Android
～開発に必要な事～

佐野　尚之

本ドキュメントのライセンスについて

変更履歴

■第１版

2011/5/29 (2010/4/23頃から作成開始)

オープンソースの「LibreOffice(リブレオフィス) 3.3.1 (※)」を使用して作成

※「OpenOffice.org」から派生したオープンソースのオフィス統合環境

目次

OS、開発環境、Androidなどの対象バージョン

■ＯＳ

Windows 7 Home Premium(32bit版)

※AndroidはWindows XP, Windows Vista, Linux, Mac OS Xでも動作します。

■開発環境

Eclipse 3.6.1 + Pleiades(プレアデス) + Android Development Toolkit(ADT)

■Androidの対象バージョン

Android SDK 2.3

■Java SE Development Kit(JDK)の対象バージョン

JDK 6 update23以降

基礎知識 (1/11)

■効率的な開発のために

1、Activityのライフサイクルで注意すべき点

(1) センサーやGPS、カメラのようなハードウェアの利用を、ライフサイクル上のタイミングで開始／終了するかを事前に検討する必要がある。

・ハードウェアの占有は、きちんと終了をしないと電力の消費の増大や他のアプリとの衝突によるエラーの原因になる。バックグラウンド処理(Service)との接続(bind)についても同様です。

(2) ライフサイクルの遷移は、端末の設定に大きな変更があった場合にも起こる。

・キーボードの出し入れのタイミングで画面が切り替わる。
Activityは一気にonDestroyまで遷移。その後onCreateから再起動します。
画面の表示内容や入力途中の内容、クラスメンバの値も初期化されます。

・再起動後の画面に情報を引き継ぎたい場合
onSaveInstanceStateメソッドをオーバーライドして、引数のBunbleオブジェクトに値を設定。
設定した値は、onCreateやonRestoreInstanceStateメソッドの引数に渡されるので、値を復帰するには、このメソッドで処理します。

<参考情報>
第3回 Androidによる開発
ライフサイクル
http://thinkit.co.jp/article/916/1
onSaveInstanceStateでインスタンスを保存する
http://techbooster.jpn.org/andriod/application/3706/
onSaveInstanceStateとonRestoreInstanceStateイベント
http://libro99.appspot.com/index3?id=190001&page=2

基礎知識 (2/11)

2、リソース修飾子

Androidのアプリ開発は「コードとリソースの分離」が徹底されています。コードはsrcフォルダ配下、リソースはresフォルダ配下に配置する規則になっています。
リソースには、xmlで定義する画面の構造や文字列、アニメーション、画像、音声などがある。それぞれ配置するフォルダが指定されています。それぞれの種類のフォルダは「-」(ハイフン)に続けて特定の記号を付けることができます。特定の記号を「リソース修飾子」と呼びます。
開発を効率的に進めるうえで、どのようなリソース修飾子があるのかを把握しておく必要があります。

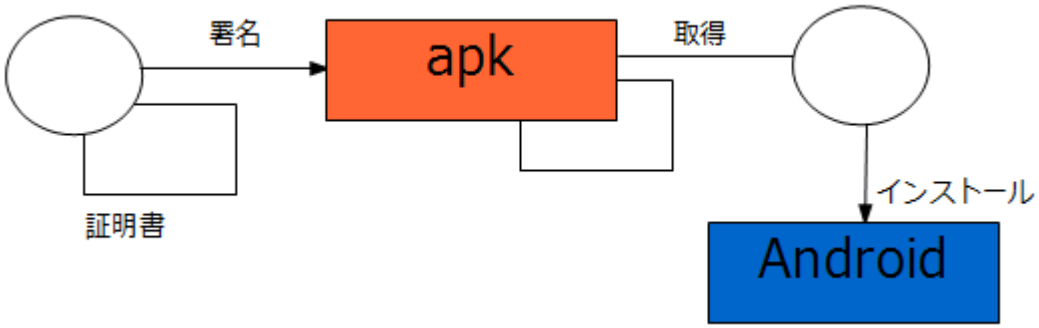
<参考情報>
7. アプリケーションリソース
https://sites.google.com/a/techdoctranslator.com/jp/android/guide/resources/providing-resources
ワイヤレスジャパンの Android アプリ開発講座メモ
http://takanory.net/takalog/1238
リソースの多言語対応
http://techbooster.jpn.org/andriod/resource/583/

基礎知識 (3/11)

3、開発用証明書の共有

(1) 証明書について

Androidアプリの配布パッケージは「.apk(Android PacKage)」の拡張子を持つ単一ファイルです。apkファイルは「jar(Java Archive)」と同じく、zip形式で圧縮されており、展開後の構造も共通です。アプリケーションのapkファイルをAndroidシステムにインストールするには、apkファイルが証明書を使って署名されている必要があります。インストール先がエミュレータでも実機端末でも同様の署名のチェックが行われます。



