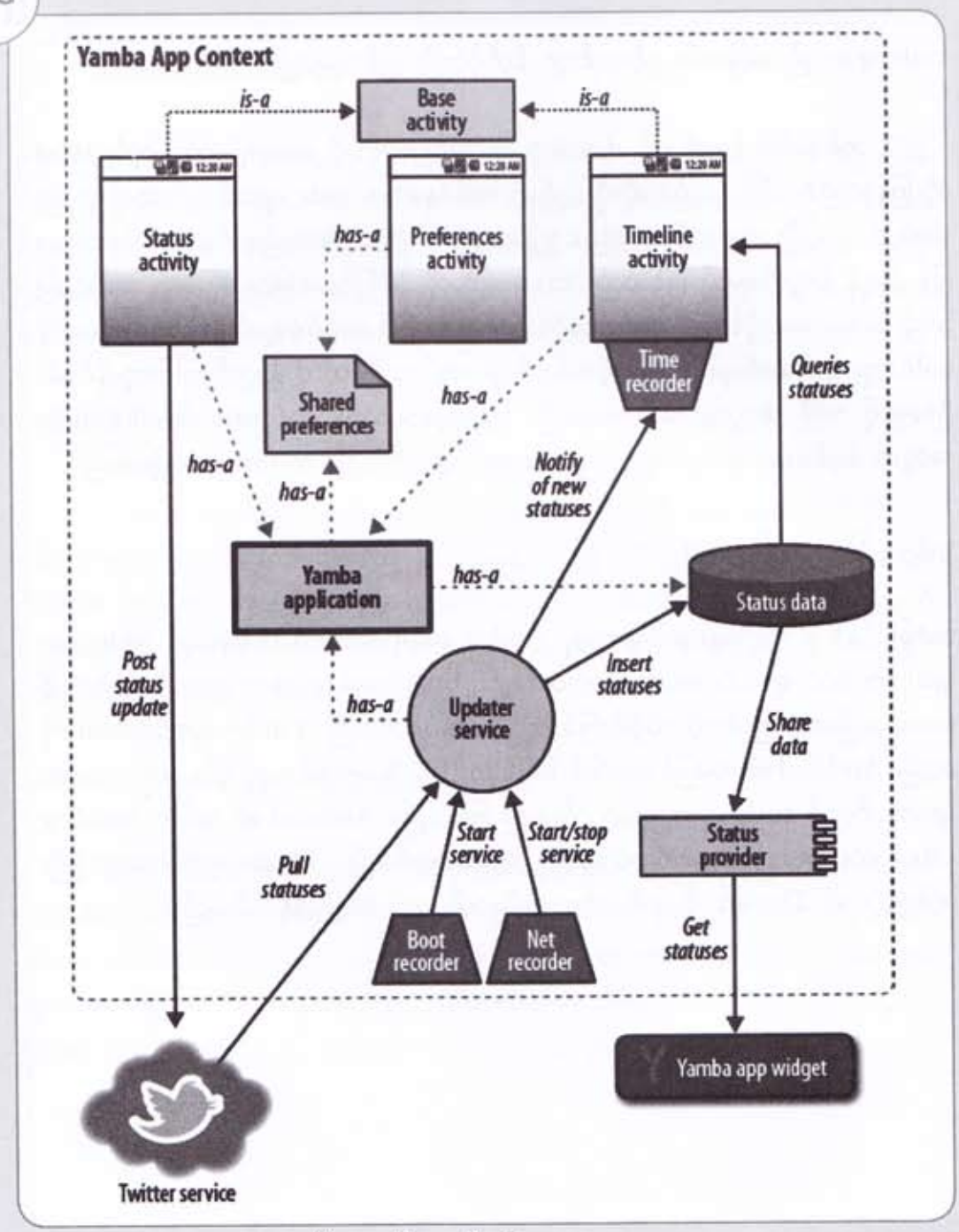
application သည် အမြဲတမ်း အလုပ်ဖြစ်နေရမည်။ တနည်းဆိုရလျှင် feature သစ်တစ်ခုစီတိုင်းကို သေးငယ်သော၊ ကိုယ့်ဘာသာကိုယ်ရပ်တည်နိုင်သော၊ အလုပ်လုပ်သော၊ error ကင်းသော အစုငယ်ကလေးများအဖြစ် ဖန်တီးပြီးနောက် ပင်မ application တွင် သွားရောက်ချိတ်ဆက်ကာ အံဝင်ခွင်ကျ ဖြစ်စေရမည်။ မည်သည့်အဆင့်တွင် ရပ်လိုက်သည် ဖြစ်စေ application သည် အမြဲတမ်း ချောမွေ့စွာ အလုပ်လုပ်နေရမည်။

code များသည် အမြဲတမ်း ရှင်းလင်းစွာ စိတ်ပိုင်းပြီး ဖြစ်နေရမည်

တစ်ချက်တစ်ချက်တွင် develop လုပ်ခြင်းအပိုင်းကို ခေတ္တရပ်ကာ application ကို ပြန်လည်စိတ်ပိုင်းရမည်။ ဆိုလိုသည်မှာ ထပ်နေသော code များကို ရှင်းထုတ်ရမည်။ design ကို ပိုပြီး ကျစ်လစ်စေရမည်။ ဤသဘောတရား၏ ဦးတည်ချက်မှာ code ကိုသာ ပြန်သုံးရန် ခွဲထုတ်ရမည် ဖြစ်သည်၊ ယန္တရားကြီးကို ပုံစံပြင်ဆင်ခြင်း မဖြစ်ရ။ အဆင်ပြေသည့် အခြေအနေတိုင်းတွင် အစိတ်အပိုင်းငယ်များအဖြစ် ထပ်မံခွဲထုတ်ပြီး တစ်ခုနှင့်တစ်ခု ပေါင်းကူး ချိတ်ဆက်ရမည်။ အဆင့်တစ်ခုစီတိုင်းတွင်လည်း ထိုသို့ ခွဲထုတ်ပေါင်းကူးနိုင်စွမ်း ရှိစေမှုကို အမြဲ ထည့်သွင်းစဉ်းစားရမည်။ (ဤနမူနာမှသည် software development အတွက် စနစ်ကျသော လုပ်ထုံးလုပ်နည်းကောင်းများကိုလည်း တပါတည်း ရရှိပေမည်)

၅.၃ Project Design

အခန်း (၄) မှ ဆွေးနွေးချက်များအရ Android application တစ်ခုသည် activities များ၊ services များ၊ content provider များ၊ broadcast receiver များ ဆီလျော်သလို စုဝေးဖွဲ့စည်းနေကြောင်းကို သိခဲ့ပြီးဖြစ်သည်။ ယခုဖန်တီးမည့် Yamba application ကိုလည်း Android ၏ အခြေခံအုတ်မြစ် building block များဖြစ်သည့် ၄င်း component များကိုပင်လျှင် ပေါင်းစပ်တည်ဆောက်ရမည် ဖြစ်သည်။ Android ၏ အဓိက building block component များကိုသာ သုံးပြီး တည်ဆောက်မည့် Yamba application ကြီး တစ်ခုလုံး ၏ Design ကို တစ်ဖက်စာမျက်နှာပါပုံတွင် လေ့လာနိုင်ပါသည်။



Android application

ရေးသားဝန်တီးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

အခြေခံအဆင့် Android User Interface

ဤအပိုင်းသည် Yamba application ၏ ပထမ အစိတ်အပိုင်းဖြစ်သည့် (စာမျက်နှာ - ၇၇ တွင် ပြထားသော) Status Update Screen ကို ဖွဲ့စည်းတည်ဆောက်ခြင်းအား အဓိက ထားသည်။ လုပ်ဆောင်ရမည့် task များမှာ activity တစ်ခုကို တည်ဆောက်ခြင်း၊ ကွန်ယက်ဆက်သွယ်မှု networking ကို တည်ဆောက်ခြင်း၊ ခွဲဖြာလုပ်ဆောင်မှု multi threading ကို တည်ဆောက်ခြင်းနှင့် ချွတ်ချော်မှု ရှာဖွေရှင်းလင်းခြင်း debugging တို့ ဖြစ်သည်။

၅.၃.၁ Activity တစ်ခုကို တည်ဆောက်ခြင်း

Android User Interface ပုံစံ model တစ်ခု ကို စတင်မိတ်ဆက်ခြင်းဖြင့် အစပြုရပေမည်။ UI ပိုင်းတွင် Android သည် developer များ ယခင်က ကျွမ်းဝင်ခဲ့ပြီးသည့် အခြား အခင်းအကျင်းများနှင့် ကွာခြား၏။ ပြိုင်ဘက်ကင်းသော feature သည်ကား UI ကို Java မှလည်းကောင်း၊ XML မှလည်းကောင်း နှစ်နည်း ချဉ်းကပ်နိုင်ခြင်းပင်တည်း။

ဤအပိုင်းတွင် Status Update Screen အတွက် UI အား တည်ဆောက်ခြင်းကို ဆွေးနွေးပါမည်။ အဓိကအားဖြင့် XML နှင့် Java တို့ကို ပေါင်းစပ်ခြင်းအားဖြင့် အလုပ်ဖြစ်သော UI တစ်ခု ဖန်တီးပါမည်။ Layout များနှင့် View များအကြောင်း၊ Android ၏ unit များ၊ image များနှင့် အလုပ်လုပ်ပုံ၊ UI ကို အမြင်တင့်အောင် အလှဆင်ပုံတို့ကိုပါ တပါတည်း ဆွေးနွေးပါမည်။

အဓိကထားရမည်မှာကား Android မောင်းနှင်စနစ်သုံး မိုဘိုင်းကြိုးမဲ့ကိရိယာ၏ မြင်ကွင်း အရွယ်အစား screen size နှင့် ရုပ်ထွက်ပြင်းအား resolution မည်သို့ပင် ရှိစေ၊ မိမိ ၏ application တွင် အမြင်လည်းတင့်၍ အလုပ်လည်း ဖြစ်စေရမည့် UI မျိုးကို တည်ဆောက်မှု အလေ့အကျင့်ကောင်းများပင် ဖြစ်ပါသည်။

၅.၃.၂ ကွန်ယက်ဆက်သွယ်မှု Networking နှင့် ခွဲဖြာလုပ်ဆောင်မှု Multithreading

အလုပ်လုပ်နေသည့် screen တစ်ခု ရှိပြီ ဆိုသည်နှင့် တိမ်တိုက်ဝန်ဆောင်မှုဆီသို့ user ၏ input ကို post လုပ်ရပေတော့မည်။ (တိမ်တိုက်ဝန်ဆောင်မှု cloud service ဆိုသည်မှာ နေရာအသီးသီးရှိ user များက ပေးပို့လိုက်သော data များကို server ရှိရာဆီတွင် process လုပ်ပေးခြင်း ဖြစ်သည်။ အမှန်အားဖြင့် တိမ်တိုက်ဝန်ဆောင်မှု cloud service သည် client-server သဘောတရားကို အဝေးဆက်သွယ်ရယူစနစ် remote-access system သို့ ချဲ့ထွင်လိုက်ခြင်းပင် ဖြစ်သည်။) ဆိုခဲ့သည့်အတိုင်း အလုပ်လုပ်နေသည့် screen မှ တိမ်တိုက်ဝန်ဆောင်မှုဆီသို့ user ၏ input ကို post လုပ်ရန်အတွက် Twitter ၏ အသုံးချပရိုဂရမ်မျက်နှာပြင် Application Program Interface (API) မှ ပင့်ကူအိမ်ကွန်ယက်ဝန်ဆောင်မှု web service လုပ်ဆောင်ချက်များခေါ်ယူခြင်း call များ နှင့် ဆိုင်သော အကူအညီများ ရယူရန် (Android platfrom တွင် စကတည်းက မပါဝင်ဘဲ ကိုယ့်ဘာသာ သီးခြား ရယူရသည့်) third-party library တစ်ခုကို သုံးပါမည်။

Network calls ကွန်ယက်ခေါ်ဆိုမှုများ ပြုလုပ်ရာတွင် ကွန်ယက်၏ ကြိုတင်မခန့်မှန်းနိုင်သော အပြောင်းအလဲများသည့် သဘာဝကြောင့်၊ တနည်းဆိုရလျှင် connection အခြေအနေကြောင့်၊ UI သည် နှေးကွေးလေးလံနေသည်ကို သတိပြုမိတတ်ပါသည်။ network ၏ တုံ့ပြန်ရန်ကြန့်ကြာမှု latency သည် ရံဖန်ရံခါတွင် ကိုယ်ဖန်တီးသည့် application ကို တုံ့ပြန်မှု ရပ်ဆိုင်းသွားခြင်း (ကွန်ပျူတာသမားများအခေါ်အားဖြင့် hang သွားခြင်း) မျိုးပင် ဖြစ်စေတတ်ပါသည်။ ထိုအနေအထားသို့ ရောက်လျှင် Android ၏ ခွဲဖြာလုပ်ဆောင်မှု multithreading ဟူသော feature ကို စတင်ထိတွေ့ရပါမည်။ multithreading fetaure သည် Android platform ပေါ်၌ ပြင်ပအခြေအနေ အပြောင်းအလဲများအပေါ်တွင် ရာနှုန်းပြည့် တည်မှီနေခြင်းမျိုးမရှိဘဲ ကောင်းစွာ အလုပ်လုပ်သည့် application တစ်ခုကို ဖန်တီးတည်ဆောက်နိုင်စေပါသည်။

၅.၃.၃ Android application ကို ချွတ်ချော်မှု ရှာဖွေရှင်းလင်းရေး debug လုပ်ခြင်း

Design နှင့် လက်တွေ့အကောင်အထည်ဖော်ခြင်းသည် တစ်ကြိမ်တည်းဖြင့် ထပ်တူထပ်မျှ ဘယ်သောအခါမျှ မဖြစ်နိုင်ပါ။ application များ တည်ဆောက်ရာတွင် ချွတ်ယွင်းမှုရှာဖွေ ရှင်းလင်းခြင်း debugging သည် မပါမဖြစ်ကိစ္စတစ်ရပ် ဖြစ်သည်။ Android SDK ၏ tool များကို သုံးကာ ပြဿနာများကို လွယ်ကူလျှင်မြန်စွာ ရှာဖွေတွေ့ရှိစေမှုနှင့် ရှင်းလင်းမှုကို သိစေပါမည်။ Developer တစ်ယောက်အတွက် debugging သည် အကျင့်ပါနေသော ကိစ္စ မလွဲမသွေ ဖြစ်ရပါမည်။

Android application

ရေးသားစနစ်တီးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

အခန်း – ၆

Android User Interface မိတ်ဆက်

Android application နမူနာ project ဖြစ်သော Yamba အတွက် ပထမဦးဆုံး Activity ကို ဖန်တီးရမည်။ ၄င်းအတွက် XML layout တစ်ခုကို တည်ဆောက်ရမည်။ ၄င်းတို့ကို Java နှင့် ချိတ်ဆက်ရမည်။ မြင်ကွင်း View များ (တနည်းအားဖြင့် widget များ) နှင့် အခင်းအကျင်း Layout များ အကြောင်းကို လေ့လာရမည်။ ခလုပ်တစ်ခုကို နှိပ်ခြင်း button clicks ကဲ့သို့သော event များကို Java ပရိုဂရမ်ဘာသာစကားဖြင့် မည်သို့ ထိန်းချုပ်မည်ကို လေ့လာရမည်။

၆.၁ UI ကို ဖန်တီးသော နည်းနှစ်သွယ်

Android တွင် user interface (UI) တစ်ခုကို ဖန်တီးရန် နည်းနှစ်သွယ် ရှိသည်။ တစ်နည်းမှာ declarative နည်း ဖြစ်ပြီး၊ ကျန်တစ်နည်းမှာ programmatic နည်း ဖြစ်သည်။ ၄င်းတို့နှစ်ခုသည် တစ်ခုနှင့် တစ်ခု များစွာ ကွဲလွဲသည်။ သို့သော် အလုပ်တစ်ခုကို ပြီးမြောက်စေရေးတွင် နှစ်နည်းစလုံးကို အတူတကွ ရောယှက်သုံးစွဲလေ့ ရှိတတ်သည်။

Declarative UI ဆိုသည်မှာ

Declarative နည်း ဆိုသည်မှာ HTML ဖြင့် web page တစ်ခုကို တည်ဆောက်သည့်နည်းနှင့် တူသည်။ ပုံပေါ်စေချင်သော UI အား XML သုံးကာ declare လုပ်ရခြင်း ဖြစ်သည်။ screen ပေါ်တွင် ပေါ်လာစေလိုသော element များအတွက် tag များကို ရေးသားရသော HTML page တစ်ခုကို အစအဆုံး လုပ်ဖူးလျှင် Android screen ပေါ်တွင် UI ဖော်ရာ၌

လည်း အတူတူပင်ဖြစ်ကြောင်း လက်တွေ့ သိရပါလိမ့်မည်။

Declarative နည်း၏ အားသာချက်တစ်ရပ်မှာ အမြင်အတိုင်းပုံပေါ်မည့် (what you see is what you get WYSIWYG - *ဝိုင်ဆီဝ့ါဂ်* ဟု အသံထွက်သည်) WYSIWYG tool များကို အသုံးပြုနိုင်ခြင်း ဖြစ်သည်။ ဤ tool အချို့သည် Eclipse Android Development Tools (ADT) ၏ တိုးချဲ့အုပ်စု extension တွင် ပါဝင်သည်။ Android မှ ထုတ်လုပ်ခြင်း မဟုတ်သည့် အခြားထုတ်လုပ်မှု Third-parties tools များလည်း ပေါပါသည်။ ထိုမျှမကသေး၊ XML သည် လူတစ်ယောက်အဖို့ မြင်လျှင် အခက်အခဲများစွာမရှိပဲ နားလည်သည့် ပရိုဂရမ်ဘာသာစကားမျိုး ဖြစ်သောကြောင့် Android ၏ platform ၊ framework တို့နှင့် စိမ်းသောသူပင်လျှင် UI ပေါ်တွင် ပါရှိ လာမည့် အရာများအား (XML code ကို ကြည့်ခြင်းဖြင့်) မှန်းဆသိရှိနိုင်ပါသည်။

Declarative နည်း၏ အားနည်းချက်မှာကား XML ၏ အကန့်အသတ်ပင်။ XML သည် user interface (UI) ၏ အထွက်မြင်ကွင်းနှင့် ကြည့်သူအပေါ် သက်ရောက်မှုကို ရက်လုပ်ရာတွင် ကောင်းသည် မှန်သော်လည်း၊ user ၏ input ကို ထိန်းကျောင်းရာတွင်မူ နည်းလမ်းကောင်းတစ်ခု မဖြစ်ချေ။

Programmatic UI ဆိုသည်မှာ

Programmatic နည်းလမ်းသည်ကား UI တစ်ခုကို တည်ဆောက်ဖန်တီးရန် အတွက် Java code များ ရေးသားရခြင်း ဖြစ်ပေသည်။ ၄င်းသည် အခြားသော ပရိုဂရမ်ဘာသာစကားများ ရှိ UI Toolkit များနှင့် သိပ်မကွာလှပါ။

အခြေခံအားဖြင့်ဆိုရလျှင် ခလုပ်တစ်ခုကို programmatic နည်းဖြင့် ဖန်တီးတည်ဆောက်လို လျှင် button အတွက် variable ကို ကြေငြာ declare လုပ်ရမည်။ ပြီးလျှင် ၄င်း variable ကို container တစ်ခုခုထဲသို့ ပို့ရမည်။ အရောင်၊ စာသား၊ စာသားအရွယ်အစား၊ နောက်ခံ

အရောင် စသည့် ဤ button ၏ properties များကို သတ်မှတ် set လုပ်ရမည်။ ထို့ပြင် ၄င်းခလုပ်ကို နှိပ်လျှင် ဖြစ်မည့် event / လုပ်မည့် action ကို ကြေငြာ declare လုပ်ရမည်။ ဤသည်လည်း နောက်ထပ် Java code တစ်စု ဖြစ်လေမည်။ ပြောရလျှင် Java ဖြင့် code အတော်များများ ရေးလိုက်ရခြင်း ဖြစ်ပေသည်။

Declarative နည်းဖြင့် လုပ်၍ ဖြစ်သမျှတိုင်းကို Programmatic နည်းဖြင့်လည်း လုပ်နိုင်ပေသည်။ သို့သော် programmatic နည်း (Java ဖြင့် code ရေးခြင်း) က ဤခလုပ်သည် နှိပ်လိုက်လျှင် မည်သို့ဖြစ်သွားမည် ဆိုသည့်အပိုင်းကိုလည်း အတိအကျသတ်မှတ် specify လုပ်ခွင့်ရစေသည်။ ဤသည်ပင်လျှင် programmatic နည်း၏ UI အတွက် အဓိက အားသာချက် ဖြစ်၏။

သို့ဆိုလျှင် မည်သည့်နည်းလမ်းကို သုံးမည်နည်း။ အကောင်းဆုံးမှာ နှစ်နည်းစလုံးကို သုံးခြင်း ဖြစ်၏။ (Screen ၏ layout နှင့် widget အားလုံး ကဲ့သို့သော) UI ၏ အပြောင်းအလဲမရှိသည့် အရာများ အတွက် declarative နည်းဖြင့် XML ဖန်တီးလိုက်မည်။ ထို့နောက် User Interface object များ၏ user နှင့် အပြန်အလှန် interactive အလုပ်လုပ်မည် ဖြစ်သည့် ကိစ္စများကို programmatic နည်းဖြင့် Java code ရေးမည်။ ဆိုလိုသည်မှာ button တစ်ခု၏ အသွင်အပြင်ကို XML ဖြင့် ပုံဖော်ပြီး လုပ်ဆောင်ချက်ကို Java ဖြင့် ယန္တရားဆင်ယူရပေမည်။

ပြောရန် တစ်ချက် ရှိသည်မှာကား XML ဆိုသည်မှာလည်း အကယ်စင်စစ်မှာ Java code အဖြစ်သို့ ပြောင်းလဲသွားသည်သာ ဖြစ်သဖြင့် တကယ် run သည်မှာ Java code သာ ဖြစ်ပါသည်။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

၆.၂ View များ နှင့် Layout များ

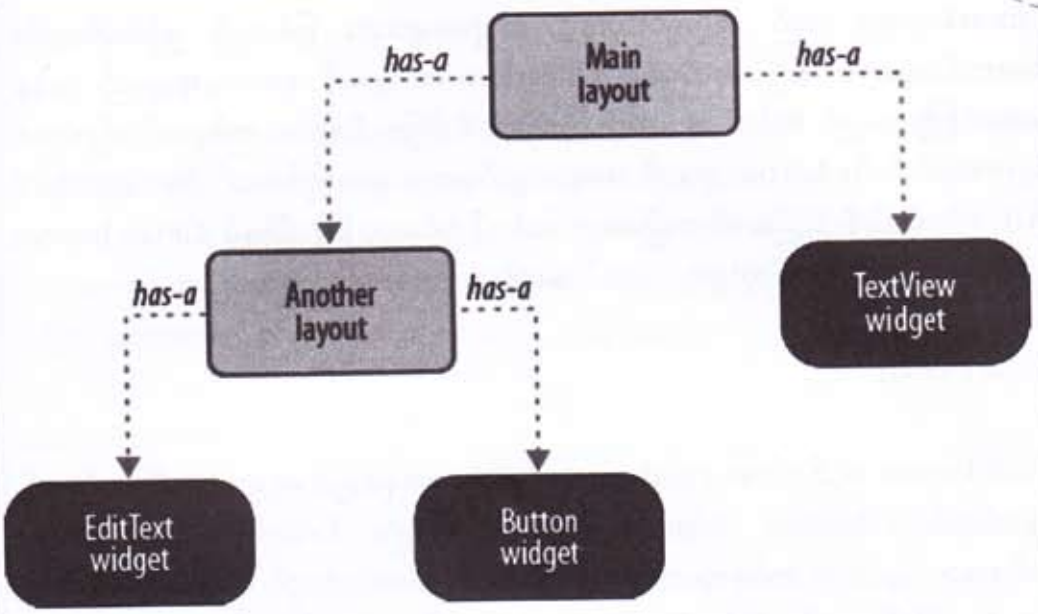
Android သည် user interface (UI) ၏ object များကို Views များနှင့် Layouts များအဖြစ် သီးခြားစီ ခွဲထုတ်စုစည်းသည်။ မြင်ရသမျှ button များ၊ label များ၊ text box များ အားလုံးသည် view များ (တနည်းအားဖြင့် widget များ) ဖြစ်၏။ Layout များက view များကို စုစည်းပေးသည်။ ဥပမာ ဆိုရလျှင် button တစ်ခုနှင့် label တစ်ခုကို group ဖွဲ့ပေးခြင်းမျိုး။

