

software pilots

TRIFORK.

Monetizing Android: In-app billing and Ads

Trifork GeekNight

24. November 2011

About me

- Christian Melchior
- Android Developer at Trifork
- This talk is based on the experience gained during the development of a hobby project: Meter Readings.
- Email: cme@trifork.com

Program

- Ads (approx 40 min)
- Break (Sandwiches) (approx. 15 min)
- In-app billing (approx. 45-60 min)

Why not just make paid apps?

- 20% of free apps and 80% of paid apps has been downloaded less than 100 times.
- 52% of total revenue from top200 games come from In-app purchase, up from 8% in 2010.

ADS

Ads 101

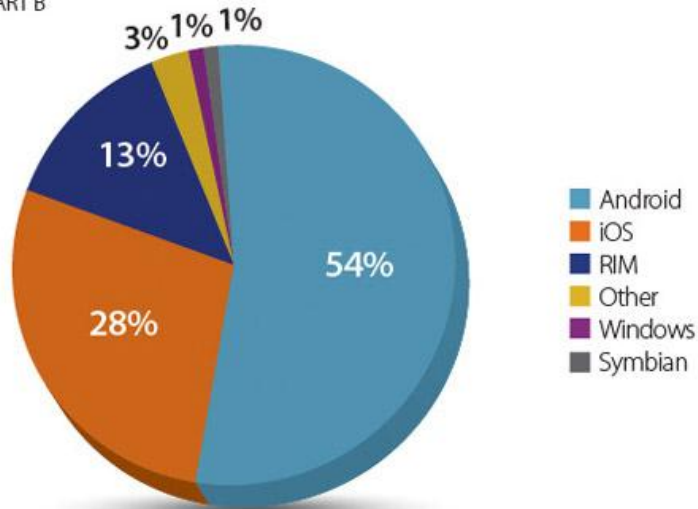
- **Fillrate** How many add requests return an ad?
- **CTR** Click-through-rate. The percentage of ads clicked
- **eCPM (RPM)** Effective cost pr. 1000 impressions.

$$\frac{\text{Total Earnings}}{\text{Impressions}} \times 1000 = \text{eCPM}$$

Ad numbers

Connected Device & Smartphone OS Mix

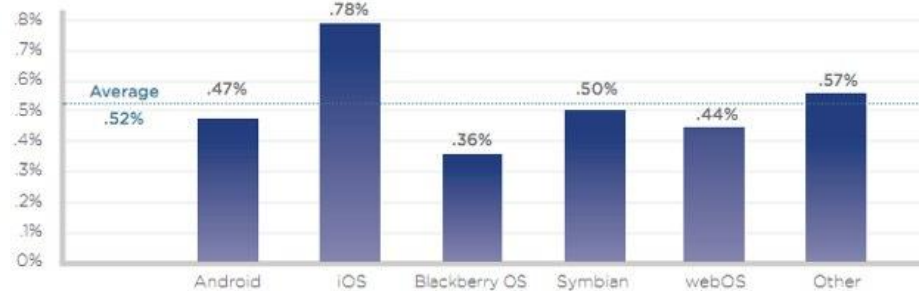
Ranked by Impressions
CHART B



Source: Millennial Media, 8/11.
Other includes webOS, Danger, Nokia OS, Palm OS.



Mobile Operating System Click-Through Rates



Source: JumpTap

Which ad network?

InMobi

Toda!Cell®

AdMarvel

Quattro
WIRELESS

Jumptap.