基礎知識 (4/11)

Androidシステムは、インストールされているアプリケーションの署名情報を保存し、次に同じアプリケーション(同じパッケージ名)のapkファイルをインストールする際には、署名の一致を確認します。
署名が一致していれば、証明書を使って署名されたとして上書きを許可します。(図1)
証明書が一致しない場合は、同じ証明書を持たない開発者が作成したapkファイルとして、上書きインストールを拒絶します。(図2)

図1

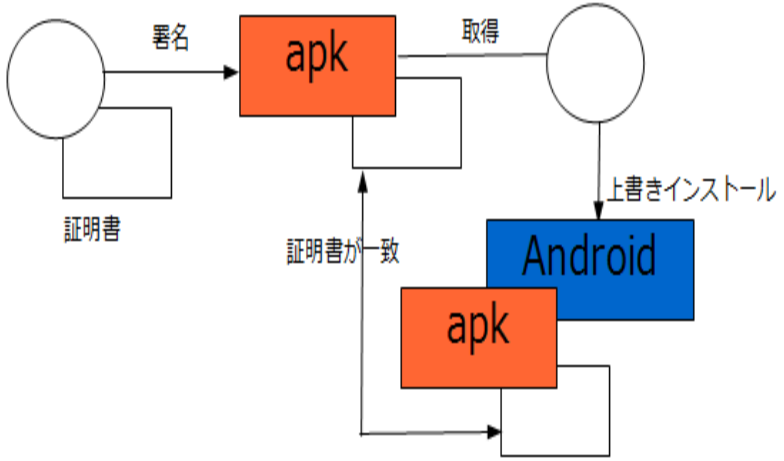
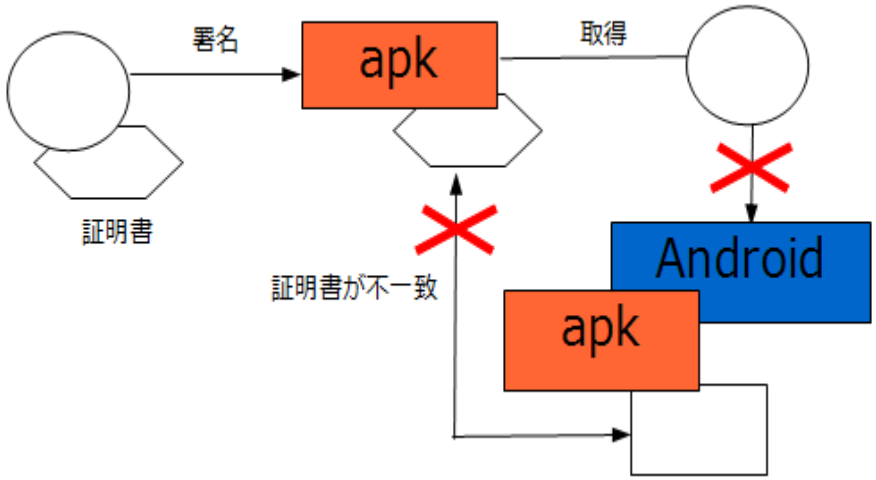


図2



基礎知識 (5/11)

署名は「JDK(Java Development Kit)」に含まれている「jarsignerツール」で行います。
EclipseからもAndroidプラグインを起動して、証明書の作成と署名を実行できます。
開発途中のアプリケーションをテストする場合、Androidプラグインは一度生成したapkファイルに仮の証明書を使って署名しています。この仮の証明書をここでは「開発用証明書」と呼びます。
開発用証明書はパスワードの設定がなく、有効期限が1年間になっています。
開発者のホームディレクトリの.androidディレクトリに「.debug.keystore」という名前でファイルが作成されます。

(2) 開発用証明書の共有について

それぞれの開発者の開発証明書が異なる場合、インストールの際に非常に手間がかかる。１つの実機端末を共有する場合、他の開発者のインストールしたアプリがあった場合、署名が一致せずに上書きインストールができません。受け渡しのたびに、前にインストールされているアプリを手動でアンインストールすることになります。

対策として、開発者のうち一人の開発用証明書を共有します。EclipseのAndroidの設定画面から、デフォルト・デバッグ・キーストアで、共通の証明書を指定します。設定するとEclipseは、apkファイルを署名する際に指定した証明書ファイルを使用します。

<参考情報>
Androidアプリ開発の極意
http://www.slideshare.net/daisaku/android-4317160
Android複数人開発時の注意
http://www.bpsinc.jp/blog/archives/1296
CA証明書とandroidの危うい関係
http://d.hatena.ne.jp/Kazzz/20110320/p1
Android のセキュリティーを理解する
http://www.ibm.com/developerworks/jp/xml/library/x-androidsecurity/index.html
開発用署名 debug.keystore
http://sites.google.com/site/matsudroid/tips/kaihatsuyoushomeidebugkeystore