View များသည် Java ၏ component များနှင့် အလားတူပြီး၊ Layout များသည် Java ၏ container များနှင် အလားတူသည်။

view ကို ဖလှယ်သုံးစွဲသော widget နှင့် App Widget များမှာ တစ်ခုနှင့်တစ်ခု မတူပါ။ App Widget ဆိုသည်မှာ (Home Screen (App) ကဲ့သို့သော) Applicaton ကြီးများ ထဲတွင် ငုံလျက်တည်ရှိသည့် app view များ (တနည်းအားဖြင့် icon များ) ကို ဆိုလိုခြင်း ဖြစ်ပါသည်။

Layout များသည် View များကို ငုံထားသည် ဖြစ်နိုင်သကဲ့သို့၊ Layout များသည် Layout များကို ငုံထားသည်လည်း ဖြစ်နိုင်ပြန်ပါသည်။ ဒုတိယအနေအထားတွင် user interface (UI) ၏ ဖွဲ့စည်းပုံသည် ရိုးစင်းသော အနေအထားမရှိတော့ပဲ ရှုပ်ထွေးအစုအဝေး complex ဖြစ်သွားမည်။

Layout သည် ၄င်းထဲတွင် ပါဝင်သည့် child (widget/view/layout) တစ်ခုစီအတွက် နေရာစီစဉ်ဖန်တီးပေးရသည်။ Layout အသီးသီးသည် ၄င်းတို့၏ child အသီးသီးအား အခင်းအကျင်း စီစဉ်ဖန်တီးပေးပုံကို ရှေ့စာမျက်နှာရှိ သရုပ်ပြပုံတွင် တွေ့နိုင်ပါသည်။



မကြာခဏ အသုံးပြုရလေ့ရှိသော ပင်မ layout (function) များမှာ LinearLayout ၊ TableLayout ၊ FrameLayout ၊ RelativeLayout နှင့် AbsoluteLayout တို့ ဖြစ်သည်။

LinearLayout

LinearLayout သည် အရိုးရှင်းဆုံးနှင့် အသုံးအများဆုံး layout တစ်ခု ဖြစ်သည်။ ၄င်းသည် child များကို horizontally ဖြစ်စေ၊ vertically ဖြစ်စေ တစ်ခုပြီး တစ်ခု တန်းစီကာ နေရာချသည်။ အရေးကြီးသည်မှာ child တို့၏ အစီအစဉ် ဖြစ်သည်။ အကြီးဆုံးက ဦးဆုံး ဝင်လာကာ နေရာယူလျှင် ဆက်လာမည့် ကျန်သော child များအတွက် screen ပေါ်တွင် နေရာမကျန်လှတော့ချေ။

LinearLayout ၏ အရေးပါသော property တစ်ခုမှာ layout_orientation ဖြစ်သည်။ ရွေးရန်ရှိသည်တို့မှာ horizontal နှင့် vertical တို့ ဖြစ်သည်။

LinearLayout သည် အရိုးရှင်းဆုံးနှင့် အသုံးအများဆုံး ဖြစ်သည် မှန်သော်လည်း အကောင်းဆုံးကား အမြဲမဟုတ်ချေ။ LinearLayout များကို စုဝေးနေရာချမည့် အနေအထား ကြိုလာလျှင် RelativeLayout ကဲ့သို့သော အခြား Layout တစ်ခုခုကို ပြောင်းလဲစဉ်းစားသင့်သည်။ Layout များကို အရေအတွက်များစွာ စုဝေးခြင်းသည် user interface (UI) ကြီးကို ပြီးစီးသည်အထိ တည်ဆောက်ရန် လိုအပ်သော ကြာချိန်နှင့် CPU ၊ battery စသော resource များ သုံးစွဲရမှု အပေါ် အကျိုးဆက်များ ဖြစ်လာစေမည်။

TableLayout

TableLayout သည် ၄င်း၏ child များအား table တစ်ခုအဖြစ် နေရာချပေးပြီး၊ ၄င်းတွင် နောက်ထပ် TableRow widget တစ်ခုသာ ပါရှိသည်။ TableRow သည် Table တစ်ခုအတွင်းမှ row တစ်ခုကို ကိုယ်စားပြုပြီး TableRow ထဲတွင် အခြား UI widget များကို သိမ်းနိုင်သည်။ TableRow widget များကို တစ်ခုနှင့် တစ်ခု horizontally ဘေးချင်းယှဉ်လျက် ခင်းကျင်းသည်။ LinearLayout ကို horizontal orientation ဖြင့် ထားသည်နှင့် တူ၏။

HTML နှင့် နှိုင်းယှဉ်ရှင်းလင်းရလျှင် TableLayout သည် <table> နှင့် တူပြီး၊ TableRow သည် <tr> နှင့် တူသည်။ HTML တွင် table ထဲရှိ cell တစ်ခုချင်းစီအတွက် <td> ရှိသည်၊ Android တွင်မူ TableRow ထဲသို့ ဖြည့်လိုက်သော view (widget) များ၏ အရေအတွက်ကိုလိုက်ပြီး column ကို အလိုလျောက် သတ်မှတ်သည်။ cell တစ်ခုချင်းစီကို သီးခြား ခွဲထုတ်မနေတော့ချေ။

TableLayout ၏ အရေးပါသော property တစ်ခုမှာ stretch_columns ဖြစ်သည်။ table ၏ မည်သည့် column အား ကျုံ့/ဆန့်ယူ stretch လုပ်မည်ကို စေ့ညွှန်းနိုင်သည်။ column အားလုံးကို stretch လုပ်မည်ဆိုလျှင် * ကို သုံးနိုင်သည်။

FrameLayout

FrameLayout သည် ကဒ်ပြားများကို စီ ထပ်သကဲ့သို့ နောက် child က ရှေ့ child များ ကို အုပ်သွားစေပြီး နေရာချသည်။ နောက်ဆုံး child သည် အပေါ်ဆုံးတွင် ရှိသည်။ ဤ layout သည် tab များရှိသော view မျိုးတွင် အသုံးတည့်သည်။ ထို့ပြင် FrameLayout ကို စ run စဉ်က မရှိဘဲ programmatic နည်းဖြင့် တဖြည်းဖြည်း ဖြည့်ထည့်ရန်ရှိလာမည့် widget များအတွက် container အဖြစ်လည်း သုံးနိုင်သည်။

RelativeLayout

RelativeLayout သည် တစ်ခုနှင့်တစ်ခု ဆက်စပ်တည်မှီ relative ဖြစ်နေသော child များအတွက် အဆင်ပြေသည်။ အမြင်အနေအထားတစ်ခုကို ဖြစ်ပေါ်စေရာတွင် မလိုအပ်သော layout များကို nest လုပ်ထားစရာ မလိုသောကြောင့် ၄င်းသည် အလွန် powerful ဖြစ်၏။ တနည်းဆိုရလျှင် ခေါ်ယူရန်လိုအပ်သော widget အရေအတွက်ကို အနည်းဆုံးဖြစ်အောင် လျှော့ချထားနိုင်သည်၊ သို့ဖြစ်၍ ကိုယ့် application ၏ တစ်ခုလုံး စွမ်းဆောင်ရည်ကို ပိုမိုကောင်းမွန်လာစေသည်။ ထိုသို့ ဖြစ်စေမှုအတွက် RelativeLayout တွင် child widget တစ်ခုစီတိုင်းအတွက် အခြား child များနှင့် ဆက်စပ် relate လုပ်နိုင် စေခြင်းငှာ ID set တစ်ခုစီ ရှိနေရန် လိုအပ်သည်။

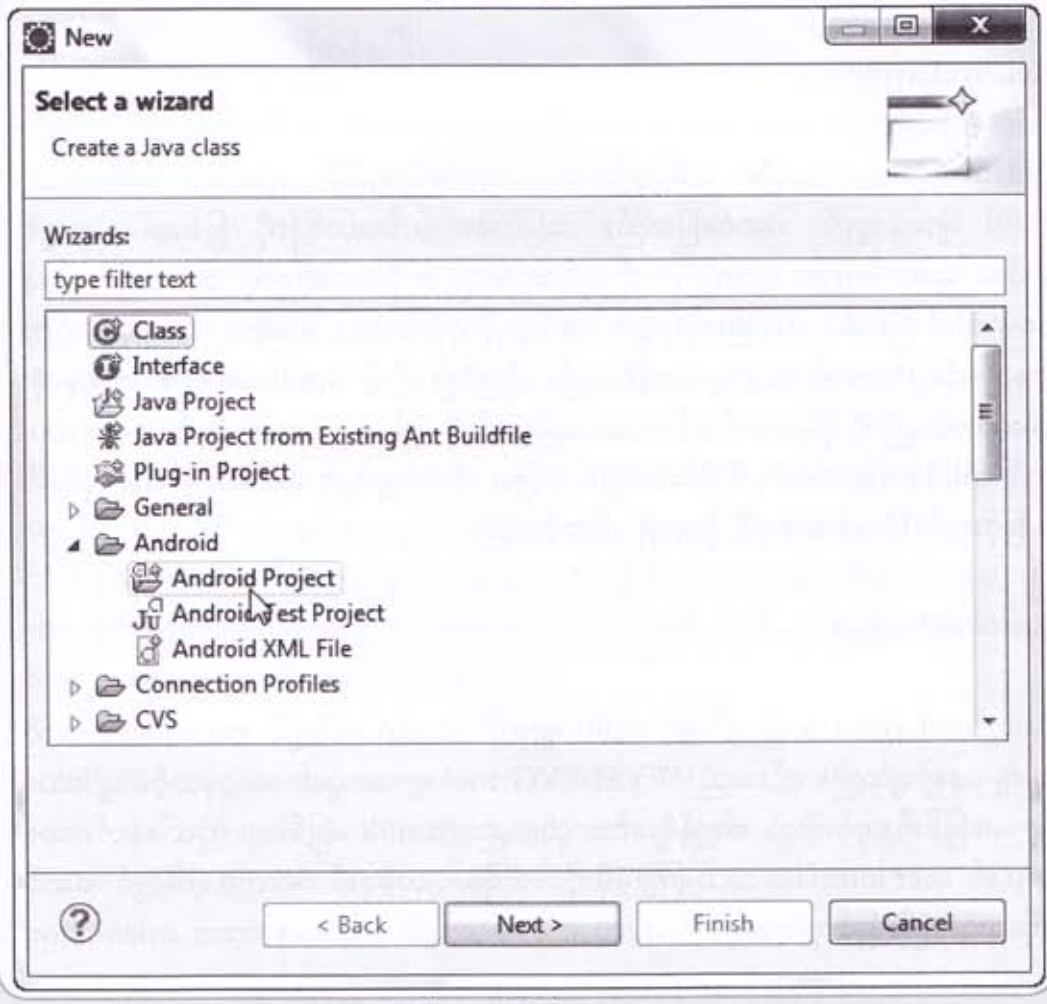
AbsoluteLayout

AbsoluteLayout သည် ၄င်း၏ child များကို screen ပေါ်တွင် coordinate အတိ အကျ နေရာချသည်။ ၄င်းသည် WYSIWYG tool များအတွက် အလွန်သင့်လျော်သော layout ဖြစ်၏။ ၄င်းသည် အလွန်ရိုးစင်းသော်လည်း flexible မဖြစ်ချေ။ ကိုယ်ဖန်တီးသော app ၏ user interface သည် ကြိုးမဲ့မိုဘိုင်းကိရိယာ တစ်ခု၏ Screen ပေါ်တွင် အဆင် ပြေသော်လည်း မြင်ကွင်းအရွယ် screen size ၊ အထောင်အလဲ့အနေအထား orientation

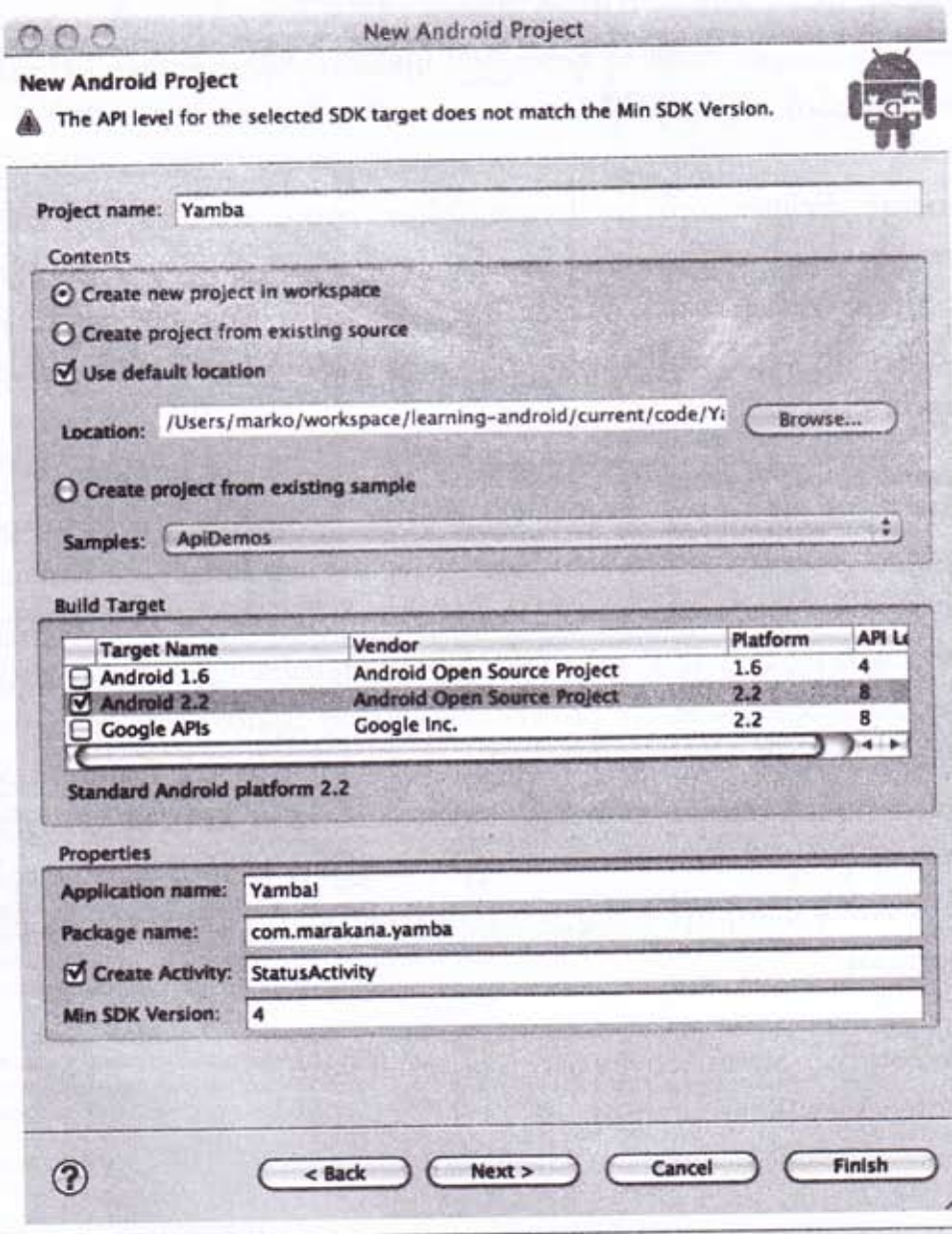
ရှပ်ထွက်ပြင်းအား resolution တို့ ပြောင်းလဲသွားသည်နှင့် အနေအထား ပျက်ယွင်းသွား ပေမည်။

၆.၃ Yamba Project ကို စတင်တည်ဆောက်ခြင်း

Yamba project ကို စတော့မည်။ Eclipse ကို ဖွင့်မည်။ File > New > Other > Android > Andriod Project ကို ရွေးပါမည်။



New Android Project (dialog window) တွင် အောက်ပါ ပုံအတိုင်း လိုအပ်သည်များကို ဖြည့်သွင်းပါမည်။



ဝင်းဝင်း
ဝင်းဟန်
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

၆.၄ Status Activity Layout ကို စတင်တည်ဆောက်ခြင်း

Screen ၏ user interface (UI) ကို design ချခြင်းဖြင့် စတင်ရပါမည်။ ပထမဆုံး screen သည် user က status ကို ရိုက်ထည့်ပြီး update လုပ်နိုင်သော Status Update screen ဖြစ်မည်ကို ဆွေးနွေးခဲ့ပါပြီ။

ပုံမှန်အားဖြင့် Eclipse သည် res/layout (folder) ထဲတွင် main.xml ဟု အမည်ရှိသော ဖိုင်တစ်ခုကို တည်ဆောက်လိုက်သည်။ ဖိုင်တို့၏ အမည်နှင့် activity တို့၏ အမည်တို့သည် အမြဲတသမတ်တည်း ဖြစ်နေစေရန်လိုအပ်သဖြင့် ယခု activity ဖြစ်သည့် Status Activity နှင့် ကိုက်ညီပြီး တသမတ်ဖြစ်စေရန် ဤဖိုင်ကို status.xml ဟု အမည်ပြောင်းလိုက်ပါမည်။

Eclipse ထဲတွင် ဖိုင်တစ်ခုကို အမည်ပြောင်း rename လုပ်ရန်အတွက် ထိုဖိုင်ပေါ်တွင် right click လုပ်ပါ။ ထို့နောက် ပေါ်လာသော pop-up menu မှ Refactor > Rename ကို click ရပါမည်။ ထို့နောက် နာမည်သစ်ကို ရိုက်ထည့်လိုက်ပါမည်။ Eclipse သည် လုပ်ဆောင်ချက် မြင့်သော IDE တစ်ခု ဖြစ်သဖြင့် ယခု rename လုပ်ခြင်းသည် အမည်ပြောင်းသွားရုံသက်သက် မ'က'ပါ။ project ထဲရှိ ဤဖိုင်ကို ညွှန်းထားသော reference အားလုံးကို လိုက်ရှာပြီး တွေ့သမျှကိုပါ update လုပ်ပေးပါသည်။ ဤ feature သည် Java file တစ်ဖိုင်ကို rename လုပ်ရာတွင် အဆင်ပြေသော်လည်း၊ XML file များတွင်မူ အကြင်းမဲ့ အလိုလျောက် fully automatic မဖြစ်ပါ။ သို့ဖြစ်၍ ဤဖိုင်ကို rename လုပ်ရာတွင် R class မှ ၄င်းကို ညွှန်းခဲ့သည့် Java code စာကြောင်းကိုမူ သွားရောက်ပြင်ဆင်ဖို့ လိုလာပါသည်။

သို့ဖြစ်သောကြောင့် StatusActivity ထဲရှိ OnCreate() တွင်
setContentView(R.layout.main); ကို
setContentView(R.layout.status); ဟု ပြင်ရပါမည်။

ဤ Screen တွင် ပါဝင်သော component လေးခု ရှိပါမည်။

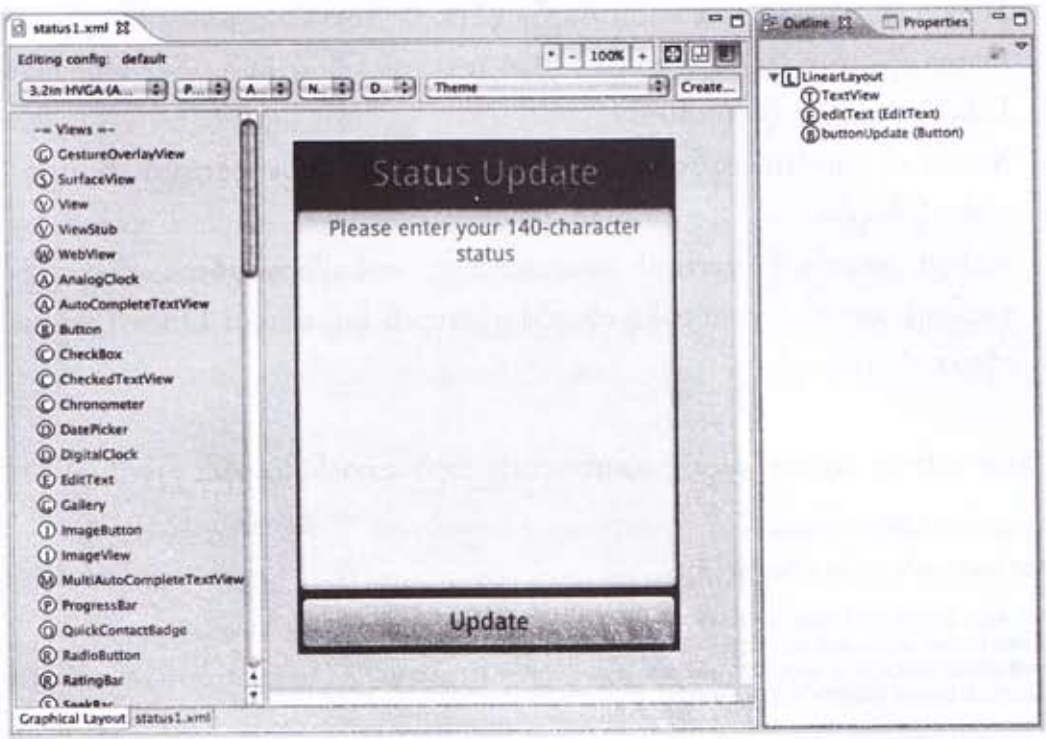
- Screen ၏ ထိပ်တွင် title တစ်ခု ရှိမည်။ ၄င်းသည် TextView *widget* ဖြစ်မည်။
- Status ကို ရိုက်ထည့်နိုင်မည့် စာလုံး 140 ဆံ့ text area ကြီး တစ်ခုရှိမည်။ ဤအတွက် EditText *widget* ကို သုံးပါမည်။
- Status ကို update လုပ်ရန် ခလုပ် button တစ်ခုပါမည်။ ၄င်းသည် Button *widget* ဖြစ်မည်။
- widget အားလုံးကို vertical အနေအထားဖြင့် တစ်ခုပြီးတစ်ခုစီကာ နေရာချ၍ စုစည်းရန် အတွက် layout တစ်ခု လိုမည်။ ဤအတွက် ရိုးရိုးစင်းစင်း LinearLayout ကိုသာ သုံးလိုက်မည်။

Status activity layout အတွက် source code သည် အောက်ပါအတိုင်း ဖြစ်ပါမည်။

Example 6-1. res/layout/status.xml

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

<!-- Main Layout of Status Activity -->
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical" android:layout_width="fill_parent"
  android:layout_height="fill_parent">

  <!-- Title TextView-->
  <TextView android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:gravity="center"
    android:textSize="30sp"
    android:layout_margin="10dp" android:text="@string/titleStatus"/>

  <!-- Status EditText  -->
  <EditText android:layout_width="fill_parent"
    android:layout_height="fill_parent" android:layout_weight="1"
    android:hint="@string/hintText" android:id="@+id/editText"
    android:gravity="top|center_horizontal"></EditText>

  <!-- Update Button -->
  <Button android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:text="@string/buttonUpdate"
    android:textSize="20sp" android:id="@+id/buttonUpdate"></Button>