admob

mojiva



millennialmedia

adfonic
GLOBAL MOBILE ADVERTISING

mcr·tm
advertising in a post-pc era

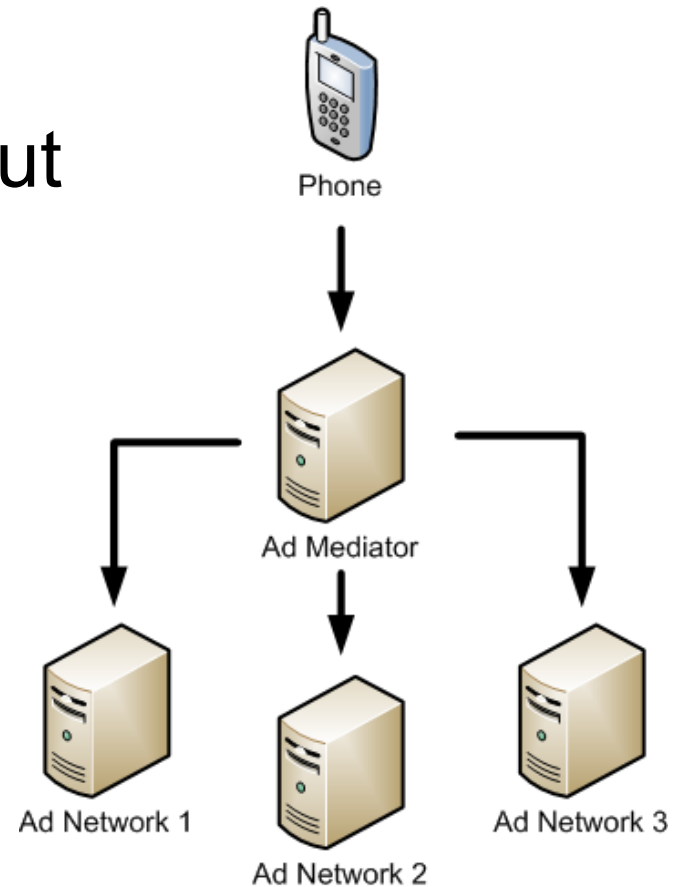
OneRiot

GREYSTRIPE

and many more...

Why choose?

- Use a Ad mediator.
- Change ad network without updating the app.
- Maximize fillrate and revenue.
- AdWhirl, Tapjoy, MobClix.



AdWhirl

- **AdWhirl** is a open source project from AdMob.
- Connectors to most common Ad networks.
- Possible to write custom connectors.
- Supports house ads.
- Requires accounts at all relevant ad networks.



CODE

AdWhirl + AdMob + InMobi

Also available at GitHub:

<https://github.com/cmelchior/Monetizing-Android-Demo-Project>

DEMO

Register app

christian@ilos.dk | [Account Settings](#) | [Help](#) | [Forum](#) | [Log Out](#)

adwhirl
by admob

Apps House Ads Reports Dev Resources

[App List](#) > [Create App](#)

Add a new App

Application Information

Name:

URL:

Platform:

Ad Serving Settings (Optional)

Background Color:

Text Color:

Refresh Rate:

Transition Animation:

Allow Location Access: ☐ OFF

© 2009-2011 AdMob, Inc.

AndroidManifest.xml

Required:

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

Optional:

```
<uses-permission android:name="android.permission.READ_PHONE_STATE" />  
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />  
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

AdMobs:

```
<activity  
    android:configChanges="keyboard|keyboardHidden|orientation"  
    android:name="com.google.ads.AdActivity" />
```

InMobi:

```
<activity  
    android:configChanges="keyboardHidden|orientation|keyboard"  
    android:name="com.inmobi.androidsdk.IMBrowserActivity" />
```

AndroidManifest.xml

AdMobs:

```
<activity  
    android:configChanges="keyboard|keyboardHidden|orientation"  
    android:name="com.google.ads.AdActivity" />
```

InMobi:

```
<activity  
    android:configChanges="keyboardHidden|orientation|keyboard"  
    android:name="com.inmobi.androidsdk.IMBrowserActivity" />
```

Main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent" >

    ...

    <FrameLayout
        android:id="@+id/ads"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentBottom="true"
        android:background="@android:color/black" >
    </FrameLayout>

</RelativeLayout>
```


Initialize ads

```
FrameLayout container = (FrameLayout) findViewById(R.id.ads);

// Setup AdWhirl
AdWhirlManager.setConfigExpireTimeout(1*60*60*1000); // Timeout - 1h
AdWhirlTargeting.setTestMode(true);
AdWhirlTargeting.setKeywords("red button fun game");

// Setup
AdWhirlLayout adsLayout = new AdWhirlLayout(this, "your adwhirl sdk key here");
int diWidth = 320;
int diHeight = 52;
float density = getResources().getDisplayMetrics().density;

adsLayout.setAdWhirlInterface(this);
adsLayout.setMaxWidth((int)(diWidth * density));
adsLayout.setMaxHeight((int)(diHeight * density));

container.addView(adsLayout);
```

Add ad networks

christian@ilos.dk | [Account Settings](#) | [Help](#) | [Forum](#) | [Log Out](#)

adwhirl

by admob

Apps

House Ads

Reports

Dev Resources

App List > [The Really Big Red Ball That Does Nothing](#) > Ad Network Settings

The Really Big Red Ball That Does Nothing

SDK Key:ef8147ec68264bce86cd371739f3f8fb

Switch App

Ad Network Settings

Backfill Priority

House Ads

App Settings

How to Activate Networks

1

Click on the key and provide the required keys to activate the explicitly supported networks

2

Use the "+ Add Custom Events" button at the top of the table to add another network to this app. Be sure to provide a name and function name.