基礎知識 (6/11)

4、難読化ツール「ProGuard」

Android SDK 2.3(ADT 8.0.0)から難読化ツール「ProGuard」が標準で搭載されています。設定だけでEclipseからビルドする際に自動でProGuardを適用してくれます。また、難読化以外に圧縮や最適化の機能もあります。

■なぜ難読化が必要なのか？

JavaやC#(.NET)などの言語は、ソースコードから機械語へ変換する際に中間言語というOSやハードウェアに依存しないバイトコードを生成しています。中間言語を解析することでソースコードをある程度復元して、処理内容を知ることができます。ちなみにバイトコードを解析して元のソースコードを得ることを「デコンパイル」。デコンパイルした内容から処理内容を解析することを「リバース・エンジニアリング」と言います。
Androidの場合はdex(Dalvik EXecutable)というバイトコードで動作します。Android実行環境 Dalvik VMで動くようにJavaクラスファイルを変換して生成するので、Javaと同様にデコンパイルが簡単。
復元したソースコードから処理の内容を解析を防ぐためにライセンスの確認などの不正利用を防止する仕組みとともに難読化することで、無料で使用されたり改変や再配布されることを防げます。
個人的には、ソースコードの処理内容が見られた時にセキュリティホールを攻撃される可能性もあるので内容を見られても攻撃するところがどこもないと思われるようなソースコードにしておくことも重要であるのではと思っています。

基礎知識 (7/11)

■**ProGuardの動作について**

圧縮、最適化、難読化、実行前検証のそれぞれのステップで以下に説明する処理が行われます。
難読化する必要がなくても、モバイルアプリケーションは限られたメモリのリソースを有効に使うように意識して作成する必要がありますので、圧縮や最適化も行ってくれるProGuardはAndroidアプリの開発に非常に役に立つツールであると思います。

(1) 圧縮

クラスとそのクラスメンバに対して、使用されているかをチェック。使用されていないコードを削除。

(2) 最適化

エントリーポイントでないクラスとメソッドはprivateまたはfinalやstaticに置き換え。使用されていないパラメータは削除され、メソッドはインライン化(リファクタリングなどにおいてパフォーマンスを上げるため、メソッドの起動をメソッド本体と置き換えること)される。

(3) 難読化

エントリーポイントでないクラスやクラスメンバの名前をより可読性の悪い名前に変更します。
(-keepオプションで指定されたエントリーポイント以外で)

(4) 実行前検証

圧縮、最適化、難読化処理に対して、正しくJavaVMで動くかどうかを検証します。

基礎知識 (8/11)

■参考情報

Android + Google Guice + ProGuard
http://www.asp-edita.jp/doda/one/doda4709_2359.html
Android SDKに統合された難読化ツール「ProGuard」を試す
http://blog.bari-ikutsu.com/entry/20101211_4533.html
Google Android開発環境の構築 SDK r10版
http://www.neko.ne.jp/~freewing/android/android_sdk_r8/
Android　Eclipse から ProGuard を使ってみた。
http://y-anz-m.blogspot.com/2010/12/androideclipse-proguard.html
最適化ツール最新版"ProGuard 4.0"登場 - Java 6への検証機能導入
http://journal.mycom.co.jp/news/2007/10/01/009/index.html
ソフトウェア技術ドキュメントを勝手に翻訳 -ProGuard-
http://www.techdoctranslator.com/android/developing/tools/proguard
Androidとセキュリティ：プログラム難読化ツール - ProGuard -
http://d.hatena.ne.jp/bs-android/20101129/1291008438
Androidとセキュリティ：Android 2.3(Gingerbread) SDKに標準搭載されたProGuardを試す
http://d.hatena.ne.jp/bs-android/20101207/1291702021
AndroidアプリにProGuardを使う
http://skyarts.com/blog/jp/skyarts/?p=2007
新しいAndroid SDKにはProGuardが標準装備されたそうです。
http://turtle2005.blog.so-net.ne.jp/2010-12-08-1
AndroidアプリをProGuardで難読化する
http://moait.gate.ne.jp/archives/741
ProGuardについて
http://rokuta96.blog137.fc2.com/blog-entry-102.html
androidでProGuard(ant build)
http://mokkouyou.blog114.fc2.com/blog-entry-76.html
Google Adsense を使用しているアプリに対するProguardの適用
http://android.asai24.com/archives/cat_10045498.html
[技術メモ]Android proguardの利用
http://d.hatena.ne.jp/Mr_MONK/20110420/p1
ProGuardを使っての難読化方法
http://d.hatena.ne.jp/hyoromo/20101120/1290216449

基礎知識 (9/11)