</LinearLayout>
```

ဤ code ကို Eclipse ၏ Graphical Layout မှ ထုတ်ပေးလိုက်ခြင်း ဖြစ်သည်။ Eclipse Graphical Layout ကို အောက်ပါပုံတွင် ကြည့်ပါ။



ဤ XML resource များ၏ အဓိပ္ပာယ်များနှင့် ပတ်သက်ပြီး ရှေ့အခန်းတွင် အခြေခံပိုင်းကို ဆွေးနွေးခဲ့ကြပြီး ဖြစ်သော်လည်း သိသင့်သော အသေးစိတ်အကြောင်းအရာများ ရှိသေး၏။

၆.၅ Widget တို့၏ အရေးကြီးသော properties များ

အမြဲသုံးဖြစ်နေမည့် properties များမှာ

layout_height နှင့် layout_width

ငင်းတို့သည် ယခု widget ကို ဝိုထားသော Parent Layout ကြီးထဲမှ ယခု widget က နေရာ မည်မျှတောင်းခံမည်နည်း ဆိုသည်ကို သတ်မှတ်သည်။ တန်ဖိုးတစ်ခုကို pixels ဖြင့် လည်းကောင်း၊ inches ဖြင့်လည်းကောင်း၊ အခြား အလားတူယူနစ်မျိုးဖြင့် လည်းကောင်း ထည့်နိုင်သော်လည်း ငင်းသည် တကယ်တမ်းတွင် အသုံးချသင့်သော အလေ့အကျင့်မဟုတ်ပါ။ ကိုယ့် application သည် screen အရွယ်အစားအမျိုးမျိုးရှိသည့် မိုဘိုင်းကြိုးမဲ့ကိရိယာများတွင် အဆင်ပြေစွာ run နိုင်ရေးအတွက်၊ component များ၏ အလျားနှင့် အနံကို ပုံသေကားချအရွယ်အစား absolute size များ သတ်မှတ်ခြင်းမပြုဘဲ နှိုင်းရအရွယ်အစား relative size များ သတ်မှတ်သင့်ပါသည်။ သို့ဖြစ်၍ ဤတန်ဖိုးများအတွက် အကောင်းဆုံး အသုံးပြုသင့်သည်မှာ fill_parent သို့မဟုတ် wrap_content ဖြစ်ပါသည်။ fill_parent ဆိုသည်မှာ ဤ widget သည် Parent Layout ထဲမှ ရနိုင်သမျှအကျယ်အဝန်းကို တောင်းဆိုသည်။ wrap_content ဆိုသည်မှာ widget သည် ငင်း၏ content ကို ခင်းကျင်းရန် လိုအပ်သလောက် အကျယ်အဝန်းကိုသာ တောင်းဆိုသည်။ သတိပြုရန် တစ်ချက်မှာ API level 8 နှင့် အထက်တွင် fill_parent ကို match_parent ဟု ပြောင်းလဲထားခြင်း ဖြစ်ပါသည်။

layout_weight

layout_weight သည် 0 နှင့် 1 ကိန်းနှစ်ခုမှ တစ်ခုဖြစ်သည်။ ဥပမာဖြင့် ရှင်းလင်းရလျှင် Status EditText ၏ layout_weight *default* တန်ဖိုးသည် 0 ဖြစ်သည်။ layout_height မှာ fill_parent ကို တောင်းဆိုသည်။ ဤသို့ဆိုလျှင် Update button သည် Screen ထဲမှ အပြင်သို့ တွန်းထုတ်လိုက်ခြင်း ခံရပေတော့မည်။ သို့သော် Status EditText ၏ layout_weight ကို 1 ထားလိုက်လျှင်မူ အဓိပ္ပာယ်မှာ ဤ widget သည် height

အားဖြင့် ရနိုင်သမျှ အကျယ်အဝန်းပြည့်ကို လိုချင်သည်၊ သို့သော် အခြား widget အတွက် (ယခုအနေအထားအရ ဆိုလျှင် Update button အတွက်) လိုအပ်သော အကျယ်အဝန်းကိုမူ မယူပါ၊ ဟု အဓိပ္ပာယ်ရပေမည်။

layout_gravity

layout_gravity မှာ ဤ Layout အတွင်း၌ ယခု widget အား မည်သို့မည်ပုံ နေရာချမည်ကို ပြဋ္ဌာန်းသည်။ တန်ဖိုးများသည် top / center / left စသည်တို့ ဖြစ်နိုင်သည်။ (ဤ property နှင့် အောက်တွင်ရှင်းလင်းထားသည့် gravity ဟူသော property တို့၏ ကွဲပြားမှုကို သတိပြုစေချင်ပါသည်။) ဥပမာ ဆိုရလျှင် width ကို fill_parent အဖြစ် သတ်မှတ် set လုပ်ထားသော widget တစ်ခု ရှိသည်ဆိုပါက ၄င်းကို center ပို့ခြင်းသည် ဘာမျှ ထူးပြီး ဖြစ်လာမည် မဟုတ်ပါ။ စကတည်းက ၄င်း widget သည် Parent Layout ထဲတွင် အလျားလိုက်အားဖြင့် အံ့ကျ ဝင်နေပြီးလေပြီ။ သို့သော် Title Text *view* ဟူသော *widget* ကို width အား wrap_content အဖြစ် သတ်မှတ် set လုပ်ထားလျှင်မူ layout_gravity ဖြင့် center ထားလိုက်ခြင်းသည် လိုချင်သောရလဒ်ကို ဖြစ်စေနိုင်ပေသည်။

gravity

gravity မှာကား widget တစ်ခုထဲရှိ content ကို ၄င်း widget ထဲတွင် နေရာချခြင်းအား ပြဋ္ဌာန်းသည်။ ဤ property ကို အထက်တွင်ရှင်းလင်းခဲ့သည့် layout_gravity ဟူသော property နှင့် ရောထွေးလေ့ရှိသည်။ မည်သည်ကို သုံးသင့်သနည်း ဟူသည်ကား မိမိသုံးသည့် widget ၏ အရွယ်အစားနှင့် လိုချင်သော အထွက်မြင်ကွင်း အပေါ်တွင်သာ တည်ပေသည်။ ဥပမာ ဆိုရလျှင် Title TextView တွင် width အားဖြင့် fill_parent လုပ်ထားခဲ့ပါက ၄င်းကို gravity ဖြင့် အလယ် center ချိန်လျှင် အလုပ်ဖြစ်ပေမည်။ layout_gravity ဖြင့် အလယ် center ချိန်လျှင်ကား ဘာမျှ ထူးလာမည်မဟုတ်ချေ။

text

widget အားလုံးတွင်တော့ ဤ property မရှိပါ။ သို့သော် Button ၊ EditText နှင့်

TextView တို့ကဲ့သို့သော widget အတော်များများတွင် ရှိပါသည်။ ဤ property သည် widget ပေါ်မှ စာသားကို ပြဋ္ဌာန်းသည်။ သို့သော် စာသားသက်သက်ကို အစကတည်းက လွယ်လင့်တကူ ရိုက်ထည့်ထားလိုက်ခြင်းသည် အလေ့အထကောင်း မဟုတ်ပါ။ ဤသို့ဆိုလျှင် ဤ widget ၏ စာသားသည် အမြဲ တသမတ်တည်းကြီး အပြောင်းအလဲမရှိ ဖြစ်နေပေလိမ့်မည်။ လုပ်သင့်သော အလေ့အထမှာ စာသား text အားလုံးကို res (folder) ထဲ၌ strings.xml ဟု ဖိုင်တစ်ဖိုင် ဆောက်ကာ သတ်မှတ် define လုပ်၍ သိမ်းထားပြီး၊ လိုအပ်လျှင် ပြန်၍ refer လုပ်ခြင်း ဖြစ်ပါသည်။
refer လုပ်ပုံမှာ နမူနာအားဖြင့် @string/titleStatusUpdate ကဲ့သို့ ဖြစ်ပါသည်။

id
id သည် ယခု widget ၏ သက်ဆိုင်ရာ layout resource file ထဲတွင် ရှိသည့် (တခြားမည်သည့်တစ်ခုနှင့်မျှ မထပ် မတူသော) identifier ဖြစ်ပါသည်။ widget တိုင်းသည် id မလိုပါ။ ဖြစ်နိုင်လျှင် မလိုအပ်သော id များကို ဖယ်ရှားပစ်ခြင်းဖြင့် မွစာကျဲနေသည်များကို ကျစ်လစ် သိပ်သည်းစေသင့်ပါသည်။ သို့သော် နောင်တွင် Java မှ လှမ်းကာ ထိန်းချုပ်စီမံရမည့် widget များအတွက်ကား id လိုအပ်ပါသည်။ id ၏ ရေးထုံးပုံစံမှာ @+id/someName ဖြစ်ပါသည်။ someName နေရာတွင် ကိုယ်ခေါ်ယူလိုသော widget ကို ပြောင်းလဲလိုက်ရုံပင်။ ဥပမာ – @+id/buttonUpdateStatus

၆.၆ String Resource များ

Adroid သည် data တို့ကို ဖိုင်များ ခွဲနိုင်သမျှ ခွဲပြီး သိမ်းမှုကို အထူးလိုလားသည်။ သို့ဖြစ်၍ layout တို့ကို ၄င်းတို့၏ သက်ဆိုင်ရာ resource များထဲတွင် သတ်မှတ် define လုပ်ပြီး၊ (button တို့၏ text များ၊ title တို့၏ text များ စသည့်) စာသားတန်ဖိုး text value အားလုံးကိုလည်း strings.xml ဟူသော ကိုယ်ပိုင်သီးခြား ဖိုင်တစ်ခုထဲတွင် define လုပ်သည်။ ဤသို့ လုပ်ပေးခြင်းဖြင့် နောင်တွင် string resource များကို English ၊ Japan ၊ Myanmar ကဲ့သို့ ဘာသာစကား များစွာဖြင့် version များ ခွဲထုတ်နိုင်စေသည်။

လက်ရှိ အခြေအနေတွင် strings.xml ဖိုင်သည် အောက်ပါအတိုင်း ရှိပေလိမ့်မည်။

Example 6-2. res/values/strings.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
  <string name="app_name">Yamba 1</string>

  <string name="titleYamba">Yamba</string>
  <string name="titleStatus">Status Update</string>

  <string name="hintText">Please enter your 140-character status</string>
  <string name="buttonUpdate">Update</string>
</resources>
```

resources များကို အမည်ပေးရာတွင် စံထားအသုံးပြုသော အမည်ပေးစနစ်မှာ 'ရှေ့ဆုံးတွင် ထို resource ၏ အမျိုးအမည်၊ ထို့နောက် ကိုယ်ပိုင်အမည်' ပုံစံ ဖြစ်သည်။ titleYamba ဆိုသည်မှာ ရှေ့မှ စကားလုံး title သည် activity ၏ အမျိုးအမည် ဖြစ်သော title ကို ဆိုလိုပြီး၊ နောက်မှ စကားလုံး Yamba မှာ ဤ title ၏ အမည် ဖြစ်သည်။ ဤစနစ်ကို လိုက်နာ အသုံးချခြင်းဖြင့် code ကို မြင်သည်နှင့် resource များကို အမျိုးအမည်၊ ကိုယ်ပိုင် အမည်တို့အားဖြင့် ရှင်းလင်းစွာ ခွဲခြားသိရှိနိုင်ပြီး trace လိုက်ရန် အဆင်ပြေစေသည်။

၆.၇ StatusActivity ဟူသော Java Class အပိုင်း

ယခုဆိုလျှင် user interface (UI) ကို XML ဖိုင်တွင် design လုပ်ပြီး ဖြစ်သည့်အတွက် Java ဆီသို့ ကူးရမည်။ Android သည် UI ကို တည်ဆောက်ရန် နည်းနှစ်နည်းကို ထားရှိ ပေးသည် ဆိုခြင်းကို ဤအခန်း၏ ရှေ့ပိုင်းတွင် ဆွေးနွေးခဲ့လေပြီ။ ပထမနည်းသည် ယခု အသုံးချခဲ့သော XML Declarative နည်း ဖြစ်သည်။ ထိုနည်းကိုသုံး၍ UI ကို ဖန်တီးခဲ့ ရာ ယခုအဆင့်သို့ တိုင်ခဲ့ပြီ။ ထိုထက်ပို၍ ရှေ့မတိုးသာတော့ချေ။ သို့ဖြစ်၍ ယခုအခြေအနေ တွင် Java ဘက်သို့ မဖြစ်မနေ ကူးရပေမည်။ အကောင်းဆုံး အလေ့အထမှာလည်း XML နှင့် လုပ်ဆောင်ရန်ရှိသမျှ အကုန်ဖန်တီးပြီးနောက်မှ Java ဘက်သို့ ကူးခြင်း ဖြစ်သည်ပင်။

၆.၈ Application ကို ပြဋ္ဌာန်းသော Object များနှင့် Initialization Code ကို ဖန်တီးခြင်း

activity များ၊ service များ၊ broadcast receiver များ၊ content provider များ စသည့် Andriod ၏ မည်သည့် အဓိက building block ကို မဆို ကိုင်တွယ်စီမံထိန်းချုပ်ရာတွင် Android framework က ဖန်တီးပေးထားနှင့်သော အခြေခံ base class တစ်ခုကို subclass များ ခွဲထုတ်ခြင်း နှင့် အချို့အချို့သော inherited method များကို ထပ်အုပ်ပြောင်းလဲ override လုပ်ခြင်း တို့ဖြင့် စတင်ရပေသည်။ ယခုအနေအထားတွင် Android ၏ Activity *class* ကို subclass ခွဲထုတ်မည် ဖြစ်ပြီး၊ ၄င်း၏ onCreate( ) *method* ကို ထပ်အုပ်ပြောင်းလဲ override လုပ်မည် ဖြစ်သည်။

activity များတွင် Life Cycle များ ရှိကြသည်ကို ဆွေးနွေးခဲ့ပြီးလေပြီ။ (စာမျက်နှာ ၅၅–၆၀ တွင် အသေးစိတ် ပြန်လည်လေ့လာနိုင်ပါသည်) activity သည် life cycle ၏ မည်သည့် state တွင် ရှိနေမည်ကို developer များက control လုပ်ခြင်း မပြုပါ။ သို့သော် state တစ်ခုသို့ အဝင် / အကူးတွင် မည်သို့ဖြစ်မည် ဆိုသည်ကမူ developer နှင့် အဓိက သက်ဆိုင်သည်။ ယခုအခြေအနေတွင် developer အနေဖြင့် ထပ်အုပ်ပြောင်းလဲ override လုပ်ဖို့ လိုအပ်သည့် အဝင် activity မှာ (ယခု activity ကို ပထမဆုံး စတင် ဖန်တီးချိန်တွင် ActivityManager က အသက်သွင်းလိုက်သော – တနည်းဆိုရလျှင် starting state မှ running state သို့ အကူးဖြစ်သော) onCreate( ) *method* ဖြစ်ပေမည်။

system က ဖန်တီးသည့် class ကို subclass များ ပြန်ခွဲထုတ်ပြီး လိုရာနေရာတွင် ကွက်လပ်များ လိုက်ဖြည့်သည့် ဤနည်းကဲ့သို့သော programming ပုံစံမျိုးကို Template pattern ဟု ခေါ်ပါသည်။

onCreate( ) သည် ဤ application အဖို့ ('စ'လျှင် 'စ'ချင်းတွင်) တစ်ကြိမ်သာ လုပ်ရန် လိုအပ်သည့် အဓိကအလုပ်နှစ်ခုကို လုပ်ပေးရမည်။ ၄င်းတို့မှာ နိုပ်လိုက်လျှင် တုံ့ပြန်ရန်နှင့်

တိမ်တိုက်ဝန်ဆောင်မှုသို့ ချိတ်ဆက်ပေးရန် button ကို set up လုပ်ရခြင်း အလုပ်နှစ်ခု ဖြစ်သည်။

onCreate() ၏ parameter သည် Bundle ဖြစ်သည်။ ၄င်းသည် activity က စတင် ဖန်တီးလိုက်သည့် intent ပေါ်တွင် ပါသွားခြင်းအားဖြင့် activity ထဲသို့ ဝင်သွားနိုင်သည့် အနည်းငယ်သော ပမာဏရှိ data ဖြစ်သည်။ Bundle က ခွင့်ပြုသော data များသည် အခြေခံ data အမျိုးအစားများ မျှသာ ဖြစ်၏။ ပိုမိုရှုပ်ထွေးအဆင့်မြင့်သော data များကိုမူ အထူးသီးသန့် encode လုပ်ယူရပေသည်။ သို့သော် ဤအပိုင်းကို အရေးပေါ်အသုံးလိုခဲ့လျှင် အဆင်ပြေစေရန် ရည်ရွယ်၍သာ ထည့်သွင်းရှင်းလင်းထားခြင်း ဖြစ်ပါသည်။ နမူနာ Yamba *project* တွင် Bundle *parameter* ကို မလိုအပ်သဖြင့် သုံးမည် မဟုတ်ပါ။

method တစ်ခုကို override လုပ်တိုင်းတွင်၊ parent က ဖန်တီးပေးလိုက်သည့် မူလ original method ကို ခေါ်ယူရမည်။ ထိုအတွက် super.onCreate() ဟူသော ခေါ်ယူမှု ညွှန်ကြားချက် ရှိပါသည်။

Application framework ၏ class တစ်ခုကို subclass ခွဲထုတ်လိုက်ပြီး၊ သင့်တော်သည့် method ကို override လုပ်ပြီး၊ super ၏ method ကို subclass ထဲသို့ ခေါ်ယူလိုက်ပြီး သည်နှင့် လက်ရှိ code သည် (အလုပ်လုပ်မှုအားဖြင့် original class နှင့် ဘာမျှ ထူးခြားမှု မရှိသေးသော်လည်း) ကိုယ့် ကိုယ်ပိုင် code များ ပေါင်းထည့်ဖို့ရာ အကန့် တစ်ကွက် ပေါ်လာပြီ ဖြစ်လေသည်။

Activity တစ်ခု၏ onCreate() တွင် လုပ်ရလေ့ရှိသည့် ပထမဦးဆုံးသော ကိစ္စမှာ XML ဖိုင်မှနေပြီး UI ကို လှမ်းခေါ်ယူကာ Java ၏ memory space ထဲတွင် ပျံ့ကားစေရခြင်း ဖြစ်ပေသည်။ တနည်း ဆိုရလျှင် အခင်းအကျင်း XML ဖိုင်ကို ဖွင့်မည့် Java code အချို့ ရေးမည်၊ အခင်းအကျင်း XML ဖိုင်ကို အပိုင်းငယ်ကလေးများအဖြစ် အစိတ်စိတ် ပိုင်းမည်၊ XML ဖိုင်ထဲမှ element တစ်ခုချင်းစီအတွက် သက်ဆိုင်ရာ Java object ကို memory

ထဲတွင် ဖန်တီးမည်။ XML element တစ်ခုချင်းစီ၏ attribute တစ်ခုချင်းစီအတွက် ဤ code သည် Java object တွင် ၄င်း attribute ကို set လုပ်ပေးလိမ့်မည်။ ဤဖြစ်စဉ်ကို XML ဖိုင်မှ ပျံ့ကားစေခြင်း inflating from XML ဟု ခေါ်သည်။ ဤကိစ္စအားလုံးကို လုပ်ပေးသော code မှာ setContentView(R.layout.status); ဖြစ်လေသည်။

R class သည် \res (folder) ထဲရှိ XML နှင့် အခြား resource များကို Java လောကနှင့် ချိတ်ဆက်ရာတွင် ကူညီသည့် အလိုလျောက်ဖန်တီးပေးလိုက်ခြင်း ခံရသော pointer အစုကြီး ဖြစ်ကြောင်းကို (စာမျက်နှာ ၄၅ – ၄၆ တွင်) ဆွေးနွေးခဲ့ပါပြီ။ သို့အတူပင် R.layout.status သည် \res\layout\status.xml ဖိုင်ကို ညွှန်းပါသည်။

setContentView( ) *method* သည် အလုပ်များစွာကို လုပ်ပေးသည်။ အခင်းအကျင်း XML ဖိုင်ကို ဖွင့်သည်။ အခင်းအကျင်း XML ဖိုင်ကို အပိုင်းငယ်ကလေးများ အဖြစ် အစိတ်စိတ် ပိုင်းသည်။ XML element တစ်ခုချင်းစီ၏ attribute တစ်ခုချင်းစီအတွက် Java object တွင် ၄င်း attribute ကို set လုပ်ပေးသည်။ object များအကြားတွင် parent နှင့် child ဆက်သွယ်မှုများကို ပြဌာန်းပေးသည်။ ခြိုး၍ ဆိုရလျှင် မြင်ကွင်းကြီးတစ်ခုလုံးကို ပျံ့ကားဖြစ်ပေါ်စေသည်။ ဤ code စာကြောင်း အပြီးတွင် Screen သည် ဆွဲသားရန် အဆင်သင့် ဖြစ်လေပြီ။

method များကို သတ်မှတ် define လုပ်ပြီး ပြင်ပမှလှုံ့ဆော်မှုများကို တုံ့ပြန်သည်မှာ ကိုယ်ဖန်တီးသော object များသာလျှင် မဟုတ်ချေ။ Android ၏ user interface (UI) object များသည်လည်း အထက်ပါ လုပ်ဆောင်ချက်နှစ်ခုကို လုပ်သည်ပင်။ သို့ဖြစ်၍ Android UI တစ်ခုဖြစ်သော button ကိုလည်း click လိုက်သောအခါတွင် လုပ်ရန် code တစ်ခုခု ထည့်သွင်းစေခိုင်းရမည်။ ဤသို့လုပ်ရန်အတွက် onClick( ) ဟူသော method ကို define လုပ်ကာ၊ အလုပ်လုပ်စေလိုသော code ကို ထည့်သွင်းရမည်။ Button တွင် setOnClickListener *method* ကို run ရမည်။ ဤကိစ္စကို setOnClickListener သို့ argument တစ်ခုအဖြစ် ပို့ရမည်၊ အကြောင်းမှာ object သည် onClick() ကို သတ်မှတ်

ခဲ့ရာတွင် ရှိသောကြောင့် ဖြစ်သည်။ StatusActivity.java ပထမအဆင့်သည် အောက်ပါ အတိုင်း ဖြစ်ပါမည်။

Example 6-3. StatusActivity.java, version 1

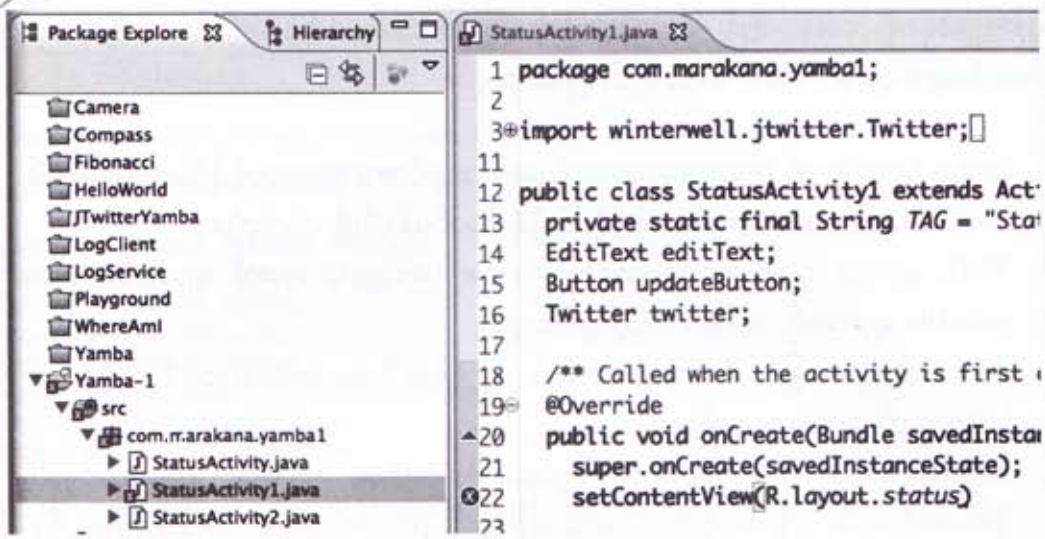
```
package com.marakana.yamba1;

import winterwell.jtwitter.Twitter;
import android.app.Activity;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;

public class StatusActivity1 extends Activity implements OnClickListener { // ❶
  private static final String TAG = "StatusActivity";
  EditText editText;
  Button updateButton;
  Twitter twitter;

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.status);

    // Find views
    editText = (EditText) findViewById(R.id.editText); // ❷
    updateButton = (Button) findViewById(R.id.buttonUpdate);

    updateButton.setOnClickListener(this); // ❸

    twitter = new Twitter("student", "password"); // ❹
    twitter.setAPIRootUrl("http://yamba.marakana.com/api");
  }

  // Called when button is clicked // ❺
  public void onClick(View v) {
    twitter.setStatus(editText.getText().toString()); // ❻
    Log.d(TAG, "onClicked");
  }
}
```

ရှေ့စာမျက်နှာရှိ code တွင် နံပါတ်များဖြင့် ပြထားသော အပိုင်းများ၏ သက်ဆိုင်ရာ ရှင်းလင်းချက်များမှာ အောက်ပါအတိုင်း ဖြစ်ပါသည်။