Networks

+ Add Custom Event

Ad Network	Ad Serving	% of Traffic
House Ads	ON	0 %
AdMob	ON	50 %
Google AdSense	Not Configured	-- %
InMobi	ON	50 %
MdotM	Not Configured	-- %
Millennial Media	Not Configured	-- %
OneRiot	Not Configured	-- %
Quattro	Not Configured	-- %
ZestADZ	Not Configured	-- %

Total Allocation: 100%

Save Changes Cancel

9f3f8fb#

Set backfill

[christian@ilos.dk](#) | [Account Settings](#) | [Help](#) | [Forum](#) | [Log Out](#)

by admob

[Apps](#) | [House Ads](#) | [Reports](#) | [Dev Resources](#)

[App List](#) > [The Really Big Red Ball That Does Nothing](#) > [Backfill Priority](#)

The Really Big Red Ball That Does Nothing

SDK Key:ef8147ec68264bce86cd371739f3f8fb

Switch App

[Ad Network Settings](#)
[Backfill Priority](#)
[House Ads](#)
[App Settings](#)

Networks	
Ad Network	Priority
House Ads	3
AdMob	1
InMobi	2

[Save Changes](#) [Cancel](#)

Backfill Priority Explained

Order in which AdWhirl will request an ad from the other ad networks if the share-based 1st request fails. Active House Ads are also used for unfilled inventory.

Configure house ads

by admob

AppsHouse AdsReportsDev Resources

House Ads » Big Red Button » Edit

Big Red Button

Switch House Ad

Information

Preview

Name: Big Red Button

Goal: Website

Goal URL: http://www.bigredbutton.co

Creative

Type: Image and Text

Ad Text: Yep, it still doesn't do anything!

Image:



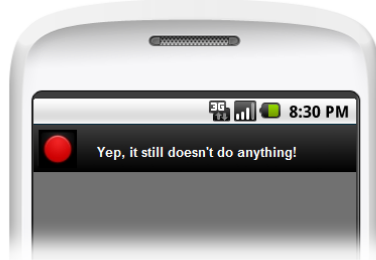
Use our default image



Select your own 38x38 image

Upload Image

Android Preview



Include in App Inventory

App Name	App Type
☐ Meter Readings	Android
☒ The Really Big Red Ball That Does Nothing	Android

Selected apps: 1 of 2

Remove Ad

Delete Ad:

Delete

 You'll be asked one more time to confirm.

Select house ads

[christian@ios.dk](#) | [Account Settings](#) | [Help](#) | [Forum](#) | [Log Out](#)

by admob

[Apps](#) | [House Ads](#) | [Reports](#) | [Dev Resources](#)

[App List](#) > [The Really Big Red Ball That Does Nothing](#) > [House Ads](#)

House Ads

Switch App ▼

[Ad Network Settings](#)
[Backfill Priority](#)
[House Ads](#)
[App Settings](#)

House Ads

House Ads are turned **OFF**

[Remove From App](#) | [+ Add House Ad](#)

☐	Ad Name	Goal	Type	Share
☐	Big Red Button	Website	Image and Text	100 %

Total Share: **100%**

[Manage House Ads](#)

Share: Desired split of house ad impressions within your house ads' share of traffic

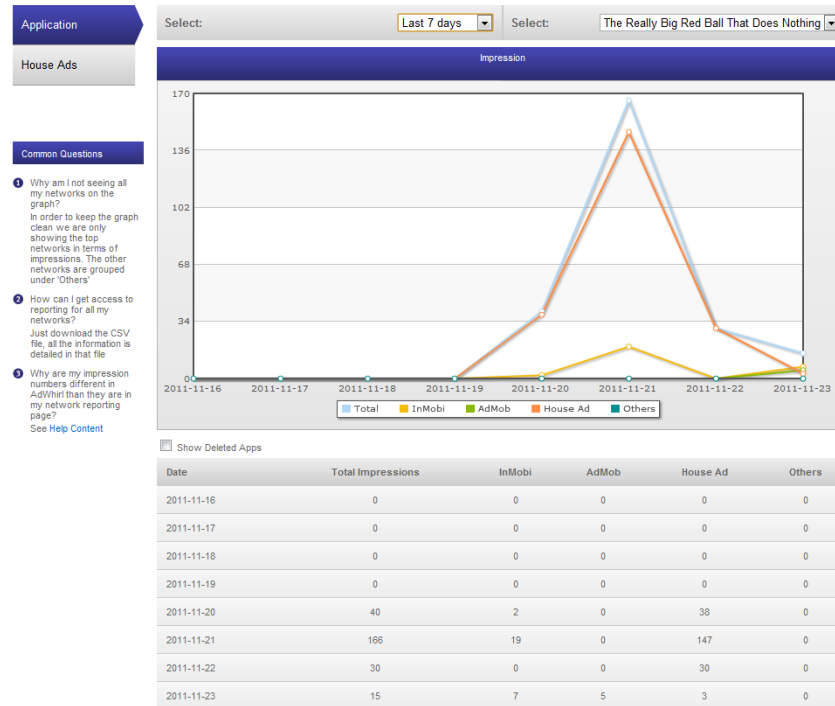
[Save Changes](#) | [Cancel](#)

© 2009-2011 AdMob, Inc.

Statistics

- AdWhirl can only track impressions.
- Go to individual networks for further data.