5、ライブラリプロジェクトの作成について

■なぜライブラリプロジェクトを作るのか？

複数のアプリケーションで同じ機能を組み込みたい場合が非常に多いので、他のプロジェクトで使用可能する方法としてライブラリプロジェクトとして管理しておくと便利であるからです。**ライブラリにバグがあった場合、ライブラリだけ修正すればいいので複数のアプリケーションで修正する手間が省けます。**
デメリットもあります。ライブラリは共通機能であるため、ライブラリを修正するとライブラリを使用している全てのプロジェクトを再テストする必要があります。

■リソースIDの競合と優先順位

ライブラリプロジェクトのビルドは、メインプロジェクトのビルドと同時に行われます。その際にライブラリプロジェクトとメインプロジェクトで同じリソースIDが定義されていてもビルド時にはエラーになりません。
リソースIDが競合する場合は、メインプロジェクトで使用しているリソースIDを優先します。
複数のライブラリで競合が起こった場合は、ライブラリ一覧で上位に指定したライブラリのリソースが優先します。優先順位を理解できずにリソースIDを競合させてしまうと、想定していない動作をする場合があるようなので注意してください。**リソースIDの競合を避けるためには、各リソースIDに接頭辞を付与するなどのリソース命名規約を決めておく必要があります。**

基礎知識 (10/11)

■ライブラリプロジェクト作成時の注意点

(1) assetsのファイルへのアクセス

ライブラリプロジェクトのassetsにファイルを配置した場合、メインプロジェクトのプログラムからはアクセスできません。**assetsにファイルを置く必要がある場合、メインプロジェクトのassetsにファイルを置いてください。**

(2) ライブラリプロジェクトのAPIレベル

ライブラリプロジェクトのAPIレベルは、メインプロジェクトに指定したAPIレベル以下にする必要があります。ライブラリプロジェクトの方がAPIレベルが高い場合はビルドに失敗します。
ライブラリプロジェクトとメインプロジェクトは可能であれば同じAPIレベルにしておくとよいと思います。Google APIs Add-Onのように外部ライブラリが含まれたビルドターゲットを指定した場合もメインプロジェクトで同じビルドターゲットを指定する必要があります。

(3) 外部ライブラリ(jar)の使用

外部ライブラリを使用してライブラリを作成した場合、**メインプロジェクトでもライブラリが参照しているjarファイルへの参照を追加する必要があります。**

(4) ライブラリプロジェクトのjar配布

リソースが含まれているライブラリプロジェクトはjarとして配布できないようです。

(5) AndroidManifest.xmlの記述

メインプロジェクトのAndroidManifest.xmlにライブラリプロジェクトから使用するActivityなどのシステムコンポーネントの宣言を記述する必要があります。

(6) ライブラリプロジェクトを修正した時

メインプロジェクトで再読込する必要があります。

基礎知識 (11/11)

■参考情報

Androidでライブラリプロジェクトを作成する
http://d.hatena.ne.jp/waochi/20101108/1289181917
外部ライブラリのすゝめ
http://d.hatena.ne.jp/tomorrowkey/20100908
外部ライブラリのすゝめ2
http://d.hatena.ne.jp/tomorrowkey/20100910/1284133106
Android ライブラリプロジェクトをテストする
http://www.pshared.net/diary/20110415.html
[Android]ライブラリプロジェクトを使う
http://genz0.blogspot.com/2011/02/android.html
[Android]ライブラリプロジェクトの作成と使用
http://blog.rabisoft.com/archives/256
Android　Quick Action の Android ライブラリプロジェクトを作ってみた
http://y-anz-m.blogspot.com/2010/11/androidquick-action.html
AndroidでActivityを含む別プロジェクトをライブラリとして参照する方法
http://tmpla.info/2011/03/13/androidでactivityを含む別プロジェクトをライブラリとして/
AndroidでJARライブラリを作成
http://n2works.net/column/pickup/id/77
プロジェクトに外部ライブラリをインポートする
http://techbooster.jpn.org/andriod/environment/4897/
Androidの共通ライブラリの開発
http://wavetalker.blog134.fc2.com/blog-entry-35.html

デバッグ (1/4)

1、ログ出力

ログ出力はLogクラスの静的メソッドで行うことができます。出力する内容の他に「タグ」と「ログレベル」の情報を設定します。タグはログに付ける識別子で、どのクラスやメソッドから出力されたものなのかを識別する時に使います。ログレベルはログの重要度を表します。

ログレベル	メソッド	説明
ERROR	Log.e	エラー
WARN	Log.w	警告
INFO	Log.i	情報
DEBUG	Log.d	デバッグ
VERBOSE	Log.v	詳細

Logクラスのprintlnメソッドで出力した内容は、Android端末やエミュレータ画面では確認できません。DDMS (Dalvik Debug Monitoring Service)のLogCatから行います。DDMSは、Android SDKに含まれています。
EclipseのAndroidプラグインからパースペクティブとして表示も可能です。
ログは、さまざまなアプリケーションから出力されるため、そのままだと大量のログが表示されます。
本当に必要なログを見逃さないためにDDMSでフィルタ設定を行って参照するのが一般的なようです。