1. StatusActivity ကို button မှလာသည့် စေခိုင်းချက်အား နားထောင်နိုင်စေရန်အတွက် OnClickListener *interface* ကို ပြီးပြည့်စုံအောင်ဖန်တီးဖို့ လိုသည်။
2. XML layout မှ ပျံ့ကားလာစေသော view (widget) များကို ရှာဖွေပြီး Java variable များထဲသို့ assign ထည့်သွင်းသည်။
3. this (အမှန်အားဖြင့် StatusActivity) ကို (နှိပ်လိုက်သောအခါတွင်) သိရှိစေရန် button ကို register လုပ်သည်။
4. Twitter API ကို support လုပ်သော online service ဝန်ဆောင်မှုသို့ ချိတ်ဆက်သည်။ ဤအဆင့်တွင် username နှင့် password ကို code လုပ်ရမည်။
5. OnClickListener *interface* ၏ အစိတ်အပိုင်းတစ်ခုဖြစ်သည့်၊ ခလုပ်နှိပ်လိုက်လျှင် ခေါ်ယူရမည့် method ဖြစ်သည်။
6. Status ကို update လုပ်ရန် ကွန်ယက်ဝန်ဆောင်မှု ပရိုဂရမ် web service API အား တိမ်တိုက်ဝန်ဆောင်မှု cloud ကို လှမ်းခေါ်စေသည်။

၆.၉ Code ကို compile လုပ်ခြင်းနှင့် Project ကို build လုပ်ခြင်း၊ File များကို save လုပ်ခြင်း၊

Java ဖိုင်နှင့် XML ဖိုင်များကို အပြောင်းအလဲများ ပြုလုပ်ပြီးသည်နှင့် ရှေ့သို့မဆက်ခင် save လုပ်ရပါမည်။ Eclipse သည် File (menu) > Save ကို click လုပ်လိုက်တိုင်း သို့ Ctrl + S နှိပ်လိုက်တိုင်း၌ Project ကို အလိုလျောက် build လုပ်ပါသည်။သို့ဖြစ်သော ကြောင့် ဖိုင်များကို save လုပ်ရန်နှင့်၊ ဖိုင်တစ်ဖိုင်နှင့်ပတ်သက်၍ ကိစ္စမပြတ်မီ အခြား file သို့ ကူးကာ အလုပ်လုပ်ခြင်း မပြုရန် အထူးအရေးကြီးပါသည်။ ဖိုင်တစ်ဖိုင်နှင့် ပတ်သက်၍ code ထဲ၌ အနီရောင် *x* သင်္ကေတလေးများ ရှင်းသွားလျှင် ကိစ္စပြတ်ပြီ ဖြစ်ကြောင်း သိနိုင်ပါသည်။ Error များ Trace လိုက်ခြင်းအတွက် တစ်ဖက်စာမျက်နှာပါ ပုံကို ကြည့်ပါ။



၆.၁၀ jtwitter.jar *library* ဖိုင်ကို ပေါင်းထည့်ခြင်း

twitter နှင့် ချိတ်ဆက်ပေးနိုင်မည့် ပရိုဂရမ် twitter compatible API ကို အလုပ်လုပ်ပေးနိုင်မည့် online service အား ချိတ်ဆက်မည်။ ဤဆက်သွယ်မှု connection သည် ဆက်တိုက်ဖြစ်သော ကွန်ယက်ဝန်ဆောင်မှုခေါ်ယူခြင်း web service call များအားဖြင့် ဖြစ်ရပေသည်။ Android သည် Java ၏ စံထားကွန်ယက်ဆက်သွယ်ရေး စွမ်းဆောင်မှုများ standard Java networking capabilities ကို အသုံးပြုသည်ဖြစ်ရာ၊ Java တွင် ရှိနှင့်ပြီးသော web service များကိုပင် တိုက်ရိုက်အသုံးပြုရုံပင် ဖြစ်သည်။ (Android တွင်လည်း အခြား ကွန်ယက်ဆက်သွယ်ရေးစွမ်းဆောင်မှု networking capability မူကွဲ version မရှိချေ)

ပင့်ကူအိမ်ကွန်ယက်ဝန်ဆောင်မှု web service များနှင့် Twitter API တို့နှင့်ပတ်သက်ပြီး ပိုမိုလွယ်ကူသွားစေရန်အတွက် Winterwell Associates မှ ဖန်တီးထားသော third-party library တစ်ခုဖြစ်သည့် jtwitter.jar ကို သုံးပါမည်။ ဤ library ဖိုင်သည် တကယ်တမ်းမှ ရိုးစင်းလှသော Java calss တစ်ခုသာ ဖြစ်၏။ ယင်းသည် online service

နှင့် အပြန်အလှန် တုံ့ပြန်ဆက်သွယ် interact လုပ်ပေးပြီး၊ network call များ ခေါ်ယူခြင်း နှင့် data ကို အပြန်အလှန် ရယူ / ပေးပို့ခြင်းတို့၏ ရှုပ်ထွေးခက်ခဲမှုများကို လျော့ပါးစေ သည်။ ဤကဲ့သို့ high-level library တစ်ခုကို မည်သူမဆိုရယူနိုင်သောရင်းမြစ် open source အဖြစ် မည်သူကမျှ ဖြန့်ဝေထားခြင်း မရှိခဲ့ပါမူကား ဤအလုပ်ကို ပြီးမြောက်စေရန် standard Java networking libraries ဖိုင်တို့ကိုပင် သုံးကာ တစ်စချင်း ဖန်တီးယူရပေ မည်။

အထက်တွင် ဆိုခဲ့သော jtwitter library ကို CD ထဲတွင် jtwitter.jar အဖြစ် တွေ့နိုင်ပါ သည်။ download ရယူနိုင်သော အင်တာနက်စာမျက်နှာမှာ http://www.winterwell.com/software/jtwitter.php ဖြစ်ပါသည်။

WINTERWELL ASSOCIATES

About us | Contact us | Hiring now!

JTwitter - the Java library for the Twitter API

Robust & easy to use. Over 80,000 downloads and many successful apps. As used in several university courses.

JTwitter is a small library providing easy access to the Twitter API. Set and get your status, manage your network of friends, message friends, maintain lists, etc. Lots of features - if you can do it at twitter.com, you can do it with JTwitter. Includes work-arounds for rough-edges and issues in the Twitter API to make coding easiser and more robust.

The latest version of the library can be downloaded:

- jtwitter.jar + source & test code + dependencies: [jtwitter-2.3.4.zip]
- Just the jar: [jtwitter.jar v2.3.4]. In which case you will also need an OAuth library - we recommend Signpost: [signpost jar]

Alternatively, if you are familiar with git, use the JTwitter repository on GitHub.

The recent versions fix a couple of issues with changes and bugs in the Twitter API. If you are using an older version, please download the new version.

ဝင်းဝင်းဟန်
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

The source code is included inside the jar file. It is also available online here: [Twitter.java]

JTwitter is very easy to use. Example usage:

```
// Make an oauth client (you'll want to change this bit)
OAuthSignpostClient oauthClient = new OAuthSignpostClient(JTWITTER_OAUTH_KEY,
JTWITTER_OAUTH_SECRET, "oob");
// Open the authorisation page in the user's browser
// On Android, you'd direct the user to URI url = client.authorizeUrl();
// On a desktop, we can do that like this:
oauthClient.authorizeDesktop();
// get the pin
String v = oauthClient.askUser("Please enter the verification PIN from Twitter");
oauthClient.setAuthorizationCode(v);
// Store the authorisation token details for future use
String[] accessToken = client.getAccessToken();
// Next time we can use new OAuthSignpostClient(OAUTH_KEY, OAUTH_SECRET,
//      accessToken[0], accessToken[1]) to avoid authenticating again.

// Make a Twitter object
Twitter twitter = new Twitter("my-name", oauthClient);
// Print Daniel Winterstein's status
System.out.println(twitter.getStatus("winterstein"));
// Set my status
twitter.setStatus("Messing about in Java");
```

You'll find the code is fully documented, and mostly self-explanatory. The javadoc is also available online here: [javadoc], with the change-log (including credits to contributors) here: change log.

OAuth support included

JTwitter has only one dependency: an OAuth library. We recommend Signpost, which is included in the .zip download.

OAuth support is also provided in the flavours:

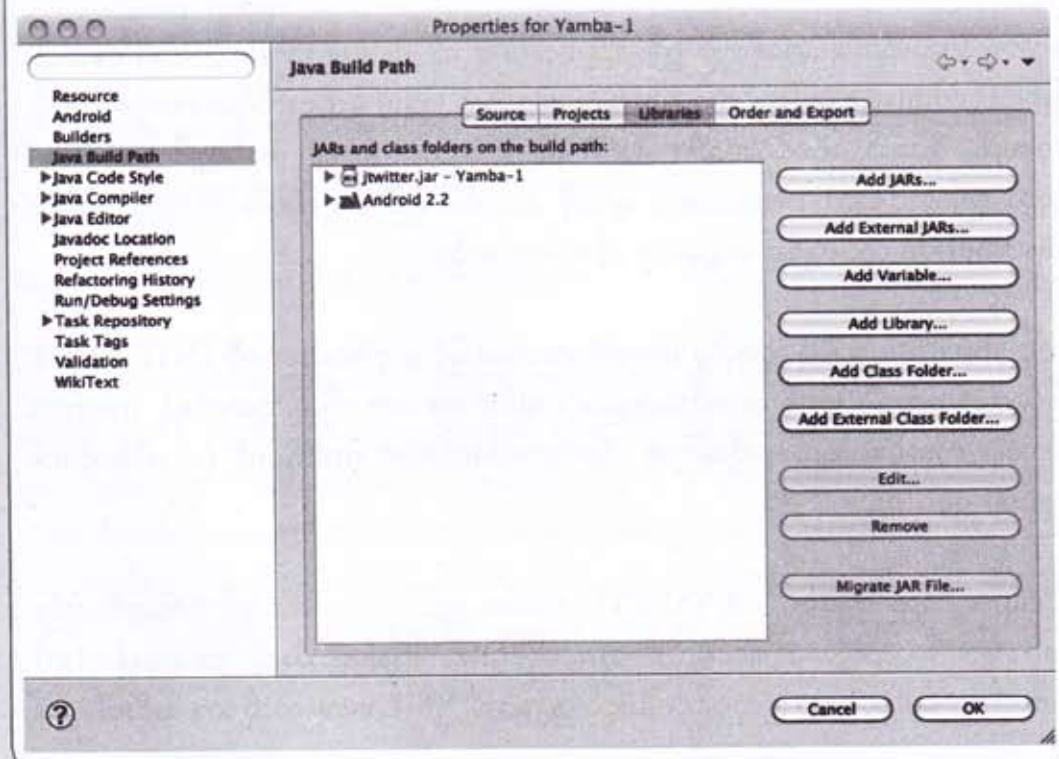
1. OAuthScribeClient which requires the Scribe OAuth library and the Apache commons codec. (documentation).
2. OAuthHttpClient.java, which requires the Java OAuth library and its considerable dependencies. The OAuthHttpClient.java file should be extracted from the jar and copied into your project if you want to use it.

C# / .NET version

Thanks to Patrick Schidler, JTwitter has been ported to C# and is available for the .Net framework. *NTwitter* is available here: http://www.blogmenot.de.

ဤ jtwitter.jar ဖိုင်ကို Eclipse ထဲရှိ ယခု project ထဲသို့ ကူးထည့်ရပါမည်။ Package Manager (window) ထဲတွင်ရှိသည့် Eclipse project ၏ root ထဲသို့ drag-and-drop ကူးထည့်လိုက်ရုံသာ ဖြစ်ပါသည်။ ဤလုပ်ဆောင်ချက်သည် ဖိုင်ကို project ထဲတွင် ပါဝင်သွားစေပါပြီ။ သို့သော် project ၏ Java code သည် ၄င်းကို မသိသေးပါ။

Java သည် class အားလုံးကို ၄င်း၏ classpath တွင် ရှာဖွေသည်။ jtwitter.jar ကို classpath ထဲတွင် ပေါင်းထည့်ရန်အတွက် project ပေါ်တွင် right-click လုပ်ပြီး properties ကို ရွေးပါ။ Properties for Yamba -1 (dialog window) ပေါ်လာပါမည်။ Java Build Path ကို ရွေးပြီး Libraries (tab) ကို click ပါ။ Add JARs ကို click ပြီး ပေါ်လာသော dialog window တွင် jtwitter.jar ဖိုင်ကို လမ်းကြောင်း သတ်မှတ်ပေးလိုက်ပါ။ အောက်ပါအတိုင်း တွေ့ရပါမည်။



ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

၆.၁၁ Internet Permission အတွက် Manifest File ကို update လုပ်ခြင်း

ဤ application ကို အလုပ်မလုပ်စေမီ developer အနေဖြင့် application က internet အား ချိတ်ခွင့်ကို user က ခွင့်ပြုရန် user ကို ခွင့်တောင်းရပေမည်။ အချို့သော အန္တရာယ်ရှိသည့် operation များ ရှိနိုင်သည့်အတွက် ခွင့်ပြုချက်တောင်းခံမှုများကို ပြဋ္ဌာန်းထားခြင်းဖြင့် Android သည် လုံခြုံရေးကို ထိန်းချုပ်သည်။ ထိုအချက်ကြောင့် user သည် application တစ်ခုကို စပြီး install လုပ်ချိန်တွင် ဤ application က တောင်းခံသော ခွင့်ပြုချက် များကို user သည် ကွဲကွဲပြားပြား ဆုံးဖြတ်ရ၏။ application က တောင်းခံသော ခွင့်ပြုချက်များကို လုံးဝ ခွင့်ပြုသည် သို့တည်းမဟုတ် လုံးဝ ခွင့်မပြုပါ ဆိုသည်ကို ဖော်ပြရသည်။ ကြား အနေအထား မရှိချေ။ ထို့ပြင် (user အနေဖြင့် လုံးဝ ခွင့်ပြုခဲ့လျှင်) လက်ရှိ ရှိနေသော app ကို upgrade လုပ်လည်း prompt မပြတော့ချေ။

လောလောဆယ်အခြေအနေတွင် ဤ application ကို debug mode ၌ run နေသည် ဖြစ်ပြီး (emulator ပေါ်တွင်သာ run ခြင်း မဟုတ်ဘဲ ကြိုးမဲ့ မိုဘိုင်းကိရိယာ တစ်ခုခုထဲသို့ တကယ် ပို့သည် ဆိုလျှင်လည်း) USB cable မှသာ install လုပ်လျှင် Android သည် ဤကဲ့သို့သော permission များကို တောင်းခံမည် မဟုတ်ပါ။ သို့သော် online distribution လုပ်လျှင်မူ မလွဲမသွေ လိုအပ်လေမည်။

ယခု application အရ ဆိုရပါမူ developer အနေဖြင့် application ကို INTERNET ဖြင့် ချိတ်ဆက်ခွင့် ပြုရန် တောင်းခံရမည်။ online service ကို ချိတ်ဆက်ရန် internet access လိုအပ်ပေသည်။ ထို့ကြောင့် AndroidManifest.xml ဖိုင်ကို (double-click လုပ်၍) ဖွင့်ပါမည်။

Eclipse သည် ဤဖိုင်ကို WYSIWYG editor တွင် ဖွင့်ပြီး၊ အောက်ဘက်တွင်လည်း tab များကို များစွာတွေ့ရပေမည်။ ထုံးစံအတိုင်းပင် အပြောင်းအလဲ အားလုံးနီးပါးကို interface ပေါ်တွင်သာ ပြုလုပ်နိုင်ပါသည်၊ သို့သော် XML view ထဲသို့သာ ဆင်းပါမည်။

အောက်ခြေရှိ tab များအနက် လက်ယာဘက် အစွန်ဆုံး tab ကို AndroidManifest.xml အဖြစ် တွေ့ရပါမည်။ (1) ထို tab ကို ဖွင့်ကာ <manifest> အဖွင့်အပိတ် အတွင်းတွင် <uses-permission android:name="android.permission.INTERNET"/> ဟု တစ်ကြောင်း ဖြည့်ပါမည်။ AndroidManifest.xml သည် အောက်ပါအတိုင်း ဖြစ်သွားပါမည်။

Example 6-4. AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  android:versionCode="1" android:versionName="1.0" package="com.marakana.yamba1">
  <application android:icon="@drawable/icon" android:label="@string/app_name">

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

  </application>
  <uses-sdk android:minSdkVersion="4" />

  <uses-permission android:name="android.permission.INTERNET" /> <!-- ❶ -->
</manifest>
```

၆.၁၂ Andriod ထဲတွင် Log မှတ်တမ်းပြုခြင်း

Android သည် system တစ်ခုလုံး အတိုင်းအတာအထိ log လုပ် မှတ်တမ်းတင်နိုင်စွမ်းကို ထားရှိသည်။ code ၏ မည်သည့်နေရာမှ မဆို Log.d(TAG, message) ဟု ခေါ်ယူလိုက်ရုံ မျှသာ ဖြစ်သည်။ TAG နေရာသည် ကိုယ့် code အတွက် အဓိပ္ပာယ်တစုံတရာ ကိုယ့်ဘာသာ ကိုယ် ထားရှိသော (data type အားဖြင့် string ဖြစ်သည့်) tag တစ်ခုဖြစ်ပြီး၊ message သည်လည်း (data type အားဖြင့် string ဖြစ်သည့်) စာသားတစ်ခုခု ဖြစ်မည်။ အများ အားဖြင့် tag အဖြစ် application ၏ အမည်၊ class ၏ အမည်၊ သို့မဟုတ် module တစ်ခုခု၏ အမည်ကို ထားလေ့ရှိသည်။ လုပ်သင့်သော အလေ့အထမှာ TAG ကို class

တစ်ခုလုံးအတွက် constant သတ်မှတ်ထားခြင်း ဖြစ်ပါသည်။ ဥပမာ –
private static final String TAG = "StatusActivity";

code ကို compile မလုပ်မီတွင် Log *class* ကို import လုပ်ဖို့ လိုပါသည်။ Eclipse တွင် Source > Organize Imports အောက်၌ အသုံးဝင်သော feature တစ်ခု ရှိသည်။ Ctrl + O နှိပ်လျှင်လည်း ရပါသည်။ ဤ feature သည် ကိုယ့် import statement များ ကို အလိုလျောက် စုစည်း organize လုပ်သွားပါမည်။ သို့သော် Log ဟူသော class တွင်မူ Log *class* များ တစ်ခုမက ဖြစ်နိုင်သောကြောင့် ပဋ္ဌိပက္ခ conflict များ ရှိနိုင်ပါသည်။ ကိုယ်တိုင် သင့်ရာကို ဆုံးဖြတ် ရွေးချယ်ရပါမည်။ ယခုအနေအထားတွင်မူ Android ၏ Log နှင့် Apache ၏ Log ဟူ၍သာ ရှိမည်ဖြစ်သောကြောင့် ရွေးရန်လွယ်ပါသည်။

Log သည် အရေးပါမှုအဆင့် အသီးသီး ရှိပါသည်။ .d( ) သည် debug အဆင့်အတွက် ဖြစ်ပါသည်။ error အတွက် .e( ) ကို လည်းကောင်း၊ warning အတွက် .w( ) ကို လည်းကောင်း၊ info အတွက် .i( ) ကို လည်းကောင်း ရွေးနိုင်ပါသည်။ လုံးဝ ဖြစ်ရန် အကြောင်းမရှိသော error မျိုးများ အတွက် .wtf( ) အဆင့်လည်း ရှိပါသေးသည်။ (wtf ဆိုသည်မှာ what a terrible failure ဟု ပြောကြသည်၊ အချို့ကလည်း what the f**k ဟု ဆိုကြသည်၊ Android ကို ထွင်သော ဆရာများသည် အတော် ဟာသဉာဏ်ရှင်သူများ ဟု ဆိုရပေမည်လော မပြောတတ်ချေ။)

Eclipse သည် message များကို ၄င်းတို့၏ အရေးပါမှုအဆင့်များအလိုက် အရောင်များခွဲ၍ ပြသသည်။

Eclipse ၏ Organize Imports *tool* သည် တခါတရံတွင် ရှာရခက်သည့် ပြဿနာများကို ဖြစ်စေတတ်သည်။ ဥပမာ – ကိုယ့် project တွင် (XML resource တစ်ခုခု၌ ပြဿနာရှိနှင့် ခဲ့သောကြောင့်) R.java ကို auto-generate မလုပ်ပေးနိုင်ခဲ့လျှင် Organize Import သည် Android.R *class* ကို import လုပ်လိုက်ပေလိမ့်မည်။ သို့ဖြစ်၍ R resource များ

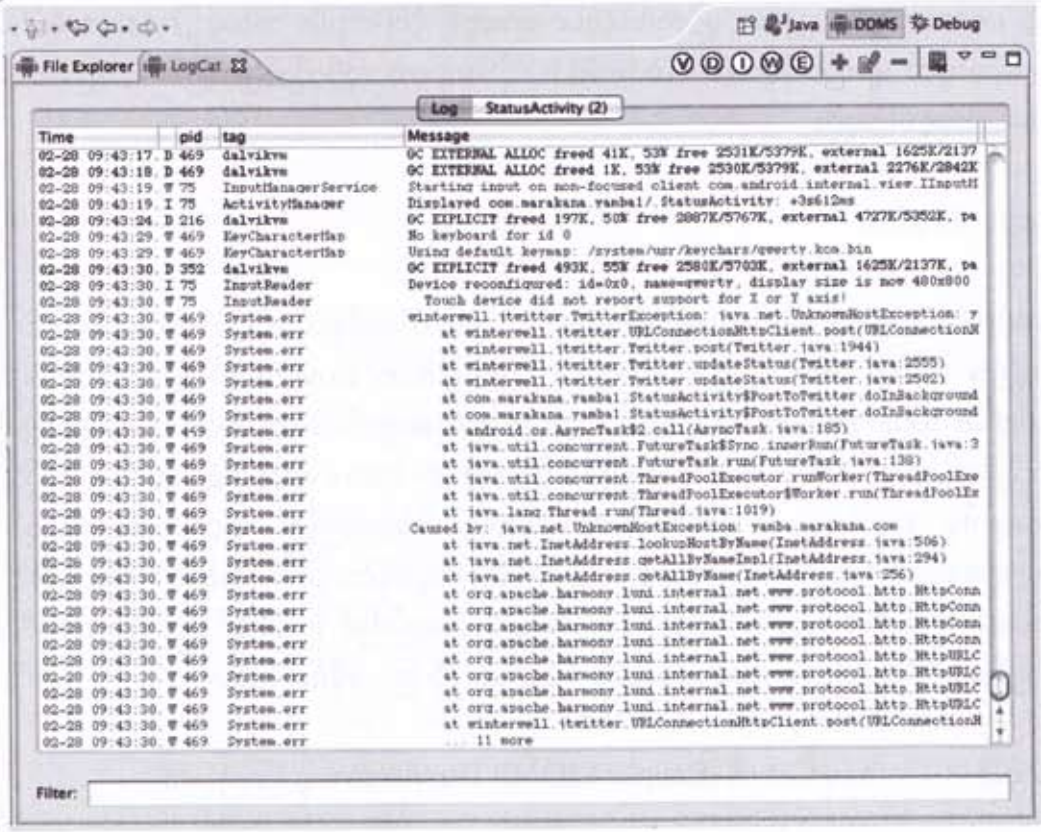
နှင့် ပတ်သက်သော ညွှန်းဆိုမှု reference များတွင် (compile အဆင့်၌) error များ အရေအတွက် များပြားနေပါက Android.R ကို import လုပ်လိုက်ခြင်းဖြစ်၏ /မဖြစ်၏ စစ်ဆေးသင့်ပါသည်။

၆.၁၃ LogCat

Android system ၏ log များအားလုံးကို system တစ်ခုလုံးဆိုင်ရာ မှတ်တမ်းတင် ယန္တရား system-wide logging mechanism ဖြစ်သော LogCat ဆီသို့ ထုတ်သည်။ LogCat သည် Java နှင့် C/C++ code အားလုံးအတွက် အလုပ်လုပ်သည်။ debug သည် log များကို လွယ်လင့်တကူ ကြည့်နိုင်ပြီး debug ၊ info ၊ warning ၊ error စသည့် အရေးပါမှု အဆင့်များ အလိုက်သော်လည်းကောင်း၊ မိမိစိတ်ကြိုက် သတ်မှတ်ထားသည့် tag များ အလိုက်သော်လည်းကောင်း စစ်ထုတ် filter လုပ်နိုင်သည်။ Android development ၏ ထုံးစံအတိုင်း LogCat ကိုလည်း နည်းနှစ်နည်းဖြင့် ကြည့်နိုင်သည်။ Eclipse မှ ဖြစ်စေ၊ Command Line မှ ဖြစ်စေ LogCat ကို ကြည့်နိုင်ပါသည်။

၆.၁၃.၁ LogCat ကို Eclipse DDMS Perspective မှ ကြည့်ခြင်း

Eclipse ထဲမှ LogCat ကို ကြည့်ရန် LogCat View *tag* ကို ဖွင့်ရပါမည်။ ရှေ့စာမျက်နှာမှ ပုံအတိုင်း တွေ့ရပါမည်။



Eclipse *menu* > Windows > OpenPerspective > DDMS မှလည်း ကြည့်နိုင်ပါသည်။

DDMS ဆိုသည်မှာ Dalvik Debug Monitor Server ဖြစ်ပါသည်။ DDMS နှင့် ပတ်သက်သော website ဖြစ်သည့် http://developer.android.com/guide/developing/debugging/ddms.html ကို တစ်ဖက်စာမျက်နှာအတိုင်း တွေ့နိုင်ပါသည်။

Debugging >

Using DDMS

Android ships with a debugging tool called the Dalvik Debug Monitor Server (DDMS), which provides port-forwarding services, screen capture on the device, thread and heap information on the device, logcat, process, and radio state information, incoming call and SMS spoofing, location data spoofing, and more. This page provides a modest discussion of DDMS features; it is not an exhaustive exploration of all the features and capabilities.

In this document

Running DDMS

DDMS is integrated into Eclipse and is also shipped in the `tools/` directory of the SDK. DDMS works with both the emulator and a connected device. If both are connected and running simultaneously, DDMS defaults to the emulator.

- From Eclipse: Click **Window > Open Perspective > Other... > DDMS**.
- From the command line: Type `ddms` (or `./ddms` on Mac/Linux) from the `tools/` directory.

How DDMS Interacts with a Debugger

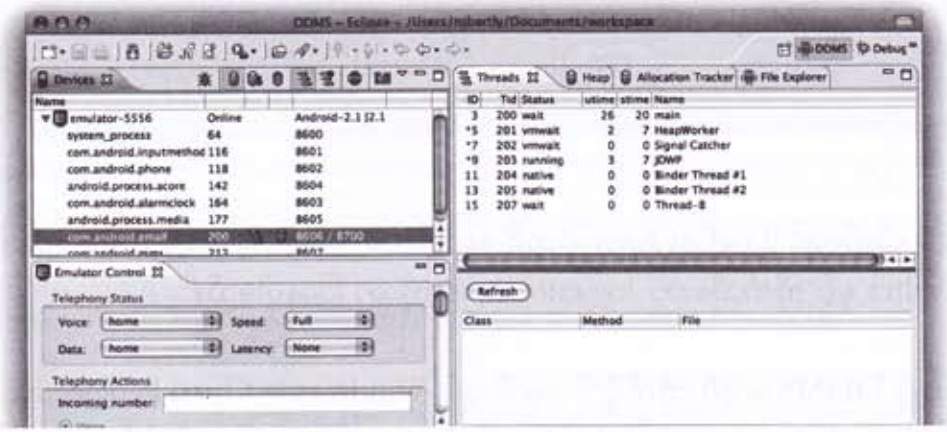
On Android, every application runs in its own process, each of which runs in its own virtual machine (VM). Each VM exposes a unique port that a debugger can attach to.

When DDMS starts, it connects to adb. When a device is connected, a VM monitoring service is created between adb and DDMS, which notifies DDMS when a VM on the device is started or terminated. Once a VM is running, DDMS retrieves the the VM's process ID (pid), via adb, and opens a connection to the VM's debugger, through the adb daemon (adbd) on the device. DDMS can now talk to the VM using a custom wire protocol.

DDMS assigns a debugging port to each VM on the device. Typically, DDMS assigns port 8600 for the first debuggable VM, the next on 8601, and so on. When a debugger connects to one of these ports, all traffic is forwarded to the debugger from the associated VM. You can only attach a single debugger to a single port, but DDMS can handle multiple, attached debuggers.

By default, DDMS also listens on another debugging port, the DDMS "base port" (8700, by default). The base port is a port forwarder, which can accept VM traffic from any debugging port and forward it to the debugger on port 8700. This allows you to attach one debugger to port 8700, and debug all the VMs on a device. The traffic that is forwarded is determined by the currently selected process in the DDMS Devices view.

The following screenshot shows a typical DDMS screen in Eclipse. If you are starting DDMS from the command line, the screen is slightly different, but much of the functionality is identical. Notice that the highlighted process, `com.android.email`, that is running in the emulator has the debugging port 8700 assigned to it as well as 8606. This signifies that DDMS is currently forwarding port 8606 to the static debugging port of 8700.



DDMS သည် ကြိုးမဲ့မိုဘိုင်းကိရိယာပေါ်တွင် run မည့် application နှင့် developer ၏ developement environment (ဥပမာ – Eclipse) ကို ဆက်သွယ်ပေါင်းကူးပေးသော အရာ ဖြစ်သည်။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

LogCat တွင် စစ်ထုတ်မှု filter များကိုလည်း ခံထားနိုင်ပါသည်။ အစိမ်းရောင် အပေါင်းပုံ ခလုပ်ကို နှိပ်လျှင် အောက်ပါအတိုင်း LogCat fiter *dialog* ပေါ်လာပါမည်။

filter ခံ လိုက်လျှင် LogCat *panel* ထဲတွင် နောက်ထပ် window တစ်ခု ထပ်ပေါ်လာပြီး ကိုယ့် filter နှင့် ကိုက်ညီသော log entries များကိုသာ ပြနေပါမည်။

အကယ်၍ DDMS ခလုပ် တိုက်ရိုက် မပေါ်လျှင် Windows > Open Perspective မှ DDMS ကို click လုပ်ပါ။ ဤနည်းဖြင့် ပထမဦးဆုံး အကြိမ် ခေါ်ယူလိုက်ပြီးလျှင် နောင် အခါများတွင် window tab ပေါ်၌ ပေါ်လာပါလိမ့်မည်။

၆.၁၃.၂ LogCat ကို command line မှ ကြည့်ခြင်း

LogCat ကို command line မှ ကြည့်ရန်အတွက်ဆိုလျှင် command prompt ကို ဖွင့်ပြီး [user:~]> adb logcat ဟု ရိုက်ထည့်လိုက်ရပါမည်။

[user:~] ဆိုသည်မှာ C:\apps\yamba1\ သို့မဟုတ် C:\android-sdk\yamba1\ သို့မဟုတ် C:\user\user\workspace\ စသည်ဖြင့် ကိုယ့် project နှင့် tools များ ရှိရာ တည်နေရာ လမ်းကြောင်း ဖြစ်ပါမည်။

ထိုသို့ ရိုက်ထည့်လိုက်လျှင် လက်ရှိ LogCat ကို တွေ့ရမည် ဖြစ်ပြီး ကိုယ့် device မှ log entries များကို generate လုပ်တိုင်းတွင် update ဖြစ်ပါမည်။ log entries များကို command line မှလည်း filter လုပ်နိုင်ပါသည်။ သို့သော် ဤနည်းသည် အလွယ်တကူ နားလည်လွယ် သုံးစွဲလွယ်သော နည်းကား မဟုတ်ချေ။ StatusActivity သက်သက် နှင့် ပတ်သက်သော entries များကို ကြည့်ရန် ဆိုလျှင် StatusActivity:* ဟု ပြဋ္ဌာန်း specify လုပ်ရပါမည်။ ဤသို့ ပြဋ္ဌာန်း specify လုပ်လိုက်ခြင်းသည် ဤ tag နှင့် ပတ်သက်ပြီး အရေးပါမှုအဆင့် severity level အားလုံးကို အလိုရှိသည်ဟု ဖော်ပြလိုက်ခြင်းလည်း ဖြစ်ပေသည်။ မကြည့်လိုသော အရာများရှိလျှင်လည်း သတ်မှတ်ပြဋ္ဌာန်း specify လုပ်ရန် လိုအပ်သည်။ *:S ဟု specify လုပ်လိုက်ခြင်းသည် အခြား tag အားလုံးကို ပျောက်သွား စေ၏။ command line သည် အောက်ပါအတိုင်း ဖြစ်သွားမည်။