The Really Big Red Ball That Does Nothing



DEMO

Common problems

- Out-dated adapters.
- AdWhirl is not well documented.
- “Sum of ration weights is 0 - no ads to be shown” :
Remember to enable ad networks in AdWhirl control panel.
- Images in House ads not scaling properly:
<http://code.google.com/p/adwhirl/issues/detail?id=137>

Increased APK size

- Increases apk file size
 - Adwhirl: 55 kB
 - AdMob: 56 kB
 - InMobi: 160 kB

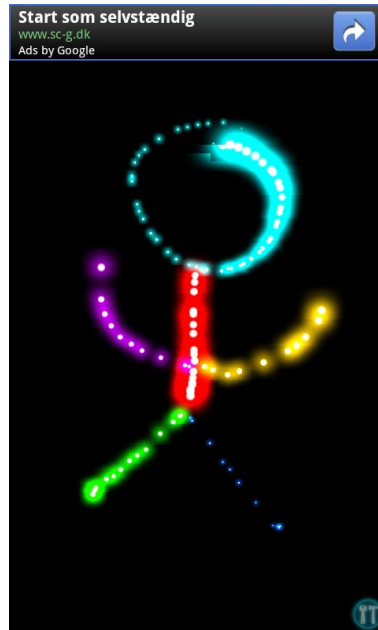
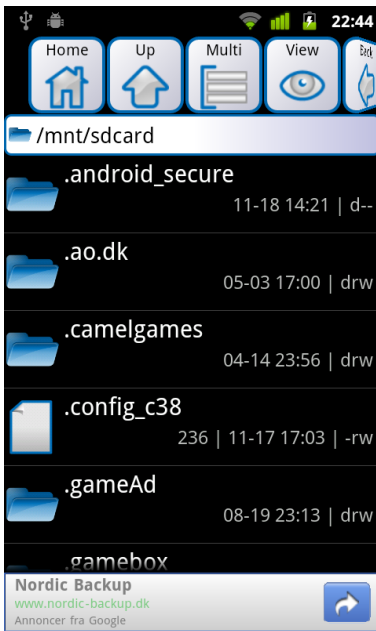
Ad size

- Standard size is ~320x50dp
- xhdpi: 1280x800 (640x100px) = 6,3%
- hdpi: 480x800 (480x75px) = 9,4%
- mdpi: 320x480 (320x50px) = 10,4%
- ldpi: 240x320 (240x38px) = 11,7%

Ad placement – Don'ts



Ad placement – Do's



Free vs. Paid - An Example

- Example: A free app with ads vs. a paid 1\$ app (0,7\$ after Googles cut).
- With an avg. eCPM of 0,5\$ you need 1400 impressions to make the same amount of money.
- 1400 impressions at a refresh rate of 30 sec:
 - ~ 11,7 hours of In-app time for a single user.
 - ~ 2,5 hours if x5 users
 - ~ 1,2 hours if x10 users.

Free vs. Paid – iPhone data

- Free apps are used 6.6 times more than paid apps.
- Apps are in average used 12 times.
- 5% of users still use the app 30 days after download. The number drops to 1% after 90 days.

Free vs. Paid

- Ads will most likely not finance your app.
 - Yes, we all heard of Angry Birds!
- Trial versions is a good idea (and also encouraged by Google).
- Experiment when you know your users.
 - Google Analytics

BREAK

IN-APP BILLING



Difference from standard paid apps?

- One APK provides different features.
- Never leaves the app.
- No 15 min. trial period.

- **Managed purchase** One-time purchases.
- **Unmanaged purchase** Able to purchase multiple times.

Managed purchases

- One time purchases, and Google keeps track.
- Upgrade from trial version.
- Remove ads.
- Access to content (eg. articles or comics).

Unmanaged purchases

- Nobody keeps track of purchases done. It is up to the developer.
- Game currency.
- Subscriptions.

Drawbacks

- Bad reviews. No free app to hide behind.
- You handle all refunds.
 - Users don't have any rights by default.
- Not possible to deploy app on other markets than Android Market.
- You can only sell digital goods.
- Google still take 30%.