デバッグ (2/4)

■参考情報

ログの参照
http://www.javadrive.jp/android/debug/index2.html
[Android] ログレベル設定方法
http://babukuma.com/2010/09/android_01.html
ログで日本語文字を見る
http://knowledge.sorich.jp/?Java/Android/応用編/ログで日本語文字を見る
ログ(Log)を出力するには
http://www.adakoda.com/android/000064.html
Eclipse を使用した Android のログ出力
http://ohwhsmm7.blog28.fc2.com/blog-entry-42.html
Logの出力
http://ichitcltk.hustle.ne.jp/gudon/modules/pico_rd/index.php?content_id=46
デバッグのためのログ出力
http://androidmemo.blog66.fc2.com/blog-entry-16.html
Android SDK/NDKのログ出力機能を使う
http://www.usefullcode.net/2010/12/android_sdkndk.html
SDカードにログを出力する方法
http://d.hatena.ne.jp/bs-android/20091001/1254407223
androidログ用クラスの作成
http://study-angelscript.blog.ocn.ne.jp/blog/2010/06/android_51b5.html
android ログ出力(Log)
http://baldwin.ivory.ne.jp/wp/2010/08/03/273/
ログ出力　小技編
http://blackcoffee-milk.blogspot.com/2011/01/blog-post.html
AndroidのNative層でのログ出力
http://www.swingingblue.net/mt/archives/002525.html
ログの出力
http://www.javadrive.jp/android/log/index2.html

デバッグ (3/4)

2、デバッガ

エミュレータや実機端末は開発用のPCとは分離されていますが、ADB(Android Debug Bridge)によって接続されており、デバッガによるデバッグ作業が可能です。デバッグ実行を使うと、エミュレータや実機端末上で実行中のアプリケーションの処理を特定の場所で一時停止したり、変数の状態を観察できるので開発の効率化につながります。デバッグ実行をしたい場合は、AndroidManifest.xmlにdebuggableという属性値を追加する必要があります。また、アプリケーションの起動時にデバッグ実行を指定する必要があります。
デバッグ実行中に一時停止したい場合、停止したい場所にブレークポイントを設定します。
変数の状態を観察したい場合、目的の変数にカーソルを合わせて右クリックでメニューを表示して[Watch]を選択すると、観察対象の変数が追加できます。観察対象の変数はExpressionsタブに追加されますので、この状態でプログラムののデバッグ実行を継続して、ブレークポイントで停止／手動で更新するたびに指定した変数の状態を確認できます。

■参考情報

デバッガ(DDMS)の利用
http://www.javadrive.jp/android/debug/
実機でデバッグするには
http://www.adakoda.com/android/000242.html
開発したアプリをエミュレーターやデバッガ上でテストする
http://ascii.jp/elem/000/000/535/535273/
Google Android用携帯アプリ作成のための基礎知識
http://www.atmarkit.co.jp/fjava/column/koyama/koyama09_4.html
android情報まとめ@ウィキ -ドキュメント/SDKとプラグインのインストール-
http://www29.atwiki.jp/android/pages/14.html
デバッグの意味を考える -Androidアプリを作る 8-
http://www.freeml.com/bl/360328/648764/
デバッグの仕方
http://wiki.android-fun.jp/?デバッグの仕方
エミュレータ/デバッガ関連
http://android.roof-balcony.com/category/debug/
Android Vista 64bitでスタンドアローン版のDDMSを使用する方法
http://www.taosoftware.co.jp/blog/2009/04/android_dalvik_debug_monitor_s_3.html
Windows7 64bit環境でddms.batを使うためには
http://calcnote.net/?p=524
Androidのデバッグダンプからソース上の問題発生箇所を調べる方法
http://blog.kmckk.com/archives/3505138.html

デバッグ (4/4)

3、StrictMode

■StrictModeについて

StrictModeはAndroid2.3から追加されました。StrictModeを設定すると、時間のかかる本来別のスレッドで実行するべき操作がメインスレッドで実行されるのを検知する。ダイアログの表示やログに出力したりして、パフォーマンスの低下とANR(Application Not Responding)表示の可能性を警告することができます。

■参考情報

Android　Android 2.3 - StrictMode -
http://y-anz-m.blogspot.com/2010/12/androidandroid-23-strictmode.html
StrictModeでパフォーマンスをチューニングする
http://d.hatena.ne.jp/bs-android/20101229/1293582940
[翻訳]StrictModeについて
http://d.hatena.ne.jp/Superdry/20101213/1292267620
ジンジャーブレッドは、AndroidのAPI StrictModeを紹介
http://geekfiles.altervista.org/ja/android-gingerbread-introduce-le-api-strictmode/
Y.A.M の 雑記帳: Android　Android 2.3 - StrictMode -
http://buzzurl.jp/entry/Y.A.M の 雑記帳%3A Android　Android 2.3 - StrictMode -/3019555