```
[user:~]> adb logcat StatusActivity:* *:S
```

အသုံးတည့်သော နည်းလမ်းတစ်ခုမှာ command prompt *window* ကို တောက်လျှောက် ဖွင့်ထားပြီး adb logcat ကို တလျှောက်လုံး run ထားစေခြင်း ဖြစ်ပါသည်။ ဤနည်းဖြင့် ကိုယ့် application တွင် မည်သို့ဖြစ်နေသည်ကို ချက်ချင်း သိနိုင်ပြီး၊ ဤနည်းသည် Eclipse ထဲမှ DDMS Perspective ကို သွားပြန်ဖွင့်နေရခြင်းထက် မြန်ဆန်ပါသည်။

၆.၁၄ Andriod ၏ Thread သဘော (Threading)

Thread တစ်ခု ဆိုသည်မှာ အစဉ်အလိုက် အကောင်အထည်ဖော် execute လုပ်သွားသည့် ညွှန်ကြားချက် instruction အစဉ်ကြီးတစ်ခု ဖြစ်ပါသည်။ CPU အားလုံးနီးပါးသည် တစ်ကြိမ်တွင် ညွှန်ကြားချက် instruction တစ်ခုကိုသာ process လုပ်နိုင်စွမ်း ရှိသည် မှန်သော်လည်း၊ မောင်းနှင်စနစ် operating system အားလုံးနီးပါးသည်ကား တစ်ခုထက် ပိုသော multiple CPU များအတွက် တစ်ခုထက်ပိုသည့် multiple thread များကို ကိုင်တွယ်နိုင်စွမ်းသည် လည်းကောင်း၊ CPU တစ်ခုတည်းတွင် အလွှာအသီးသီး ခွဲထုတ်နိုင်စွမ်းသည် လည်းကောင်း ရှိပေသည်။ thread များ မတူသည့်အားလျော်စွာ ဦးစားပေး အဆင့်လည်း မတူပေ။ သို့ဖြစ်၍ ထိုသို့ ဦးစားပေးအဆင့် မတူညီသော thread များသည် CPU တစ်ခုတည်းကို မျှဝေသုံးစွဲကြသောအခါတွင် thread တစ်ခုစီအား အချိန်မည်မျှစီ ပေးမည် ဆိုသည်ကို မောင်းနှင်စနစ် operating system က ဆုံးဖြတ်လေသည်။

Android OS သည် Linux ပေါ်တွင် အခြေခံထားသည်၊ သို့ဖြစ်ရကား တစ်ချိန်တည်းတွင် တစ်ခုထက်ပိုသော multiple thread များအား run နိုင်မှုကို အပြည့်အဝ စွမ်းဆောင်နိုင်၏။ သို့သော် developer သည် မိမိ၏ application ကို ကောင်းမွန်အောင် design ချရာတွင် thread များအား application များ မည်သို့အသုံးပြုသည်ကိုမူ သိရပေမည်။

၆.၁၄.၁ Single Thread

ပုံမှန်အားဖြင့် Android application တစ်ခုသည် thread တစ်ခုတည်း single thread ပေါ်တွင် run သည်။ thread တစ်ခုတည်းပေါ်တွင် run သော application များသည် ညွှန်ကြားချက် command အားလုံးကို တစ်ခုပြီးတစ်ခု အစဉ်အလိုက် serial ပုံစံဖြင့် run သည်။ ရှေ့ command အဆုံးမသတ်မချင်း နောက်တစ်ခု ပြီးရန်မရှိချေ။ Andriod အသုံးအနှုန်းဖြင့် ဆိုရပါမူ call တစ်ခုစီတိုင်းသည် block ဖြစ်နေသည်။

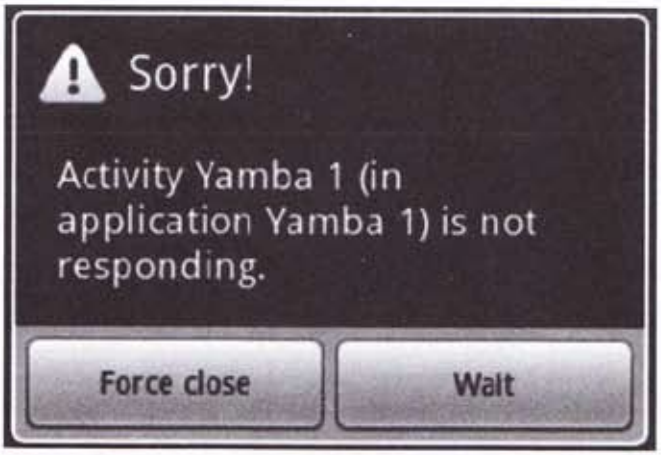
ဤ single thread ကို UI thread ဟုလည်း ခေါ်သည်။ အကြောင်းမှာ ဤ thread သည်ပင်လျှင် user interface ပေါ်မှ command အားလုံးကို process လုပ်ပေးခြင်း ကြောင့် ဖြစ်၏။ UI Thread သည် screen ပေါ်ရှိ element အားလုံးကို ဖန်တီးတည် ဆောက်ပေးရုံ မက၊ screen သို့ user က လာရောက် ထိခြင်း၊ button များကို နှိပ်ခြင်း စသည့် user ၏ event အားလုံးကိုလည်း တာဝန်ယူပေးရသည်။ တစ်ခုတည်း ရှိသော single UI thread ပေါ်တွင် code အား execute လုပ်ပုံကို အောက်ပါပုံတွင် လေ့လာနိုင် ပါသည်။

StatusActivity ကို single thread ပေါ်တွင် run ခြင်း၏ ပြဿနာမှာ status ကို update လုပ်ရန်အတွက် ကွန်ယက်ခေါ်ဆိုမှု network call ဖြစ်သည်။ ကွန်ယက်ခေါ်ဆိုမှု network call အားလုံးတွင် အကောင်အထည်ဖော် execute လုပ်ရန် ကြာချိန်သည် developer ၏ ထိန်းချုပ်နိုင်စွမ်းအောက်တွင် မရှိသော ကိစ္စ ဖြစ်၏။

twitter.updateStatus() *function* ကို call လုပ်ခြင်းသည် ကွန်ယက်ရရှိနိုင်မှု အားလုံးနှင့် ပတ်သက်လေသည်။ user တွင် cloud ကို ချိတ်ဆက်ရန်အတွက် အလင်းအလျင်နီးနီး မြန်သော Wi Fi ဆက်သွယ်မှု ရှိသည်လည်း ဖြစ်နိုင်သလို၊ အခြေအနေဆိုးဝါးလှသော network connectivity မျိုးကို အသုံးပြုရသည်လည်း ဖြစ်နိုင်ပေသည်။

ရှင်းစွာ ဆိုရလျှင် single thread ကို သုံးလိုက်ပါက application သည် network call *function* မပြီးမချင်း ဘာမျှ ဆက်လုပ်တော့မည် မဟုတ်ချေ။

Android စနစ်သည် ကြာချိန်ကာလတစ်ခု (အများအားဖြင့် activity တစ်ခုအတွက် ငါးစက္ကန့်) အတွင်း တုံ့ပြန်မှုမလာတော့သော application ကို ပိတ်သိမ်း kill လုပ်ခွင့် ပြုသော လုပ်ဆောင်ချက်ကို ပေးထားသည်။ ၄င်းကို application not responding dialog သို့မဟုတ် အတိုကောက်အားဖြင့် ANR ဟု ခေါ်သည်။ ANR ပုံကို ရှေ့စာမျက်နှာတွင် ကြည့်ပါ။



၆.၁၄.၂ Multi Thread

ပိုကောင်းသော နည်းလမ်းမှာ အချိန်ကြာရန် အလားအလာရှိသော လုပ်ဆောင်ချက် operation များကို သီးခြား thread တစ်ခုတွင် run ခြင်း ဖြစ်သည်။ တစ်ခုထက်ပိုသော အလုပ် multiple tasks များကို တစ်ခုထက်ပိုသော multiple thread များတွင် ခွဲဝေပြီး တပြိုင်နက် run သောအခါတွင် CPU ကို (လုပ်ဆောင်ချက် task တစ်ခုတည်းက သိမ်းပိုက်ထားခြင်း မရှိစေရန်အတွက်) မောင်းနှင်စနစ် operating system က အစိတ်များစွာ ပိုင်းလိုက်သည်။ သို့ဖြစ်၍ တစ်ခုထက်ပိုသော task များသည် တစ်ချိန်တည်း တပြိုင်တည်းတွင် အတူတကွ အလုပ်လုပ်နေသကဲ့သို့ ဖြစ်သွားစေသည်။

ယခု နမူနာတွင် status ကို cloud မှ လှမ်းဆွဲယူကာ update လုပ်ရန် ကွန်ယက်ခေါ်ဆိုမှု network call ကို သီးခြား thread တစ်ခုပေါ်သို့ ပို့နိုင်သည်။ ဤနည်းအားဖြင့် network ကို စောင့်ဆိုင်းနေစဉ်တွင် UI thread သည် block မဖြစ်နေတော့ပဲ၊ application သည်လည်း တုံ့ပြန်မှုအနေအထား ပိုမိုကောင်းမွန်လာပေမည်။ ပင်မ main thread ကြီးသည် ရှေ့ပိုင်းတွင် run နေသည် ဖြစ်ပြီး၊ ထပ်ဖြည့် thread များသည် နောက်ကွယ်တွင် run နေသည် ဖြစ်ရပေမည်။ ၄င်းတို့အားလုံးသည် အမှန်မှာ အတူတူပင် ဖြစ်သည်၊ မိုဘိုင်းကြိုးမဲ့ကိရိယာ၏ CPU ပေါ်တွင် တလှည့်စီ run နေကြခြင်းသာ ဖြစ်သည်။ သို့သော် user ၏

အထင်အမြင်တွင်မူ main thread သည် user interface (UI) နှင့် ပတ်သက်သဖြင့် ရှေ့ပိုင်းတွင် run နေပေသည်။ code ၏ thread နှစ်ခုဖြစ်သော ပင်မ UI thread နှင့် run ချိန် ကြာမြင့်မည့် အလားအလာရှိ အရံ auxiliary thread တို့အား အကောင်အထည်ဖော် execute လုပ်ပုံကို အောက်ပါပုံတွင် လေ့လာနိုင်ပါသည်။

multithreading ကို ဖြစ်စေရန် နည်းလမ်းများစွာ ရှိပါသည်။ ဤလုပ်ဆောင်ချက် operation များကို ဆောင်ရွက်သော Thread ဟူသည့် class တစ်ခုသည် Java တွင် ရှိပါ၏။ network call ကို နောက်ကွယ်တွင် အလုပ်လုပ်နေစေရန်အတွက် Java ၏ သမားရိုးကျ feature များကို သုံးနိုင်သည်သာပင်။

သို့သော် သမားရိုးကျဖြစ်သည့် Thread ဟူသော Java ၏ class ကို သုံးခြင်းကား ပြဿနာ ရှိပေသည်။ အကြောင်းမှာ ပင်မ UI thread ထဲရှိ element များကို အခြား thread မှ လှမ်းကာ update လုပ်ခြင်းအား ခွင့်မပြုသောကြောင့်တည်း။ ၄င်းသည် အဓိပ္ပာယ်ရှိသည်။ အကြောင်းမှာ UI thread ကို update လုပ်ရန်အတွက် ဆိုလျှင် UI thread ၏ object များနှင့် ညှိယူ synchronize လုပ်ရမည် ဖြစ်ပြီး၊ ဤသို့ synchronize လုပ်ရခြင်းသည် သပ်သပ် အလုပ်တစ်ခု ထပ်ဖြစ်လာဦးမည် ဖြစ်သောကြောင့်ပင်။

Android သည် Java ၏ သမားရိုးကျ threading ပုံပိုးမှုသာမက ဤသို့သော ကိစ္စမျိုး များအတွက် အထူးရည်ရွယ် design ထုတ်ထားသည့် AsyncTask ဟူသော utility class တစ်ခုကိုလည်း ဖန်တီးပေးထားပါသည်။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

၆.၁၄.၃ AsyncTask

AsyncTask သည် UI thread သို့ သတင်းပို့ရန် လိုအပ်သည့် ကြာမြင့်သော လုပ်ဆောင်ချက် long operation များကို ကိုင်တွယ်ရာတွင် ထောက်ကူရန်အတွက် ဖန်တီးထားသည့် Android ယန္တရားစနစ်တစ်ခု ဖြစ်သည်။

၄င်းကို အသုံးပြုရန်အတွက် AsyncClass၏ subclassသစ် တစ်ခုကို ဖန်တီးလိုက်ရပေမည်၊ ထို့နောက် doInBackground( ) ၊ onProgressUpdate( ) နှင့် onPostExecute( ) ဟူသော method များကို ပြီးမြောက်စေရမည်။

တနည်းဆိုရလျှင် နောက်ကွယ်တွင် မည်သည်တို့ကို လုပ်ဆောင်မည်နည်း ဟူသောအပိုင်း၊ အပြောင်းအလဲများ ဖြစ်လာလျှင် မည်သည်တို့ကို လုပ်ဆောင်မည်နည်း ဟူသော အပိုင်းနှင့် လုပ်ဆောင်ချက် task ပြီးမြောက်သွားလျှင် မည်သည်တို့ကို လုပ်ဆောင်မည်နည်း ဟူသော အပိုင်းတို့တွင် ကွက်လပ်ဖြည့်ရမည်။

ရှေ့မှ နမူနာကို cloud ဆီသို့ asynchronous post လုမ်းလုပ်သည့် အပိုင်း ထပ်မံဖြည့်စွက်လိုက်မည်။ ရှေ့စာမျက်နှာရှိ code ကို ကြည့်ပါ။

Example 6-5. StatusActivity.java, version 2

```
package com.marakana.yamba1;

import winterwell.jtwitter.Twitter;
import winterwell.jtwitter.TwitterException;
import android.app.Activity;
import android.os.AsyncTask;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

public class StatusActivity2 extends Activity implements OnClickListener {
  private static final String TAG = "StatusActivity";
  EditText editText;
  Button updateButton;
  Twitter twitter;

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.status);

    // Find views
    editText = (EditText) findViewById(R.id.editText);
    updateButton = (Button) findViewById(R.id.buttonUpdate);
    updateButton.setOnClickListener(this);

    twitter = new Twitter("student", "password");
    twitter.setAPIRootUrl("http://yamba.marakana.com/api");
  }

  // Asynchronously posts to twitter
  class PostToTwitter extends AsyncTask<String, Integer, String> { // ❶
    // Called to initiate the background activity
```

```
    @Override
    protected String doInBackground(String... statuses) { // ❷
      try {
        Twitter.Status status = twitter.updateStatus(statuses[0]);
        return status.text;
      } catch (TwitterException e) {
        Log.e(TAG, e.toString());
        e.printStackTrace();
        return "Failed to post";
      }
    }

    // Called when there's a status to be updated
    @Override
    protected void onProgressUpdate(Integer... values) { // ❸
      super.onProgressUpdate(values);
      // Not used in this case
    }

    // Called once the background activity has completed
    @Override
    protected void onPostExecute(String result) { // ❹
      Toast.makeText(StatusActivity2.this, result, Toast.LENGTH_LONG).show();
    }
  }

  // Called when button is clicked
  public void onClick(View v) {
    String status = editText.getText().toString();
    new PostToTwitter().execute(status); // ❺
    Log.d(TAG, "onClicked");
  }