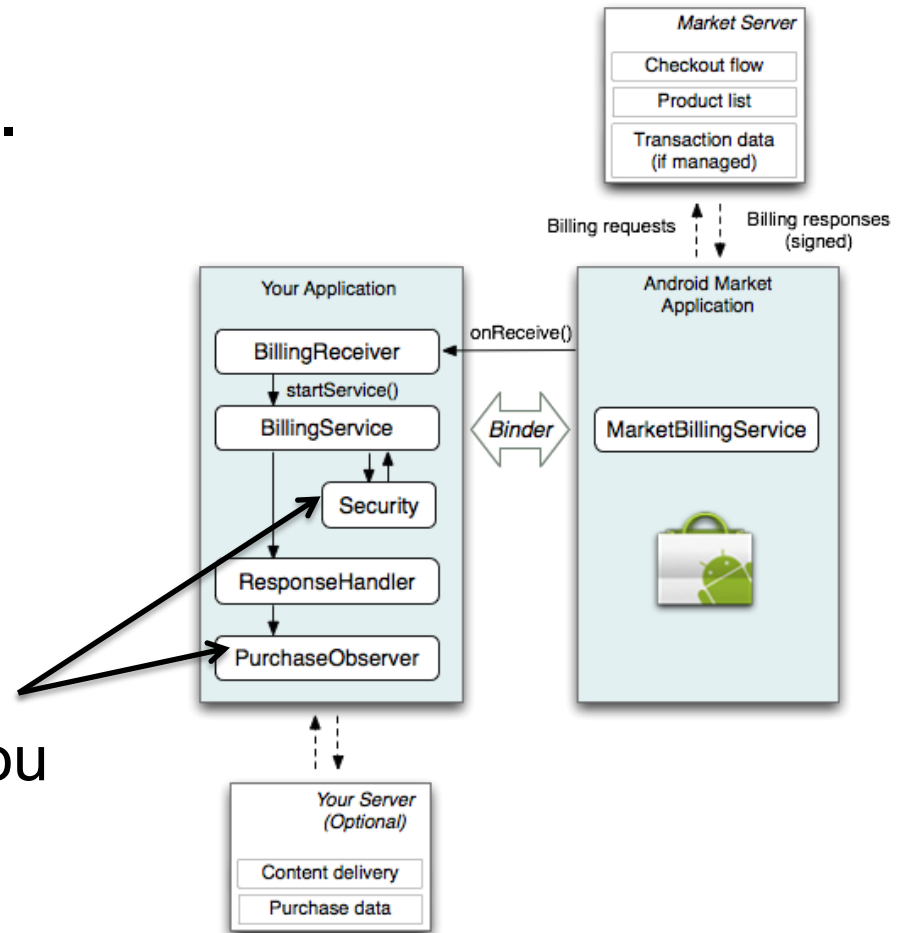
Requirements

- Google Android Market Account (25\$)
- Google Checkout Account (Free)
- Android Market installed (v. 2.3.4)
 - API level 4 (1.6) and above.

Don't use your phone account. End-to-end testing becomes impossible.

App architecture

- It looks complicated.
- It's not.



Implement this and you are done.

CODE

Also available at GitHub:

<https://github.com/cmelchior/Monetizing-Android-Demo-Project>

AndroidManifest.xml

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

<!-- Permissions -->
<uses-permission android:name="com.android.vending.BILLING" />

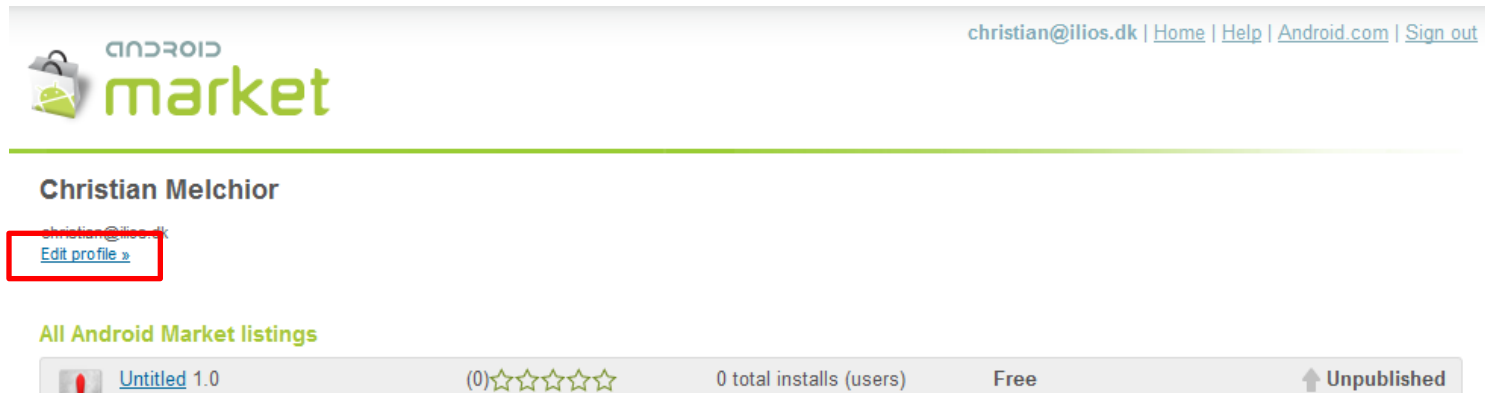
<!-- Services -->
<service android:name="dk.trifork.geeknight.billing.BillingService" />

<!-- BroadcastReceivers -->
<receiver android:name="dk.trifork.geeknight.billing.BillingReceiver">
    <intent-filter>
        <action android:name="com.android.vending.billing.IN_APP_NOTIFY" />
        <action android:name="com.android.vending.billing.RESPONSE_CODE" />
        <action android:name="com.android.vending.billing.PURCHASE_STATE_CHANGED" />
    </intent-filter>
</receiver>
```

No INTERNET permission!

Security.java

- Add Android Market public key to Security.java
- Find it under your market account profile settings..