テスト (1/20)

1、ユニットテスト

■Androidのユニットテスト

クラスやメソッドに条件を入力後の出力を確認して、正しい動作かどうかを確認するもので「単体テスト」とも呼ばれています。Javaではユニットテストを自動化するツールとして「JUnit」が有名ですがAndroidでは、JUnit3での自動テストに対応しています。
システムコンポーネントのうち、「Activity」「Service 」「ContentProvider」のテストが可能です。

■JUnitによるユニットテストについて

・テスト対象のクラスとテストをする側のクラス(テストケース)を別々に作成します。
・テストメソッドは接頭辞「test」から始める規則になっています。
・取得結果の検証メソッドとして、以下のメソッドが用意されています。
・テストメソッドの実行で、全てのテストが成功すればEclipseのJUnitウインドウは緑色のバーになり、1つでも失敗すると赤色のバーになります。
・テストの前処理「setUp()」とテストの後処理「tearDown()」用のメソッドもあります。

検証メソッド	概要
assertEquals	期待値と実績値が等しい場合は正常終了。等しくない場合は失敗終了する。
assertFalse	実績値が false の場合は正常終了。true の場合は失敗終了する。
assertNotNull	実績値が null でない場合は正常終了。null の場合は失敗終了する。
assertNotSame	期待値と実績値が同じ参照をしていない場合は正常終了。同じ参照をしている場合は失敗終了する。
assertNull	実績値が null の場合は正常終了。null でない場合は失敗終了する。
assertSame	期待値と実績値が同じ参照をしている場合は正常終了。同じ参照をしていない場合は失敗終了する。
assertTrue	実績値が true の場合は正常終了。false の場合は失敗終了する。
fail	このメソッドが実行されると失敗終了する。

テスト (2/20)

■参考情報

Eclipse + ADT + TestNG でInternalError
http://blogs.yahoo.co.jp/udumge/51030204.html
アノテーションを使用した簡単テストフレームワーク「TestNG」
http://blog.mikuriya.biz/archives/tag/testng
アノテーション(7) - 採用事例 TestNG(4)
http://journal.mycom.co.jp/column/java/032/index.html
第7回 単体テストを楽にするプラグイン
http://itpro.nikkeibp.co.jp/article/COLUMN/20071029/285773/
JUnitチュートリアル
http://www.次世代創造機構.jp/android/androidLecture/JUnit/JUnit.html
AndroidアプリケーションをJUnitでテストする
http://itinfo.main.jp/tan/?p=35
androidおけるJUnitの使い方
http://blogs.yahoo.co.jp/bananamanbow2006/17581730.html
Androidチュートリアルインデックス
http://www.次世代創造機構.jp/android/androidLecture/
TestNGとJUnit4における、テストメソッドへのパラメータの考え方の違い
http://d.hatena.ne.jp/jyukutyo/20080808/1218159314
Androidのテスト環境について
https://groups.google.com/group/android-group-japan/browse_thread/thread/09e605ecaaaba9f6/52e659a996ab2545?hl=ja
JUnit/テストアプリケーションの作成
http://wikiwiki.jp/android/?JUnit%2F%A5%C6%A5%B9%A5%C8%A5%A2%A5%D7%A5%EA%A5%B1%A1%BC%A5%B7%A5%E7%A5%F3%A4%CE%BA%EE%C0%AE
Hello, Testing
http://developer.android.com/intl/ja/resources/tutorials/testing/helloandroid_test.html

テスト (3/20)