}
```

1. PostToTwitter *class* သည် StatusActivity *class* ၏ အတွင်းငုံ inner class တစ်ခု ဖြစ်သည်။ ၄င်းသည် တဖန် AsyncTask ကို subclass ထပ်ခွဲသည်။ ဤ method များ တွင် AsyncTask သုံးသော data type များကို describe လုပ်ရန် Java generics အား အသုံးပြုပါသည်။ ပထမ data type ကို doInBackground တွင် သုံးပါသည်။ ဒုတိယ data type ကို onProgressUpdate တွင် လည်းကောင်း၊ တတိယ data type ကို onPostExecute တွင် လည်းကောင်း အသီးသီး သုံးထားပါသည်။
2. doInBackground() သည် သီးခြား thread တွင် အမှန်တကယ် အလုပ်လုပ်ရမည်ဖြစ်သော callback ကို ပြဋ္ဌာန်း specify လုပ်သည်။ String... ဟူသော argument သည် generic များ၏ list ထဲတွင် ဤ အတွင်းငုံ inner class အတွက် သတ်မှတ်ခဲ့သော data type များအနက် ပထမ data type ဖြစ်သည်။ အစက်သုံးစက်မှာ ယခု data type သည် Strings ၏ array တစ်ခုဖြစ်ကြောင်း ညွှန်းဆိုသည်။ status တစ်ခုတည်း အတွင်းခ်ိန်သာ pass လုပ်မည် ဖြစ်စေကာမူ ဤအတိုင်းသာ declare လုပ်ရမည်။
3. onProgressUpdate( ) ကို လုပ်ဆောင်ချက် task ၏ အကောင်အထည်ဖော်မှု execution ဖြစ်တိုင်းတွင် call ခေါ်ရသည်။ လက်ရှိအခြေအနေတွင် report လုပ်စရာ လုပ်ဆောင်ချက် progress အထွေအထူး ရှိမနေသေးပါ။ အကယ်၍ ဤအခြေအနေ နေရာတွင် file download တစ်ခုသာ ဖြစ်ပါက ယခု call ၏ ရလဒ်သည် ပြီးမြောက်မှု ရာခိုင်နှုန်း % of completion ဖြစ်ပေမည်။ ထိုကဲ့သို့ ဖြစ်လျှင် data type သည် Integer ဖြစ်မည်။
4. onPostExecute( ) ကိုမူ အလုပ်ပြီးမြောက်သောအခါတွင် call ခေါ်ရမည်။ ဤ method သည် user ကို လုပ်ဆောင်ချက် task ပြီးစီးပြီ ဖြစ်ကြောင်း အသိပေးဖို့ရာ user interface ကို update လုပ်ရန် ဖြစ်သည်။ ဤကိစ္စတွင် screen ပေါ်၌ ခေတ္တ message တစ်ခုကို ပြသ display လုပ်ဖို့ရာ Android UI ၏ Toast ဟူသော feature တစ်ခုကို အသုံးပြုလိုက်ပါသည်။ Toast သည် အမှန်တကယ် message ကို ပြုလုပ်ရန် makeText() ဟူသော static method အား အသုံးပြုသည်ကို သတိပြုပါ။ show( ) ကိုပါ ထည့်ရေးရန်လည်း မေ့၍မဖြစ်ချေ။ ထိုသို့ မဟုတ်ပါက message သည် display ပေါ်လာတော့မည် မဟုတ်သဖြင့် error ရှိနေလျှင် ရှိမှန်းမသိဘဲ

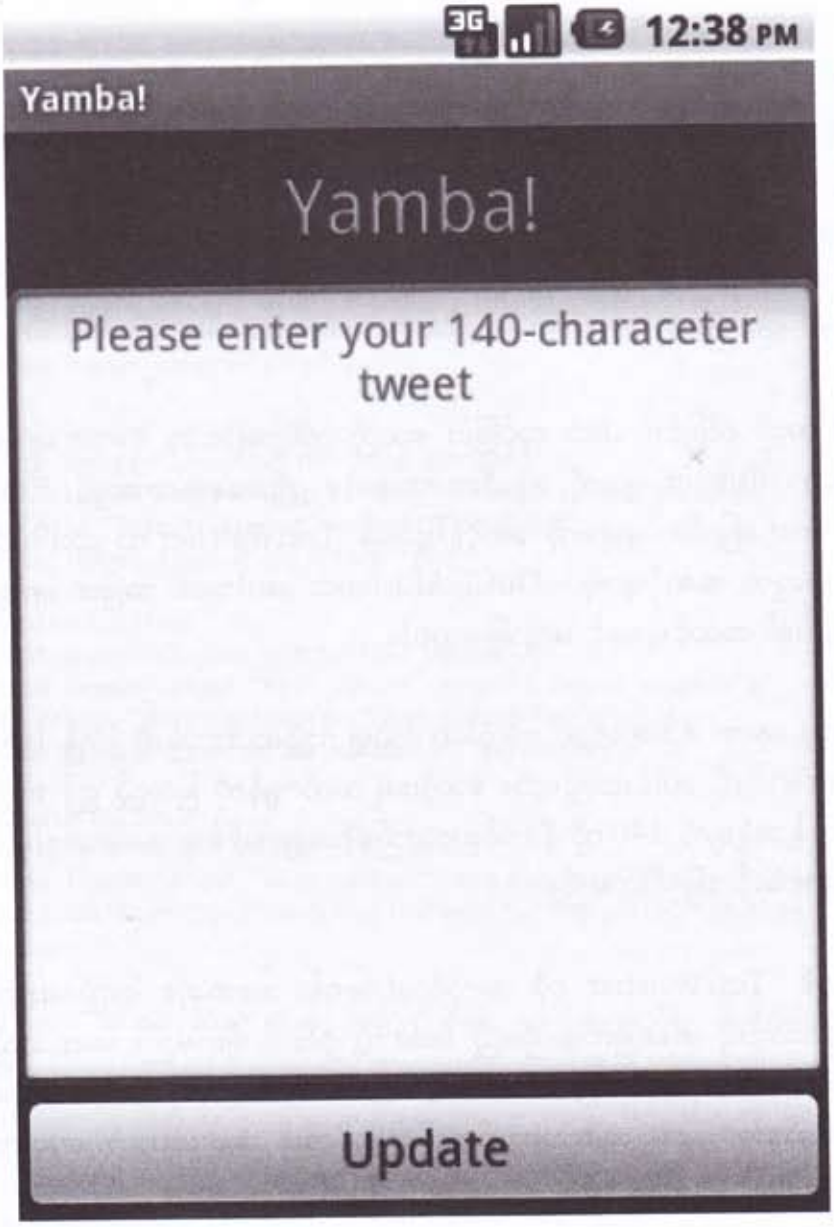
ရှာရခက်သော bug ဖြစ်ပေမည်။ ဤ method မှ ရရှိသော argument မှာ doBackground( ) မှ ပြန်ပို့လိုက်သော တန်ဖိုးဖြစ်မည်။ ယခုဖြစ်ရပ်တွင် String တစ်ခု ဖြစ်မည်။ ယင်းသည်လည်း class definie လုပ်ခဲ့စဉ်က တတိယ generics data type အတိုင်း ဖြစ်ရခြင်းပင်။

5. AsyncTask ကို set up လုပ်ပြီးသည်နှင့် ၄င်းကို သုံး၍ရပေပြီ။ execute() ဖြင့် call ခေါ်မည်။ ထည့်သွင်း pass လုပ်လိုက်သော argument မှာ doInBackground( ) ထဲသို့ ရောက်သွားသည့်အတိုင်းပင် ဖြစ်သည်။

ဤအနေအထားသို့ ရောက်လျှင် Update Status *button* ကို user က နှိပ်လိုက်သောအခါ application ၏ activity သည် AsyncTask ကို သုံးကာ သီးခြား separate thread တစ်ခုကို ဖန်တီးလိုက်မည်၊ ထို့နောက် ဤ thread ပေါ်တွင် ကွန်ယက် လုပ်ဆောင်မှု network operation ကို အမှန်တကယ် အလုပ်လုပ်မည်။ လုပ်ဆောင်မှုပြီးလျှင် Asyn Task သည် (လုပ်ဆောင်ချက် အောင်မြင်/မအောင်မြင် ဆိုသည်ကို user အား အသိပေးဖို့ရာ) Toast မှ ဖန်တီးသော message ဖျတ်ကနဲ ပေါ်လာခြင်းဖြင့် ပင်မ UI thread အား update လုပ်မည်။

ဤလမ်းကြောင်းအားဖြင့် application သည် ပို၍ တုံ့ပြန်မှုကောင်းစေသည်။ user သည် (စာမျက်နှာ – ၁၂၆ တွင် ပြထားသကဲ့သို့သော) error message မျိုးကို တွေ့စရာ မရှိတော့ချေ။

ဤအခြေအနေသို့ ရောက်ပြီဆိုလျှင် application သည် run လိုက်သောအခါ၌ တစ်ဖက်စာမျက်နှာရှိ ပုံအတိုင်း ဖြစ်လာပေတော့မည်။



ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

၆.၁၅ user interface (UI) ပေါ်မှ အခြားသော event များ

click လုပ်လိုက်သော event များအား onClickListener( ) ကိုသုံးခြင်းနှင့် (ခလုပ်နှိပ်လိုက်ချိန်တွင် အလုပ်လုပ်မည့်) ၄င်းထဲမှ onClick( ) *method* ကို ပေးခြင်းအားဖြင့် မည်ကဲ့သို့ ထိန်းချုပ်သည်ကို လေ့လာခဲ့ကြပေပြီ။ အများဆုံးဝင်ဆံ့စာလုံးရေ 140 အနက် ထည့်သွင်းရန် စာလုံးရေ မည်မျှ ကျန်သေးသည်ကို user အား အသိပေးရန် အရေအတွက် counter တစ်ခု ထားလိုသည် ဆိုပါစို့။ ထိုသို့လုပ်နိုင်ရန် အခြား listener ပုံစံတစ်မျိုး ထပ်မံ လိုအပ်ပေမည်။

Android သည် ထိခြင်း၊ click လုပ်ခြင်း စသည့် အမျိုးမျိုးသော event များအတွက် အမျိုးမျိုးသော listener များကို ဖန်တီးပေးထား၏။ ဤအနေအထားတွင် Edit Text *field* ထဲ၌ text ပြောင်းလဲမှုများကို စောင့်ကြည့်ရန် TextWatcher ကို သုံးပါမည်။ ဤ listener အတွက် အဆင့်များမှာ OnClickListener အပါအဝင် အခြား များစွာသော listener များ၏ အဆင့်များနှင့် မကွာခြားလှပါ။

user နေရာမှ နေကာ စဉ်းစားလျှင် စာလုံးရေ မည်မျှ ကျန်သေးသည်ကို ပြရန် layout ထဲတွင် TextView ကို ပေါင်းထည့်မည်။ စာလုံးရေ လက်ကျန်ကို ပြသည့် ဤ text သည် စာလုံးရေ ကန့်သန့်ချက် 140 ကို နီးကပ်လာသည်နှင့်အမျှ အစိမ်းမှ အဝါ၊ အဝါမှ အနီသို့ အရောင်အဆင့်ဆင့် ပြောင်းလဲမည်။

Java ထဲတွင် TextWatcher ကို အလုပ်လုပ်စေရန် အစအဆုံး တည်ဆောက်မည်။ ထို့နောက် စာသားကို user ရိုက်ထည့်မည့် field သို့ ၄င်းကို ချိတ်မည်။ user သည် text ကို အပြောင်းအလဲ လုပ်သည်နှင့် TextWatcher *method* များကို အသက်ဝင်စေမည်။ ရိုက်ထည့်လိုက်သော text အရေအတွက်အတိုင်း လိုက်၍ အရေအတွက် counter ကို update လုပ်မည်။ ရှေ့စာမျက်နှာရှိ code အတိုင်း ဖြစ်သွားပေမည်။

Example 6-6. res/layout/status2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!-- Main Layout of Status Activity -->
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:orientation="vertical" android:layout_width="fill_parent"
  android:layout_height="fill_parent">

  <!-- Title TextView-->
  <TextView android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:gravity="center"
    android:text="@string/titleStatus" android:textSize="30sp"
    android:layout_margin="10dp" />

  <!-- Text Counter TextView ❶ -->
  <TextView android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:layout_gravity="right"
    android:id="@+id/textCount" android:text="000"
    android:layout_marginRight="10dp" />

  <!-- Status EditText  -->
  <EditText android:layout_width="fill_parent"
    android:layout_height="fill_parent" android:layout_weight="1"
    android:hint="@string/hintText" android:id="@+id/editText"
    android:gravity="top|center_horizontal"></EditText>

  <!-- Update Button -->
  <Button android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:text="@string/buttonUpdate"
    android:textSize="20sp" android:id="@+id/buttonUpdate"></Button>
</LinearLayout>
```

1. TextView အသစ် သည် user အဖို့ ရိုက်ရန် စာလုံးရေမည်မျှ ကျန်ရှိသေးသည်ကို ကိုယ်စားပြုသည်။ 140 မှ စထားပြီး user က text ကို ရိုက်ထည့်သည်နှင့် လျှော့သွား မည်။

ယခု TextWatcher *interface* ကို အလုပ်လုပ်စေရန် အစမှ အဆုံး တည်ဆောက်ထားသော အပိုင်းပါ ဖြည့်ပြီးသည့် StatusActivity ၏ code ကို ရှေ့စာမျက်နှာတွင် ကြည့်ပါ။

Example 6-7. StatusActivity.java, final version

```
package com.marakana.yamba1;

import winterwell.jtwitter.Twitter;
import winterwell.jtwitter.TwitterException;
import android.app.Activity;
import android.graphics.Color;
import android.os.AsyncTask;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

public class StatusActivity extends Activity implements OnClickListener,
    TextWatcher { // ❶
  private static final String TAG = "StatusActivity";
  EditText editText;
  Button updateButton;
  Twitter twitter;
  TextView textCount; // ❷

  /** Called when the activity is first created. */
  @Override
  public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.status);

    // Find views
    editText = (EditText) findViewById(R.id.editText);
    updateButton = (Button) findViewById(R.id.buttonUpdate);
    updateButton.setOnClickListener(this);

    textCount = (TextView) findViewById(R.id.textCount); // ❸
    textCount.setText(Integer.toString(140)); // ❹
    textCount.setTextColor(Color.GREEN); // ❺
```

```
  editText.addTextChangedListener(this); // ❻

  twitter = new Twitter("student", "password");
  twitter.setAPIRootUrl("http://yamba.marakana.com/api");
}

// Called when button is clicked
public void onClick(View v) {
  String status = editText.getText().toString();
  new PostToTwitter().execute(status);
  Log.d(TAG, "onClicked");
}

// Asynchronously posts to twitter
class PostToTwitter extends AsyncTask<String, Integer, String> {
  // Called to initiate the background activity
  @Override
  protected String doInBackground(String... statuses) {
    try {
      Twitter.Status status = twitter.updateStatus(statuses[0]);
      return status.text;
    } catch (TwitterException e) {
      Log.e(TAG, e.toString());
      e.printStackTrace();
      return "Failed to post";
    }
  }

  // Called when there's a status to be updated
  @Override
  protected void onProgressUpdate(Integer... values) {
    super.onProgressUpdate(values);
    // Not used in this case
  }

  // Called once the background activity has completed
  @Override
  protected void onPostExecute(String result) {
    Toast.makeText(StatusActivity.this, result, Toast.LENGTH_LONG).show();
  }
}

// TextWatcher methods
public void afterTextChanged(Editable statusText) { // ❼
  int count = 140 - statusText.length(); // ❽
  textCount.setText(Integer.toString(count));
  textCount.setTextColor(Color.GREEN); // ❾
  if (count < 10)
    textCount.setTextColor(Color.YELLOW);
  if (count < 0)
    textCount.setTextColor(Color.RED);
}

public void beforeTextChanged(CharSequence s, int start, int count, int after) { // ❿
}
```

```
    public void onTextChanged(CharSequence s, int start, int before, int count) { // ⓫
    }

}
```

1. StatusActivity သည် TextWatcher ကို implement လုပ်ပြီးဖြစ်ကြောင်း ကြေငြာ declare လုပ်သည်။
2. textCount သည် *Example 6-6* တွင် သတ်မှတ် define လုပ်ခဲ့သော text view ဖြစ်သည်။
3. inflate လုပ်ထားပြီးသည့် layout ထဲတွင် textCount ကို ရှာရမည်။
4. Application တွင် status message ၏ အများဆုံး စာလုံးရေ ဖြစ်သော စာလုံး 140 ကို သတ်မှတ်သည်။
5. textCount *field* ကျန်ရှိသော စာလုံးရေ အရေအတွက်ကို လိုက်ပြီး အရောင် အလိုလို ပြောင်းလဲသွားမည်။ အစိမ်းရောင်ဖြင့် စတင်မည်။ Color ဟူသော class သည် Andriod framework တွင် ပါဝင်ခြင်း ဖြစ်ပြီး၊ Java framework တွင် ပါဝင်သော အရာ မဟုတ်ကြောင်း သတိပြုစေချင်ပါသည်။ ဆိုလိုသည်မှာ android.graphics.Color ကို သုံးခြင်း ဖြစ်သည်၊ java.awt.Color.Color.GREEN သည် ဤ class ထဲတွင် constant အဖြစ် သတ်မှတ်ထားသည့် အရောင်တစ်ခု မဟုတ်ပါ။
6. TextWatcher ကို editText *field* တွင် ချိတ်သည်။ တနည်း ဆိုရလျှင် this အခြေအနေဖြစ်သည့် TextWatcher တွင် ဤ object ကိုယ်တိုင်ကို ပြန်လည် reference လုပ်သော editText ကို call ခေါ်မည်။
7. afterTextChanged ( ) သည် TextWatcher *interface* က ဖန်တီးပေးထားသော method များအနက်မှ တစ်ခု ဖြစ်သည်။ TextWatcher က စောင့်ကြည့်နေလျက် ဖြစ်သည့် ဤ view ထဲတွင် text များ ပြောင်းလဲသွားတိုင်း ဤ method ကို ခေါ်မည်။
8. 140 ဟူသော စာလုံးရေအကန့်အသတ်တွင် စာလုံးရေ မည်မျှ ကျန်သေးသည်ကို တွက်ထုတ်သည်။
9. counter ၏ အရောင်ကို text ကျန်သေးသည့် အရေအတွက်အပေါ် အခြေခံပြီး update လုပ်သည်။ 10 လုံး အထက် ကျန်သေးလျှင် အစိမ်းရောင်ပင် ဖြစ်နေမည်။

10 လုံး အောက် လျော့သွားပြီ ဆိုလျှင် counter သည် အဝါရောင် ပြောင်းသွားမည်။ စာလုံးရေ အကန့်အသတ် ဖြစ်သည့် 140 ထက် ကျော်သွားလျှင်မူ counter သည် အနီရောင် ပြောင်းသွားပေမည်။

10. text ကို replace လုပ်သည့် အဆင့် မတိုင်မီလေးတွင် ဤ method ကို ခေါ်ရမည်။ ယခု ဖြစ်ရပ်တွင် ဤ method ကို တကယ်တမ်း မလိုအပ်ပါ။ သို့သော် TextWatcher *interface* ကို implement လုပ်ခြင်းတွင် အပါအဝင်ဖြစ်နေသည့် အတွက် (မည်သည့် လုပ်ဆောင်ချက် code မှ မပါဘဲ အခွံသက်သက် ဖြစ်နေသည့်တိုင်အောင်) ဤ method ကို call ခေါ်ပေးရပါမည်။

11. onTextChanged( ) ကိုလည်း ယခုအခြေအနေတွင် မသုံးသေးပါ။ သို့သော် ကွက်လပ် အတိုင်း implementation ကိုပင် ဖန်တီး provide လုပ်ပေးရမည်။

run လိုက်သောအခါတွင် application ၌ TextWatcher ၏ အထွက်ကို ရှေ့စာမျက်နှာရှိ ပုံတွင် တွေ့နိုင်ပါသည်။

Android application

ရေးသားဝန်ထမ်းခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

၆.၁၆ Color များနှင့် Graphic များ ပေါင်းစပ်ခြင်း

application သည် ကောင်းစွာက အလုပ်လုပ်နေလေပြီ။ သို့သော် ကြည့်ရသည်မှာ ပသာဒ မဖြစ်သေးချေ။ အရောင်နှင့် graphic အနည်းငယ် ဖြည့်စွက်လျှင် စိုပြည်လာပေမည်။ application တစ်ခုကို စိုစိုပြည်ပြည်ဖြစ်စေရန် Andriod တွင် ပုံပိုးမှုများစွာ ရှိပေသည်။ ထိုအပိုင်းကို ဆက်ပါမည်။

၆.၁၆.၁ Image များ ဖြည့်ခြင်း

အစဦးဆုံးအဆင့် အနေဖြင့် screen တွင် နောက်ခံ background ဖြည့်ပါမည်။ နောက်ခံ background သည် graphic file တစ်မျိုးမျိုး ဖြစ်ပေမည်။ Android တွင် image အားလုံးနီးပါးသည် drawable ဟု အမည်ရှိသော resource folder ထဲသို့ သွားရ၏။ drawable ဟူသော အမည်ပါသည့် folder ကား res အောက်တွင် သုံးခုရှိသည်ကို တွေ့ရပါမည်။

- res\drawable-hdpi သည် ရုပ်ထွက်သိပ်သည်းဆ မြင့်သည့် မြင်ကွင်း high density screen များအတွက် ဖြစ်ပါသည်။
- res\drawable-mdpi သည် ရုပ်ထွက်သိပ်သည်းဆ အလယ်အလတ် မြင်ကွင်း medium density screen များအတွက် ဖြစ်ပါသည်။
- res\drawable-ldpi သည် ရုပ်ထွက်သိပ်သည်းဆ နိမ့်သည့် မြင်ကွင်း low density screen များအတွက် ဖြစ်ပါသည်။

res\drawable ဟု အမည်ပေးပြီး နောက်ထပ် drawable (folder) တစ်ခုကို ဖန်တီးလိုက်ပါမည်။ res (folder) ပေါ်တွင် right-click ပြီး New > Folder ကို ရွေးလိုက်ရုံသာဖြစ်ပါသည်။ အမည်အတွက် drawable ဟု ရိုက်ထည့်လိုက်မည်။ မြင်ကွင်း ရုပ်ထွက်သိပ်သည်းဆ screen density နှင့် လိုက်ကာ ပြောင်းလဲမှု လွတ်ကင်းသော graphic များ

ကို ယခုဆိုလျှင် ဤ folder ထဲတွင် သိမ်းနိုင်ပါပြီ။ developer စိတ်တိုင်းကျ graphic တစ်ခုကို ဤ folder ထဲတွင် background.png အမည်ဖြင့် သိမ်းလိုက်သည် ဆိုပါစို့။

Android သည် graphic file format အတော်များများကို လက်ခံသည် မှန်သော်လည်း portable network graphic (png) သည် ရုပ်ထွက်ပေါ်ရာတွင် ပြုန်းတီးမှု ကင်းသောကြောင့် လည်းကောင်း၊ မူပိုင်မှတ်ပုံတင်မှု patent license ကင်းလွတ် သောကြောင့် လည်းကောင်း png ကို စံထားသော gif ထက်ပို၍ international android developer များ နှစ်သက်ကြသည်။

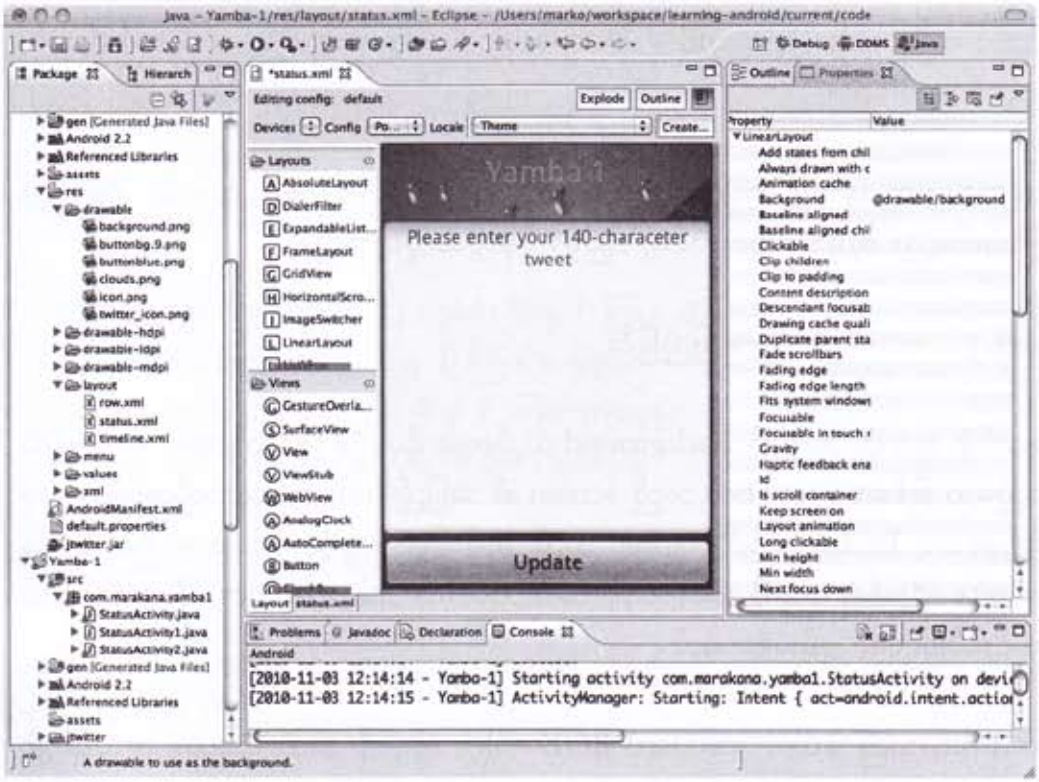
resource အားလုံးသည် Eclipse ၏ စောင့်ကြည့် watch လုပ်ခြင်းအား ခံနေရသည်ကို မမေ့သင့်ချေ။ သို့ဖြစ်၍ res (folder) ထဲသို့ တစ်ခုခု ဖြည့်စွက်လိုက်သည်နှင့် Eclipse သည် Android SDK tools များကို သုံးကာ R class ကို အလိုလျောက် update လုပ်လိမ့်မည်။ ထို့ကြောင့် ယခုအခြေအနေတွင် R.drawable.background ဟူသော အည္ဆန်း reference တစ်ခုကို Java ဘက်မှ လှမ်းသုံးလိုက သုံး၍ ရပေပြီ။ သို့သော် မသုံးသေးပါ။

status activity layout (file) ဖြစ်သော res\layout\status.xml ဖိုင်ကို update လုပ်ခြင်းက နောက်တစ်ဆင့် ဖြစ်ရပေမည်။ ဤနောက်ခံ background file ကို screen ကြီး တစ်ခုလုံး၏ နောက်ခံ graphic အဖြစ် ရှိသွားစေချင်သည် မဟုတ်ပါလား။ ထို့ကြောင့် ဖိုင်ထဲမှ top layout ကို update လုပ်မည်၊ ၄င်း၏ background ကို ယခု background png file အား လှမ်းညွှန်မည်။ ဤသို့ဆိုလျှင် status.xml layout file ကို ဖွင့်ရပေမည်။

background ကို top layout သို့ သွားရောက်ပေါင်းစပ်ရာတွင်လည်း ထုံးစံအတိုင်း နည်း နှစ်နည်း ရှိပါသည်။