```
String base64EncodedPublicKey = "your public key here";
```

MainActivity

- Abstract class InAppBillingActivity handles all the boring stuff.
- All we need is these four:

```
protected void onBillingSupported(boolean supported);
```

```
protected void onRequestPurchaseResponse(RequestPurchase request,  
    ResponseCode responseCode);
```

```
protected void onRestoreTransactionsResponse(RestoreTransactions request,  
    ResponseCode responseCode);
```

```
protected void onPurchaseStateChange(PurchaseState purchaseState, String itemId,  
    int quantity, long purchaseTime, String developerPayload);
```

Implementation #1

```
@Override
protected void onBillingSupported(boolean supported) {
    // Do nothing
}

@Override
protected void onRequestPurchaseResponse(RequestPurchase request,
    ResponseCode responseCode) {
    if (responseCode != ResponseCode.RESULT_OK) {
        // Request to purchase failed
        // Chance to show a message to the user
    }
}

@Override
protected void onRestoreTransactionsResponse(RestoreTransactions request,
    ResponseCode responseCode) {
    if (responseCode == ResponseCode.RESULT_OK) {
        PersistentState.setPurchasesInitialized(true);
    }
}
```

Implementation #2

```
@Override
protected void onPurchaseStateChange(PurchaseState purchaseState, String
itemId, int quantity, long purchaseTime, String developerPayload) {

    if (itemId.equals("upgrade")) {
        if (purchaseState == PurchaseState.PURCHASED) {
            PersistentState.setAppUpgraded(true);
            button.setBackgroundDrawable(getResources()
                .getDrawable(R.drawable.green_button));
        } else {
            PersistentState.setAppUpgraded(false);
            button.setBackgroundDrawable(getResources()
                .getDrawable(R.drawable.red_button));
        }
    }
}
```

Creating In-app product

- Upload draft app.
- Create product.
 - Publish product

christian@ilios.dk | Home | Help | Android.com | Sign out

Home » > In-app Products »

Edit In-app Product



Version:
Application State:  Unpublished

An in-app product will appear UNPUBLISHED until the owning application is PUBLISHED, at which point the in-app product's publishing state applies. Please use the [application editor](#) to modify the application's publishing state.

Questions about in-app products? Click [here](#) to learn more!

In-app Product ID
upgrade

Purchase Type
☒ Managed per user account ☐ Unmanaged

Publishing State
 Published

Language
[add language](#) | *English (en) |
The **default language** is inherited from the owning application. Click [here](#) to change default language!

Title
Upgrade
7 characters (best viewed under 25, max 55)

Description
Green is the new red.
21 characters (80 max)

Price
DKK 6.00

Automatically populate the fields with a one-time conversion to local currencies from the default price with the current exchange rate

The targeted countries are inherited from the owning application. Click [here](#) to change targeted countries!

Australia	AUD	1.10	Netherlands	EUR	0.81
Austria	EUR	0.81	New Zealand	NZD	1.45
Belgium	EUR	0.81	Norway	NOK	6.31
Canada	CAD	1.13	Portugal	EUR	0.81

Purchase/restore products

■ Functions available

```
restorePurchases();  
requestPurchase(String productId);  
requestPurchase (String productId, String developerPayload);
```

■ Easy to implement

```
@Override  
public boolean onOptionsItemSelected(MenuItem item) {  
  
    switch(item.getItemId()) {  
        case OPTION_RESTORE: restorePurchases();  
        case OPTION_PURCHASE: requestPurchase("upgrade");  
        case OPTION_TEST_PURCHASED: requestPurchase("android.test.purchased"); break;  
        case OPTION_TEST_CANCELED: requestPurchase("android.test.canceled"); break;  
        case OPTION_TEST_REFUNDED: requestPurchase("android.test.refunded"); break;  
        case OPTION_TEST_ITEM_UNAVAILABLE: requestPurchase("android.test.item_unavailable"); break;  
        default:  
            return super.onOptionsItemSelected(item);  
    }  
  
    return true;  
}
```

Testing

- Start with static responses
 - "android.test.purchased"
 - "android.test.canceled"
 - "android.test.refunded"
 - "android.test.item_unavailable"
- Real purchases
 - Test accounts

Remember to refund your test purchases.

DEMO

Saving purchases

- SharedPreferences
- Purchase database
- Remotely

- Loosing unmaged purchases will piss your users off. Backup the data.
 - Google Cloud Backup API from API8.

Security

- Preferences and databases are not safe.
 - SharedPreferences in clear text.
 - Database is not bound to the app.
 - Both can be copied independently of app.
- Decompiling is easy.
 - APK_OneClick

Raising the bar

- Disable logging output.
- Use ProGuard.
- Encryption of clear text values.
 - SimpleCrypto
- Not "real security", but a hacker will need to work harder.

Logging

- Use a special logger class. Easy to disable/enable logging.