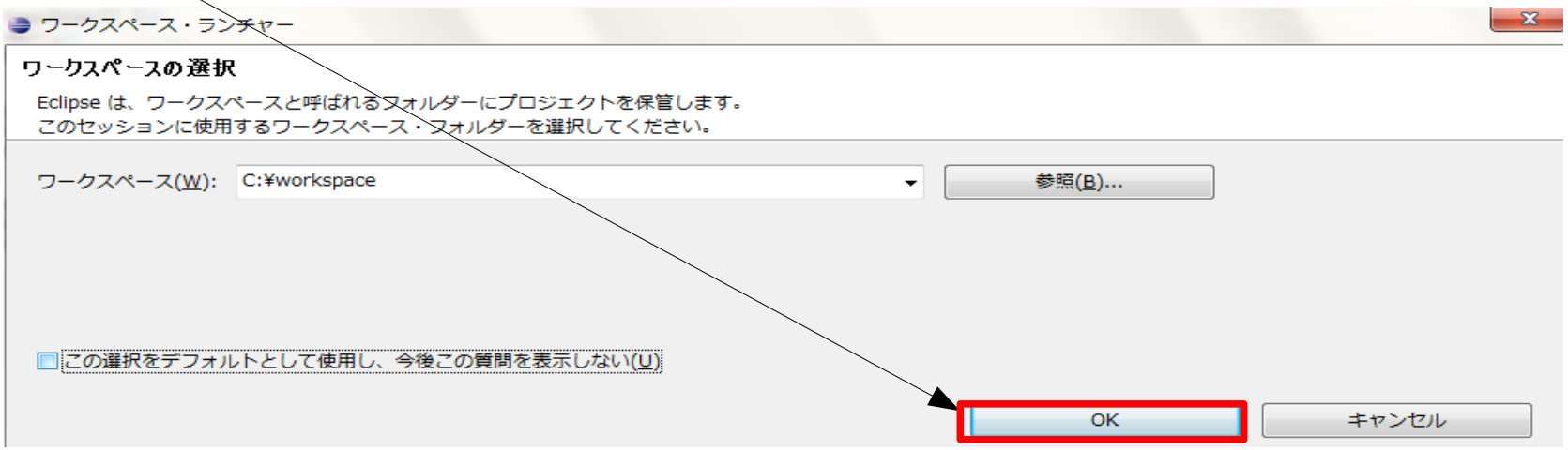
AndroidのアプリでJUuitを使用する場合は、通常のプロジェクトとは別にテスト専用の「テストプロジェクト」を作成する必要があります。

2、ユニットテスト

■テスト対象クラスとテストクラスの作成例・・・JUnit4の場合 (AndroidはJUnit3)

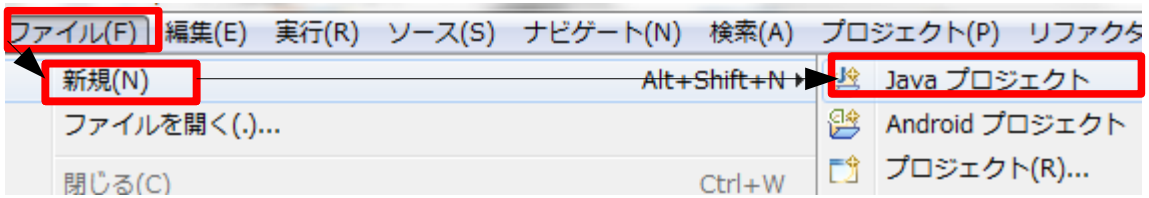
(1) デスクトップにあるEcipseのショートカットをダブルクリックします。

(2)「OK」ボタンをクリックします。

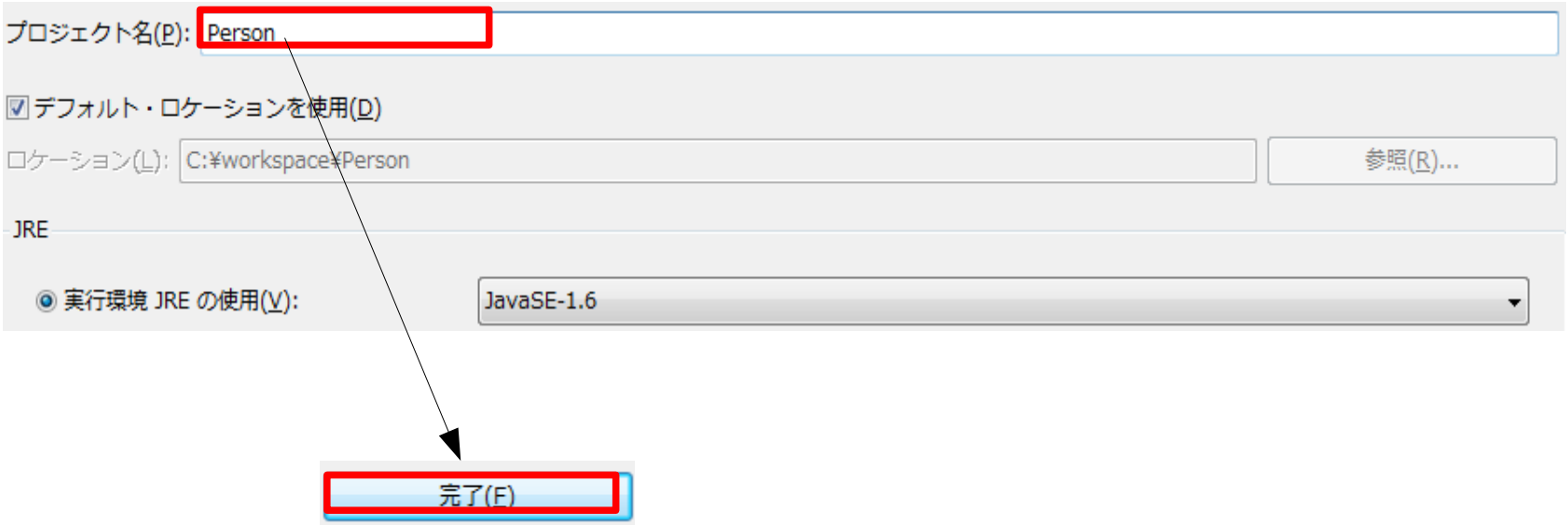


テスト (4/20)