Eclipse မှ WYSIWYG Editor ကို သုံးခြင်း

ပထမတစ်နည်းသည်ကား အောက်ပါပုံတွင် ပြထားသည့်အတိုင်း Eclipse ၏ WYSIWYG Editor ကို သုံးလိုက်ခြင်းပင်တည်း။



XML code တွင် တိုက်ရိုက် update လုပ်ခြင်း

နောက်တစ်နည်းမှာမူ XML code ထဲသို့ တိုက်ရိုက် သွားကာ အပြောင်းအလဲများကို ပြုလုပ်ခြင်းပင်။ Eclipse ၏ tool များဖြင့် ပြုလုပ်၍ ရသမျှကို plain-text editior တွင် စာရိုက်ထည့်ခြင်း သက်သက်ဖြင့်လည်း ပြုလုပ်၍ ရသည် ဟူသောအချက်ကို မမေ့သင့်ပါ။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

XML view သို့ ပြောင်းရန်အတွက် ပင်မ window ၏ အောက်ခြေရှိ Layout (tab) ၏ လက်ယာဘက် ဘေးမှ status.xml (tab) ကို click လုပ်လိုက်ရုံသာ ဖြစ်သည်။ standard XML editor ထဲတွင် status.xml ဖိုင် ပွင့်လာပါမည်။

activity ကြီး တစ်ခုလုံးသို့ background resource ဖြည့်ရန်ဆိုလျှင် <LinearLayout> ထဲတွင် android:background="@drawable\background" ဟု code တစ်ကြောင်း ဖြည့်လိုက်ရပါမည်။

ယခုအနေအထားမှ စ၍ အပြောင်းအလဲများ ပြုလုပ်လျှင် XML code ထဲမှသာ တိုက်ရိုက် ပြုလုပ်ပါမည်။ သို့မှသာ ရှင်းလင်းရသည်မှာ ပိုမိုလွယ်ကူပါသည်။

၆.၁၆.၂ Color များ ဖြည့်ခြင်း

နောက်ခံ screen အတွက် background သည်ကား ရှိပါပြီ။ သို့သော် user က စာရိုက်ထည့်မည် ဖြစ်သော text box သည် screen ၏ အပြည့်နီးပါး အုပ်နေသည့်အတွက် ယခု ဖြည့်ခဲ့သော background graphic သည် မည်သို့မျှ မထူးခြားသေးသယောင် ဖြစ်နေသေး၏။ သို့ဖြစ်၍ ၄င်းကို အရောင်နှင့် အကြည်ပေါက် transparency တို့ အနည်းငယ် ပေါင်းစပ်ခြင်းဖြင့် မွမ်းမံရပေမည်။

Android သည် စံထား standard RGB color set ကို အသုံးပြုသည်။ အလိုရှိလျှင် ၄င်းကို Alpha channel တစ်ခုဖြင့် တိုးချဲ့ expand လုပ်နိုင်သည်။ သို့ဖြစ်၍ အရောင်ကို RGB သို့မဟုတ် ARGB အားဖြင့် ဖော်ပြနိုင်သည်။ A သည် အကြည်ပေါက် ပမာဏ Amount of Transparency ၊ R သည် အနီရောင်ပမာဏ amount of Red ၊ G သည် အစိမ်းရောင်ပမာဏ amount of Green ၊ B သည် အပြာရောင်ပမာဏ amount of Blue ဖြစ်သည်။ ဤ အရောင်သုံးရောင်နှင့် လိုလျှင်ထည့်သုံးသည့် အကြည်ပေါက်ပမာဏ တို့ ပေါင်းပြီး အဖြူမှ အနက်ကြား ဖြစ်နိုင်သမျှ အရောင်အားလုံးကို ထုတ်လုပ်ပေးနိုင်သည်။

လုံးဝ အပိတ် opaque အနေအထားမှ လုံးဝ အကြည်ပေါက် fully transparent အနေအထားကြား မည်သည့်အနေအထားမဆိုကိုလည်း ထုတ်လုပ်ပေးသည်။ တန်ဖိုးတစ်ခုချင်းစီတိုင်းအတွက် ဖြစ်နိုင်ချေ 256 မျိုး ရှိသည်။

channel တစ်ခုချင်းစီ ဖြစ်သော A ၊ R ၊ G နှင့် B တို့ အတွက် တန်ဖိုးများကို 0 နှင့် 255 ကြား ကိန်းတစ်ခုခုဖြင့်လည်း ဖော်ပြနိုင်သည်။ ဆယ့်ခြောက်လီစိတ်စနစ် hexadecimal system ဖြစ်သော 0 နှင့် F ကြား တန်ဖိုးတစ်ခုခုကို သုံး၍လည်း ဖော်ပြနိုင်သည်။

ဤစနစ်ဖြင့် ဆိုလျှင် အရောင်တန်ဖိုးများသည် #ARGB ဟု ဖြစ်သွားပေမည်။

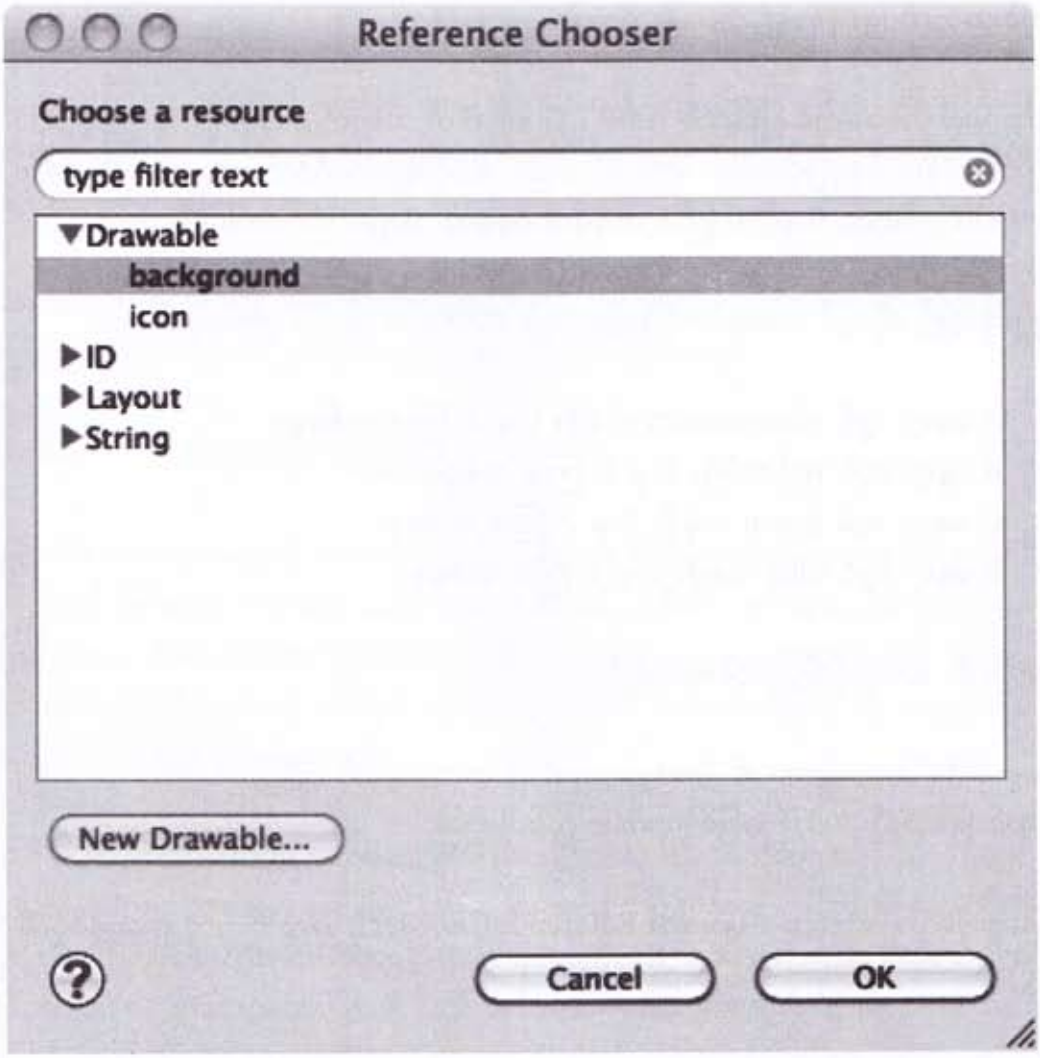
- A နေရာ တွင် transparency တန်ဖိုး 0 မှ F ကြား တစ်ခုခု၊
- R နေရာ တွင် red တန်ဖိုး 0 မှ F ကြား တစ်ခုခု၊
- G နေရာ တွင် green တန်ဖိုး 0 မှ F ကြား တစ်ခုခု၊
- B နေရာ တွင် blue တန်ဖိုး 0 မှ F ကြား တစ်ခုခု၊

အသီးသီး အစားထိုးဖြည့်သွင်းရပေမည်။

ယခု EditText *element* ၏ background ကို အတန်အသင့် အကြည်ပေါက်သော အဖြူရောင် ဖြစ်သည့် #cfff အဖြစ် update လုပ်ပါမည်။

ရှာဖွေလိုသော reference များအား Reference Chooser မှ အလွယ်တကူ ရှာဖွေနိုင်ပုံကို ရှေ့စာမျက်နှာတွင် တွေ့နိုင်ပါသည်။

နောက်တဆင့်အဖြစ် ခေါင်းစဉ်စာသား title text ၏ အရောင်ကိုလည်း TextView မှ textColor *property* ကို ပြောင်းလဲခြင်းအားဖြင့် update လုပ်ပါမည်။ ဥပမာ အနေနှင့် အဖြူရောင် ဆိုလျှင် ကောင်း၏ဆိုပါစို့။ အဖြူကို ဆယ့်ခြောက်လီစိတ်စနစ်ရေးထုံးအားဖြင့်


Reference Chooser
Choose a resource
type filter text
Drawable
background
icon
ID
Layout
String
New Drawable...
Cancel
OK

#fff (A ၊ R ၊ G နှင့် B အားလုံးကို အဆုံးတင်ထားသည့် တန်ဖိုးများ အစု) အဖြစ် ပြဋ္ဌာန်း specify လုပ်နိုင်သည်။ နောက်တစ်နည်းမှာ @android:color/white ဟု ရေးနိုင်သည်။ android: ဟူသော အပိုင်းမှာ Android မောင်းနှင်စနစ် operating system ၏ ရင်းမြစ်များအစုအဝေး set of resources ကို လှမ်းညွှန်း ခေါ်ဆိုခြင်း ဖြစ်သည်။ ယခု အနေအထားအရ ဆိုပါက သတ်မှတ်ထားနှင့်ပြီး အဖြူရောင် pre-defined color (white) ကို လှမ်းညွှန်း ခေါ်ဆိုခြင်း ဖြစ်သည်။

update များ ထပ်ဖြည့်ပြီးနောက် လက်ရှိအနေအထားအလိုက် ပြောင်းလဲသွားသော status.xml ၏ code ကို ရှေ့စာမျက်နှာတွင် တွေ့နိုင်ပါသည်။

1. res\drawable (folder) ထဲမှ background.png ဂရပ်ဖစ်ဖိုင်ကို ပင်မ mail layout တွင် နောက်ခံအဖြစ် သတ်မှတ်ရန် လှမ်းညွှန်း point လုပ်သည်။
2. title text ၏ color ကို system ရှိ color resources များထဲမှ white အဖြစ် သတ်မှတ်ထားသော definition အား အရောင်အဖြစ် သတ်မှတ်ရန် လှမ်းညွှန်း point လုပ်သည်။
3. EditText ၏ နောက်ခံဧရိယာ background area ကို ဆယ့်ခြောက်လီစိတ်စနစ် hexadecimal ARGB တန်ဖိုးတစ်ခု ဖြစ်သော #cfff ဟု ပြဋ္ဌာန်း specify လုပ်ခြင်း ဖြင့် အကြည်ပေါက်အဖြူရောင်ကို သတ်မှတ်သည်။

ယခုဆိုလျှင် activity ထဲရှိ view(widget) အသီးသီး၏ အမျိုးမျိုးသော properties များ အတွက် အရောင်များ ပြဋ္ဌာန်း specify လုပ်သည့် နည်းလမ်း အသီးသီးကို ဆွေးနွေးခဲ့ပြီး ဖြစ်ပေသည်။ မိမိကိုယ်တိုင် ချဲ့ထွင်စဉ်းစား စမ်းသပ်ခြင်းဖြင့် မိမိ၏ application UI ကို မိမိစိတ်ကြိုက် တည်ဆောက်နိုင်ပါမည်။

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Example 6-8. res/layout/status.xml

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

<!-- Main Layout of Status Activity -->
<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"><!--❶-->

  <!-- Title TextView-->
  <TextView android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:gravity="center"
    android:text="@string/titleStatus" android:textSize="30sp"
    android:layout_margin="10dp" android:textColor="@android:color/white" /><!--❷-->

  <!-- Text Counter TextView -->
  <TextView android:layout_width="wrap_content"
    android:layout_height="wrap_content" android:layout_gravity="right"
    android:id="@+id/textCount" android:text="000"
    android:layout_marginRight="10dp" />

  <!-- Status EditText  -->
  <EditText android:layout_width="fill_parent"
    android:layout_height="fill_parent" android:layout_weight="1"
    android:hint="@string/hintText" android:id="@+id/editText"
    android:gravity="top|center_horizontal" android:background="#cfff" /><!--❸-->

  <!-- Update Button -->
  <Button android:layout_width="fill_parent"
    android:layout_height="wrap_content" android:text="@string/buttonUpdate"
    android:textSize="20sp" android:id="@+id/buttonUpdate" />