```
public class Logger {  
  
    public static void e(String tag,String msg){  
        if( isLoggingEnabled()){  
            Log.e(tag, (msg != null) ? msg : "null");  
        }  
    }  
  
    private static boolean isLoggingEnabled() {  
        return true;  
    }  
}
```

Enabling Proguard

- Create build.xml

```
C:\PathToProject> android update project --name Monetizing-Android-Demo-Project --target 8 --path .
```

- Update ant.properties

```
key.store=keystore/demo.keystore  
key.alias=geeknight\_demo  
key.store.password=Trifork1234  
key.alias.password=Trifork1234
```

- Update local.properties

```
proguard.config=proguard.cfg
```

```
C:\PathToProject>ant release
```

- Update proguard.cfg

```
-keep class com.android.vending.billing.**
```

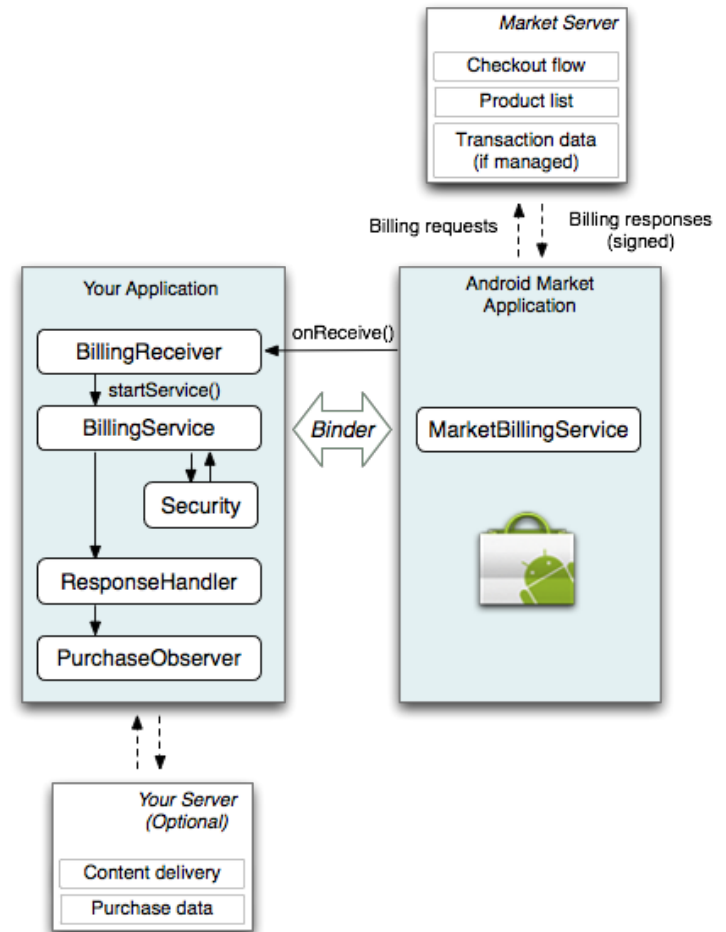
Encrypting clear text

- Introduce a PersistentState wrapper.
- Encrypt all keys with a device specific key using SimpleCrypto.

```
public static void setAppUpgraded(boolean isUpgraded) {  
    SharedPreferences.Editor editor = prefs.edit();  
    if (isUpgraded) {  
        editor.putBoolean(encryptKey("upgraded"), isUpgraded);  
    } else {  
        editor.remove(encryptKey("upgraded"));  
    }  
  
    editor.commit();  
}
```

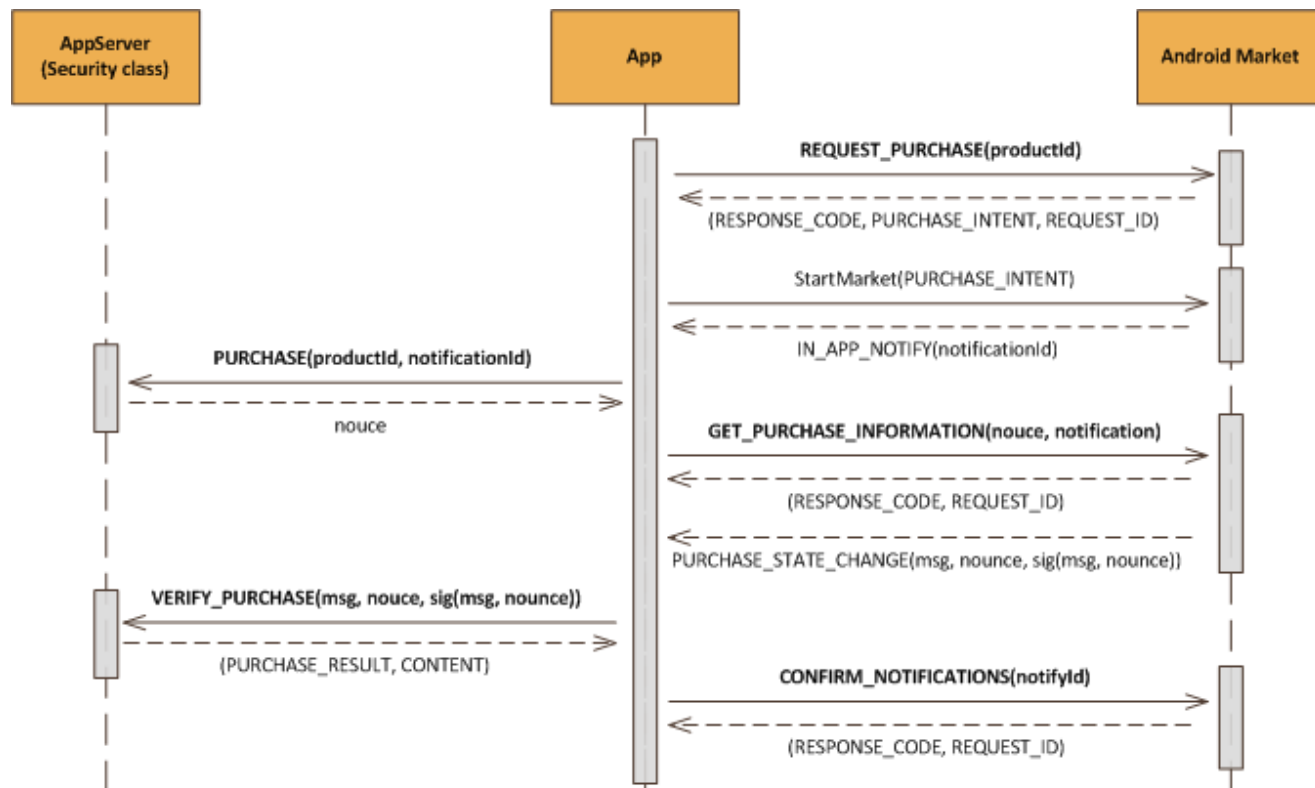
Real security

- Don't trust the client.
- Server needed.
- Remember this?
- Not so easy anymore

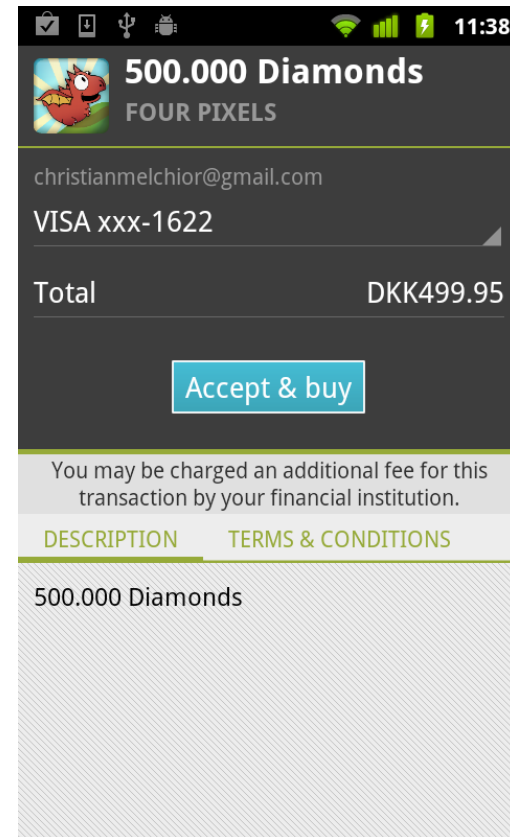


Validation is server side

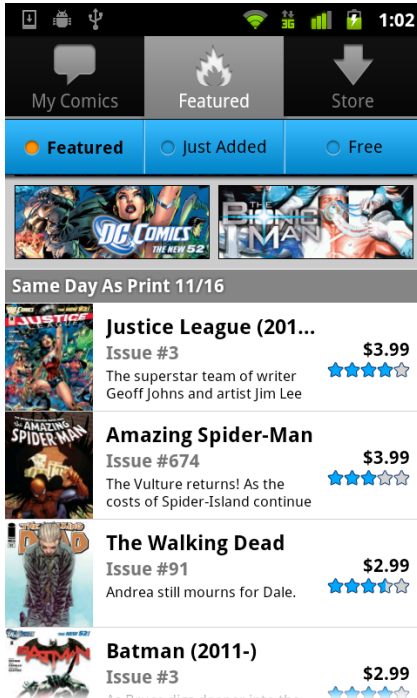
- Security/validation must move to server.
- Verify purchase state on startup.



In-app Billing – Don'ts



In-app Billing – Do's



?

?

?

?

?

QUESTIONS

?

?

?

?

?

?