</LinearLayout>
```

၆.၁၇ User Interface ကို Optimize လုပ်ခြင်း

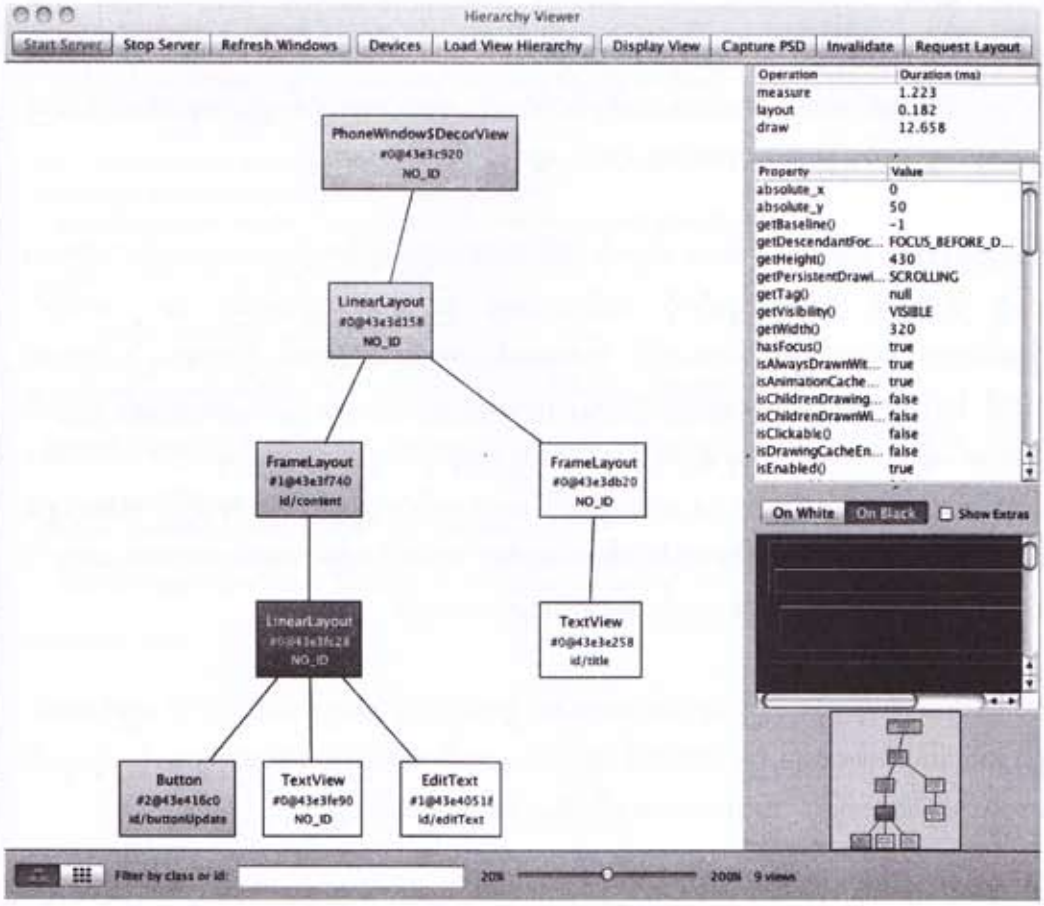
Android application တစ်ခုတွင် user interface (UI) သည် စနစ်၏ ရင်းမြစ်များ system resources ကို အသုံးအပြုဆုံး အစိတ်အပိုင်းတစ်ခု ဖြစ်သည်။ အရိုးရှင်းဆုံးသော screen တစ်ခုကို တည်ဆောက်မည် ဆိုလျှင်ပင် application သည် resources ထဲမှ XML ဖိုင်ကို ပျံ့ကားစေရမည်။ element တစ်ခုချင်းစီအတွက် Java object သစ်တစ်ခု ကို ဖန်တီးရမည်၊ ၄င်းအတွက် properties များကို သတ်မှတ် assign လုပ်ရမည်။ ပြီးလျှင် screen ပေါ်၌ widget တစ်ခုစီတိုင်းကို ဆွဲသားရမည်။ ဤအရာအားလုံးသည် တွက်ချက်မှု စက်ဝန်း computing cycle များစွာကို သုံးပေမည်။

ထိုသို့ဖြစ်ရကား optimization ကိစ္စကို အမြဲတစေ ထည့်သွင်း စဉ်းစားရပေမည်။ screen ပေါ်ရှိ widget အရေအတွက်ကို ကန့်သတ်ရန် ကြိုးစားကြည့်ရမည်။ အထူးအားဖြင့် လိုအပ်သော အထွက်မြင်ကွင်းအတိုင်း ဖြစ်စေရန် (widget များကို ငုံထားသည့် layout များကို layout ထဲတွင် ပြန်စုထားရသော) nested layout များကို သုံးထားလျှင် ပိုရှုပ်ပင်လိုအပ်ပေမည်။ layoutပိုင်းမှ စဉ်းစားနည်းသည် ရံဖန်ရံခါတွင် ထိန်းချုပ်ရခက်တတ်ပေ၏။ loop ထဲတွင် မလိုအပ်သော object များကို nest လုပ်မိလျှင် (ဥပမာအားဖြင့် data များကို row လိုက် တစ်တန်းပြီး တစ်တန်း display လုပ်ခြင်းမျိုး၊) widget အရေအတွက် မတရား များပြားလာပြီး UI သည် နှေးဆင်းသွားပေမည်။

ယေဘုယျအားဖြင့် ဆိုရလျှင် application ၏ ဖွဲ့စည်းပုံကို အတွင်းနက် deep မဖြစ်စေဘဲ၊ ပျံ့ကျယ် flat ဖြစ်စေရမည်။ nested layout များကို relative layout များနှင့် လဲလှယ်အစားထိုးခြင်းအားဖြင့် application ကို flat ဖြစ်စေနိုင်ပါသည်။

၆.၁၈ Hierarchy Viewer

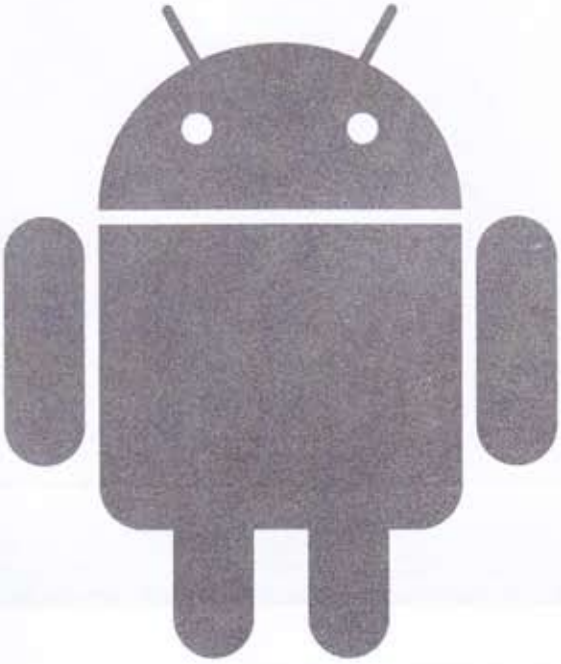
Android SDK တွင် တပါတည်း ပါလာသော အလွန်အသုံးဝင်သည့် tool တစ်ခုမှာ Hierarchy Viewr ဖြစ်ပါသည်။ SDK\tools (folder) ထဲတွင် ရှိပါသည်။ အောက်ပါ ပုံအတိုင်း ဖြစ်ပါမည်။



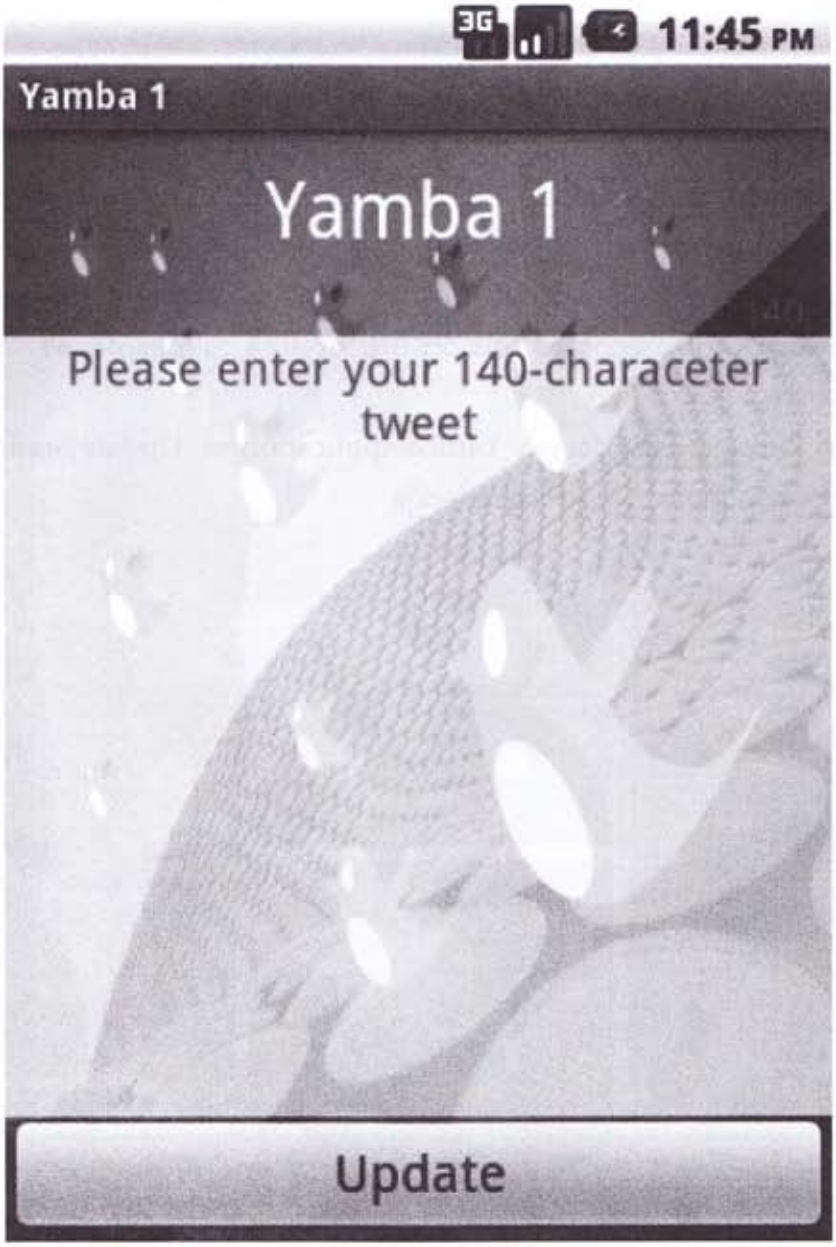
၆.၁၉ နိဂုံး

ဤအခန်းသည် Android application တစ်ခုကို code ပိုင်း စတင် အကောင်အထည် ဖော်သော အခန်း ဖြစ်သကဲ့သို့ Android application developing ကို တကယ် စတင် ထိုးထွင်း လေ့လာသည့် အပိုင်းလည်း ဖြစ်သဖြင့် ကဏ္ဍခွဲများ လွန်စွာ များပြားသကဲ့သို့၊ အသုံးချသော သဘောတရား များ၊ ချိန်ထိုးသုံးသပ်ရမှုများ၊ လေ့လာမှတ်သားရခြင်းများ၊ စဉ်းစားစရာများ များစွာ ပါရှိပါသည်။ သို့ဖြစ်၍ တစ်ဆင့်ချင်း အေးဆေးစွာ လေ့လာပြီး ကောင်းစွာ သဘောပေါက်သည်အထိ အထပ်ထပ် ဖတ်ရှုပါရန် အကြံပြုပါရစေ။

လက်ရှိအထိ ပြီးသမျှအနေအထားတွင် Yamba application ၏ Update Status (UI) သည် ရှေ့စာမျက်နှာရှိပုံအတိုင်း ဖြစ်သွားပါမည်။



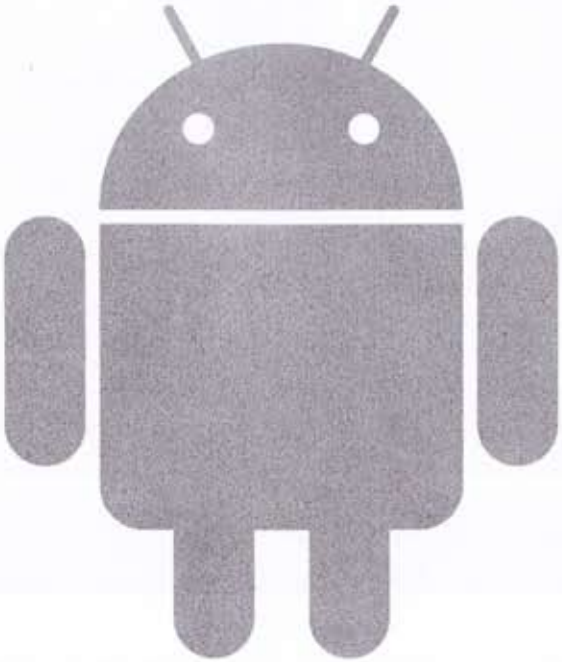
ဝင်းဟိန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)



Android application

ရေးသားဝန်ထမ်းခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

ဝင်းဟန်:
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)



Android application

ရေးသားပန်တီးခြင်းဆိုင်ရာ အခြေခံသဘောတရားများ

Android application များ ရေးသားဖန်တီးခြင်း အဆင့်မြင့် သဘောတရားများ

ဒုတိယပိုင်း အဖြစ် ဆက်လက်ထွက်ရှိမည် ဖြစ်သည့် Android appplication များ ရေးသားဖန်တီးခြင်း အဆင့်မြင့် သဘောတရားများ အပိုင်းတွင်

Preferences များ၊ Filesystem စနစ်၊ Option Menu နှင့် Intent များ

ဤအပိုင်းသည် Preference Screen များ တည်ဆောက်ပုံနှင့် လုံးဝ သက်ဆိုင်သည်။ ယခု နမူနာထား တည်ဆောက်လျက် ရှိသည့် Yamba application ကိုပင် ဆက်လက် နမူနာထား ဆွေးနွေးသွားမည် ဖြစ်ပါသည်။ ဤအပိုင်းပြီးလျှင် Yamba application တွင် screen နှစ်ခု ဖြစ်သွားမည်။ တစ်ခုသည် Status Updates Screen ဖြစ်ပြီး၊ နောက်တစ်ခုသည် Preferences Screen ဖြစ်မည်။ ဤအဆင့်သို့ ရောက်လျှင် Yamba သည် user အမျိုးမျိုးအတွက် စိတ်ကြိုက် ပြုပြင်ပြောင်းလဲ configure လုပ်ယူနိုင်သော အနေအထား ဖြစ်သွားမည်။ ဤအဆင့်တွင် ဖန်တီးရမည့် အစိတ်အပိုင်း element များမှာ activity များ၊ menu system နှင့် intent များ၊ ထို့ပြင် ဖိုင်သိုမှီးမှုစနစ် file system တို့ ဖြစ်လာပေမည်။ ဤ အပိုင်းတွင် ကဏ္ဍခွဲများမှာ တစ်ဖက်ပါအတိုင်း ဖြစ်မည်။

ဝင်းဟိန်း

B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Activity

ပထမဦးဆုံးအဖြစ် Android ၏ building block component တစ်ခုဖြစ်သော Screen ကို ဖန်တီးလိုက်မည်။ ဤသည်ပင်လျှင် activity ဖြစ်သည်။ ပါဝင်သော အဆင့်များကို မြင်တွေ့လျှင် Screen သစ်များ ဖန်တီးရန် မည်သည်တို့ လိုအပ်သည်ကို နားလည်ပါမည်။

Menu system နှင့် Intent များ

နောက်တစ်ဆင့်အားဖြင့် ထို Screen ကို ရောက်အောင် သွားနိုင်မည့် နည်းလမ်းကို လိုအပ်မည်။ သို့ဖြစ်၍ Menu system တစ်ခုကို စတင်မည်။ ဤ Menu system ၏ အလုပ်လုပ်ပုံကို လေ့လာမည်။ တပါတည်းအဖြစ် activity တစ်ခုကို ဖွင့်ရန် intent များ မည်သို့ ပေးပို့သလဲ ဆိုသည်နှင့် intent များ၏ အကြောင်းကိုလည်း လေ့လာမည်။

ဖိုင်သိုမှီးမှုစနစ် Filesystem

နောက်ဆုံးအနေဖြင့် Android မောင်းနှင်စနစ်သုံး ယေဘုယျကြိုးမဲ့မိုဘိုင်း ကိရိယာတစ်ခုရှိ ဖိုင်သိုမှီးမှုစနစ် filesystem ကို လေ့လာမည်။ ဤမှတဆင့် မောင်းနှင်စနစ် operating system ကြီး တစ်ခုလုံးကို မည်ကဲ့သို့ ပေါင်းယှက်ဆက်စပ် တည်ဆောက်ထားသည်ကို လည်းကောင်း၊ Android ၏ လုံခြုံရေးစနစ် security ကိုလည်းကောင်း ပိုမိုသိရှိပါမည်။

Android ၏ ဝန်ဆောင်မှုလုပ်ဆောင်ချက် Services များ

ဤအပိုင်းသည် background services နောက်ကွယ်လုပ်ဆောင်ချက်များကို စတင်ထိတွေ့ရမည့် အပိုင်း ဖြစ်သည်။ တိမ်တိုက်ဝန်ဆောင်မှုဆီသို့ သတ်မှတ်ထားသော အချိန်အလိုက် ချိတ်ဆက်ပြီး friends များ၏ status update များကို ရယူခြင်း ဖြစ်သည်။

ဝန်ဆောင်မှု လုပ်ဆောင်ချက် Services များ

Android services များသည် အလွန်အသုံးဝင်လှသော building block များ ဖြစ်သည်။ user interface (UI) မလိုအပ်ဘဲ နောက်ကွယ်တွင်သာ process တစ်ခုကို run နေစေ၏။ Yamba သည် တိမ်တိုက်ဝန်ဆောင်မှုဆီသို့ သတ်မှတ်ထားသော အချိန်အလိုက် ချိတ်ဆက်ပြီး data ကို ဆွဲယူရမည့် သဘာဝရှိသောကြောင့် ဤ feature သည် ယခုနမူနာ project ဖြစ်သည့် Yamba အတွက် လုံးဝအံဝင်ခွင်ကျပင်။ တပါတည်း အဖြစ် ဤအပိုင်းတွင် နောက်ကွယ်ဝန်ဆောင်မှု background services များကို အသုံးချရာ၌ ထည့်သွင်းစဉ်းစားရမည့် multithreading ဆိုင်ရာများကိုလည်း လေ့လာသိရှိပါမည်။

Applicatin object များ

ဤ အဆင့်တွင်မူ code ပိုင်းတွင် ထပ်နေသည်များ ရှိလာပြီး နမူနာ project ၏ application သည် သပ်ရပ်လှပကျစ်လစ်ခြင်း မရှိတော့သည်ကို သတိထားမိလာမည်။ သို့ဖြစ်၍ Yamba ကို အစိတ်အပိုင်းငယ်အသီးသီး အဖြစ် ခွဲထုတ်နိုင်သည့် Application ဟူသော object တစ်ခုကို စတင်ထိတွေ့ပြီး အချိုးအစားချိန်ရ လွယ်ကူစေပါမည်။

Database များဖြင့် အလုပ်လုပ်ခြင်း

ယခုအခါ updater service မှ data များ ရရှိနိုင်သည့် အနေအထားသို့ ရောက်ပြီ။ ထို့ကြောင့် သိုမှီးစရာ လိုအပ်လာပြီ။ သို့ဖြစ်၍ Android ၏ database support များကို စတင်ထိတွေ့ပါမည်။

SQLite နှင့် ၄င်းအား Android ၏ ပံ့ပိုးမှု

Android တွင် SQLite ဟုခေါ်သော built-in database တစ်ခု တစ်ပါတည်း ပါသည်။ ထိုမျှသာမက Android ၏ framework သည် SQLite ကို developer များ အသုံးပြုရ လွယ်ကူစေသည့် API များကိုလည်း စုံလင်စွာ ဖန်တီးထည့်သွင်းပေးထားပါသည်။ သို့ဖြစ်၍ ဤအပိုင်းတွင် SQLite နှင့် ၄င်းအတွက် API အား မည်သို့ အသုံးချမည်ကို လေ့လာပါမည်။ SQL ဂုရုတစ်ဦး ဖြစ်နေရန်မလိုပါ။ အခြေခံမျှလောက် သိထားလျှင် များစွာ အဆင်ပြေ ချောမွေ့ပါမည်။

Code ကို ထပ်မံ၍ စိတ်ပိုင်း refactor လုပ်ခြင်း

ဤအဆင့်တွင် code ပိုင်းကို ထပ်မံပြီး refactor လုပ်ရန် အခွင့်ကြုံလာပြန်ပါဦးမည်။ ဤတစ်ကြိမ် refactor လုပ်ခြင်းသည် ပိုမိုထိရောက်ပြီး နောက်ပိုင်းအတွက် ပိုအသုံးတည့်သည်ကို တွေ့ရပါမည်။

အပိုင်း (၅) List များနှင့် Adapter များ

List များနှင့် Adaptor များသည် Android တွင် user interface (UI) element များ ထက် ပိုမိုစနစ်ကျသော အထောက်အကူများ ဖြစ်ပေသည်။ သေးငယ်လှသော UI ကို ကြီးမား များပြားလှသော data set များဖြင့် ထိရောက်ပြီး အချိုးအစားပြုပြင်နိုင်သော အနေအထားနှင့် ချိတ်ဆက်ရာတွင် powerful ဖြစ်သော component များကို ဖြစ်စေသည်။ Yamba ကို ကာလကြာစွာ သုံးသော်လည်း လုပ်ဆောင်ရည် မကျဆင်းအောင် ထိန်းထားနိုင်စေသည်။

အထက်တွင် ဆိုခဲ့သည့် အတိုင်း updater service က ဆွဲချခဲ့သော data သည် လောလောဆယ် အနေအထားတွင် database ထဲသို့ အကုန် ရောက်နေလေပြီ။ သို့သော် ၄င်းကို ကြည့်ရန်နည်းလမ်း မရှိသေးချေ။ သို့ဖြစ်၍ online တွင် friend များ အစီအရီ တင်နေသော post များကို user မြင်တွေ့ရစေရန် နည်းလမ်းတစ်ခုဖြစ်သည့် (နမူနာ project ဖြစ်သော Yamba ၏ အဓိက လိုအပ်ချက်လည်း ဖြစ်သည့်) Timeline Activity ကို စတင်ထိတွေ့ပါမည်။

Timeline Activity

တတိယနှင့် နောက်ဆုံးဖြစ်သော ဤ Activity ကို အဆင့်တစ်ခုတည်းမကပဲ အဆင့်ဆင့် ခွဲကာ လုပ်ဆောင်ပါမည်။ ပထမအဖြစ် Android platform နှင့် ပတ်သက်ပြီး လက်ရှိ အထိ သိခဲ့ပြီးသော သဘောတရားကို ပေါင်းစပ်အသုံးချကာ ဖန်တီးကြည့်မည်။ ထို့နောက် ထို design ကို မွမ်းမံမည်။ စဉ်းစားရမည့် အဓိက အချက်မှာ လက်တွေ့တွင် (ဤ application မျိုးမှာ အသုံးပြုသူများပြားလှသောကြောင့်) တက်လာသော data အင်အား၏ ဝန်ကို ခံနိုင်စေရန်၊ ပြိုတက်လာသော data တို့ကို UI ပေါ်တွင် စု တင်နိုင်ရန် List များနှင့် Adapter များကို ထည့်သွင်းချိတ်ဆက် သုံးစွဲမည်။

ဝင်းဟိန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

Code ကို ထပ်မံ၍ စိတ်ပိုင်း refactor လုပ်ခြင်း

ဤအဆင့်တွင် code ပိုင်းကို ထပ်မံပြီး refactor လုပ်ရန် အခွင့်ကြုံလာပြန်ပါဦးမည်။ ဤ တစ်ကြိမ် refactor လုပ်ခြင်းဖြင့် user အား screen အသီးသီးကို အပြန်အလှန် သွားရာတွင် ပိုမိုချောမွေ့အောင် ဖန်တီးပေးနိုင်သွားခြင်းဖြင့် application အပေါ် ပိုမို ယုံကြည်သွားစေပြီး၊ developer အတွက်လည်း ရှေ့ဆက်ရမည့် အနေအထားအတွက် code များကို စီမံခန့်ခွဲရာတွင် ပိုမို လွယ်ကူသွားစေပါမည်။

Android application များ ရေးသားဖန်တီးခြင်း အဆင့်မြင့် သဘောတရားများ

ဒုတိယပိုင်း အဖြစ် ဆက်လက်ထွက်ရှိမည် ဖြစ်သည့် Android appplication များ ရေးသားဖန်တီးခြင်း အဆင့်မြင့် သဘောတရားများ စာအုပ်တွင် ပါဝင်မည့် အခန်းများမှာ အောက်ပါအတိုင်း ဖြစ်ပါသည်။

'အခန်း (၁) – Reference များ၊ File စနစ်၊ Options menu နှင့် intent များ ' သည် application developer တစ်ယောက်အဖို့ အလုပ်လုပ်ရာတွင် ပိုမိုလွယ်ကူသွားစေသော မောင်းနှင်စနစ် operating system ဆိုင်ရာ feature များအကြောင်း ဖြစ်ပါသည်။

'အခန်း (၂) – Service များ' သည် နောက်ကွယ်လုပ်ဆောင်ချက် background task များကို process လုပ်ပေးနိုင်သော Android service တစ်ခုအား တည်ဆောက်ခြင်းအကြောင်း ဖြစ်ပါသည်။

'အခန်း (၃) – Database မိတ်ဆက်'သည် Android တွင် တပါတည်း ပါသော အင်ယင်စား SQL (SQLite) database အကြောင်းနှင့် ကိုယ်ပိုင် application တွင် data များကို ထိန်းသိမ်းရန် ၄င်းအား အသုံးချပုံအကြောင်း ဖြစ်ပါသည်။

'အခန်း (၄) – List များနှင့် Adapter များ' သည် သေးငယ်သော screen မြင်ကွင်းများ အတွင်း များပြားလှသော data အစုအဝေးများကို ထိရောက်စွာ ချိတ်ဆက်အသုံးပြုနိုင်သည့် Android ၏ အရေးပါသော feature များကို ခြုံငုံဆွေးနွေးထားခြင်း ဖြစ်ပါသည်။

'အခန်း (၅) – 'Broadcast Receiver များ'သည် စနစ်က သတ်မှတ်ထားသည့် သတင်းပေး ပို့မှု system-defined messages များ၊ ပရိုဂရမ်ရေးသားသူက သတ်မှတ်သည့် သတင်းပေး

ဝင်းဟန်း
B.C. Tech (UCS,Y); D.B.S (Y.I.Eco); Dip Info-Tech (NCC, UK)

ပို့မှ user-defined messages များ စသည့် အမျိုးမျိုးသော message များကို တုံ့ပြန်သည့် Android ၏ public-subscribe mechanism ယန္တရားစနစ်အား မည်သို့ အသုံးပြုရမည် ဆိုသည်ကို ရှင်းလင်းထားခြင်း ဖြစ်သည်။

'အခန်း (၆) – Content Provider များ'သည် application များအချင်းချင်း အတွင်း data များ မျှဝေ share လုပ်ရန်အတွက် content provider တစ်ခုအား မည်သို့ design ထုတ် တည်ဆောက်ရပုံကို ဆွေးနွေးထားခြင်း ဖြစ်သည်။ နမူနာထား ဆွေးနွေးလျက်ရှိသော Yamba application အရဆိုလျှင် content provider ကို သုံးကာ Home Screen ပေါ်ရှိ Yamba ၏ application widget တွင် data များ ဖော်ပြ display လုပ်နိုင်စေရန် လက်တွေ့ တည်ဆောက်ပြသ ရှင်းလင်းထားခြင်း ဖြစ်သည်။

'အခန်း (၇) – System Service များ' သည် application developer တစ်ယောက်အနေ ဖြင့် ချိတ်ဆက်သုံးစွဲနိုင်သည့် အမျိုးမျိုးသော Android စနစ်၏ဝန်ဆောင်မှု system service များကို မိတ်ဆက်ရှင်းလင်းထားခြင်း ဖြစ်သည်။

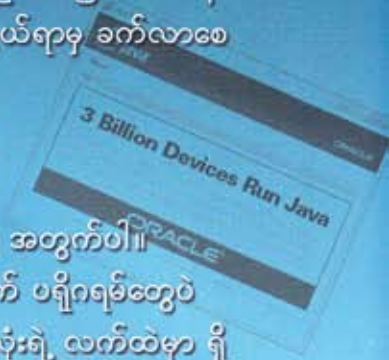
.....

သိလိုသည်များ၊ ဝေဖန်အကြံပြုလိုသည်များ ရှိပါက winhein.myanmar@gmail.com သို့ email ပေးပို့ဆက်သွယ်နိုင်ပါသည်။

.....

ဒီစာအုပ်ဟာ ဘာအကြောင်းရေးထားတဲ့ စာအုပ်လဲ ?

Android လို့ခေါ်တဲ့ မောင်းနှင်စနစ် သုံးသော မိုဘိုင်းဖုန်းတွေအတွက် ပရိုဂရမ်တွေ၊ application တွေ ရေးနည်းကို ရှင်းပြထားတဲ့ စာအုပ်ပါ။ programming စာအုပ်ပေါ့။ Android အတွက် application တွေ ရေးသားနည်း အခြေခံသင်တန်းတစ်ခုမှာ သင်ရမယ့် သင်ခန်းစာတွေကို တဆင့်ပြီး တဆင့် လွယ်ရာမှ ခက်လာစေသည့် အစီအစဉ်နဲ့ စုစည်းထားတဲ့ စာအုပ်ပါ။



ဒီစာအုပ်ဟာ ဘယ်သူတွေအတွက် ရည်ရွယ်သလဲ ?

အဓိက ကတော့ programming ကို စိတ်ပါဝင်စားတဲ့သူတွေ အတွက်ပါ။ ဒီနေ့ခေတ်မှာ programming ဆိုရင် ကွန်ပျူတာတွေ အတွက် ပရိုဂရမ်တွေပဲ ဖန်တီးရေးသားတဲ့ကိစ္စ မဟုတ်တော့ဘူးလေ။ အဓိက လူအားလုံးရဲ့ လက်ထဲမှာ ရှိလာနေတာက မိုဘိုင်းဖုန်းတွေ ဖြစ်တော့ မိုဘိုင်းဖုန်းတွေအတွက် ပရိုဂရမ်တွေ ရေးခြင်းဟာလည်း ခေတ်ရဲ့ လိုအပ်ချက် တစ်ရပ် ဖြစ်နေပါပြီ။ ဒါကြောင့် မိုဘိုင်းဖုန်းများအတွက် ပရိုဂရမ်ဖန်တီးရေးသားခြင်း အတတ်ပညာဟာလည်း သတင်းဆက်သွယ်ရေး နည်းပညာ Information Communicarion Technology (ICT) ရဲ့ ပင်မအကိုင်းအခက်ကြီး တစ်ခု ဖြစ်လာနေပါပြီ။ ဒီအတွက် လိုအပ်ချက်တစ်ရပ်ကို တတ်စွမ်းသလောက် ဖြည့်ဆည်းတာပါ။ ပရိုဂရမ်ရေးသားခြင်းနှင့် အသက်မွေးရန် ကြိုးစားနေသူတွေ၊ electronic engineering ပိုင်းဆိုင်ရာကို စိတ်ဝင်စား လေ့လာလိုက်စားနေသူတွေ၊ နည်းပညာသစ်ရဲ့ အားသာချက်ကို အသုံးချပြီး စွန့်ဦးတီထွင်လိုသူတွေ၊ အင်ဂျင်နီယာပညာရပ်နဲ့ ကွန်ပျူတာ ပညာရပ်ဆိုင်ရာ ကျောင်းသား သင်တန်းသားတွေအတွက် အဓိက ရည်ရွယ်ပါတယ်။

ဒီစာအုပ်ကို ဖတ်ပြီး ဘာအကျိုးကျေးဇူးတွေ ရသွားမလဲ ?

ဒီစာအုပ်ကို ပြုစုရခြင်းရဲ့ အဓိက ဦးတည်ချက် ဖြစ်သော Android application developing ဆိုင်ရာ အခြေခံသဘောတရားတွေကို ကိုယ်တိုင်ကိုယ်ကျ ထိုးထွင်းနားလည်သွားမယ်။ တဆက်တည်းမှာ Java နဲ့ C/C++ တို့ကို ကွန်ပျူတာမဟုတ်သော အခြား smart electronic device တွေ၌ ဘယ်လို ဆက်စပ်အသုံးချတယ်၊ Linux သည် Java တို့ C/C++ တို့နှင့် ဘယ်လို ယှဉ်တွဲအလုပ်လုပ်တယ် ဆိုတာတွေကိုပါ ချိတ်ဆက်လေ့လာသိရှိမယ်။

Design-thargyi