(3)「ファイル」－「新規」ー「Java プロジェクト」を選択します。

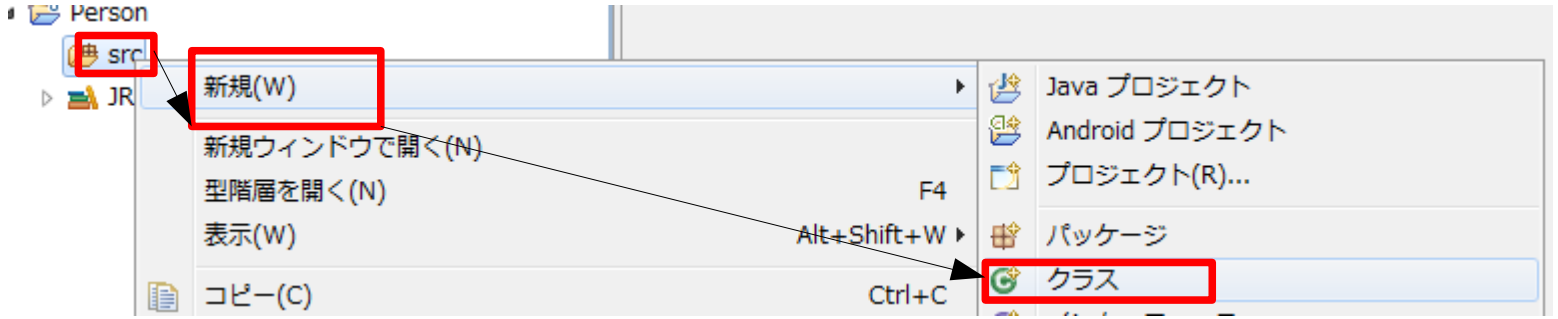


(4) プロジェクト名に「Person」と入力後に「完了」ボタンをクリックします。

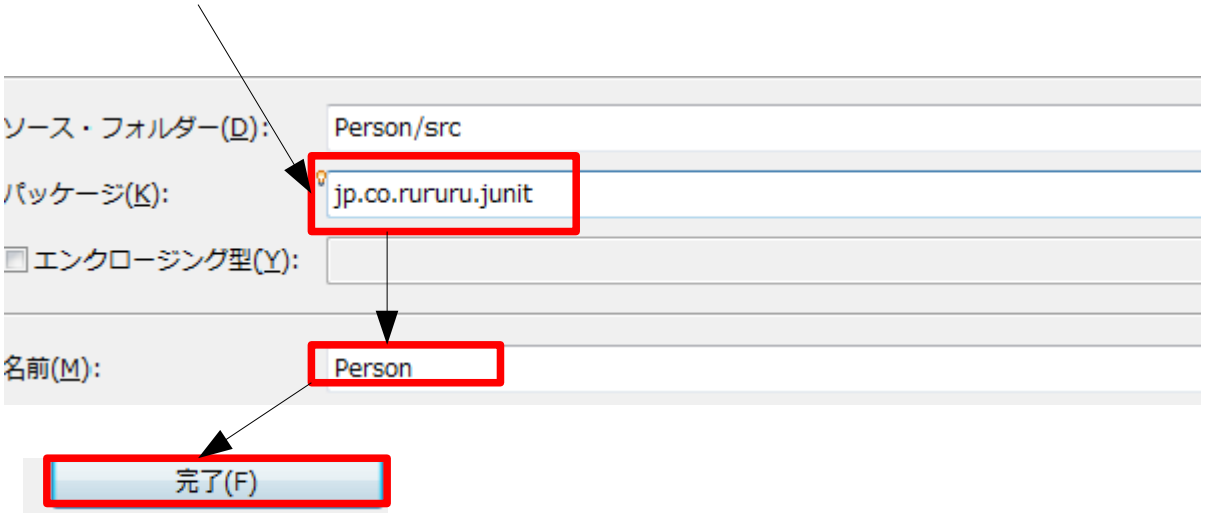


テスト (5/20)

(5)「Person」プロジェクトー「src」を選択し、右クリックメニューの「新規」－「クラス」を選択します。



(6)「パッケージ」に「jp.co.rururu.junit」、名前に「Person」を入力後に「完了」ボタンをクリックします。

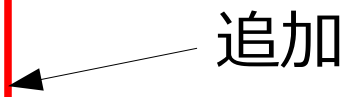


テスト (6/20)

(7) テスト対象クラスにコードを追加

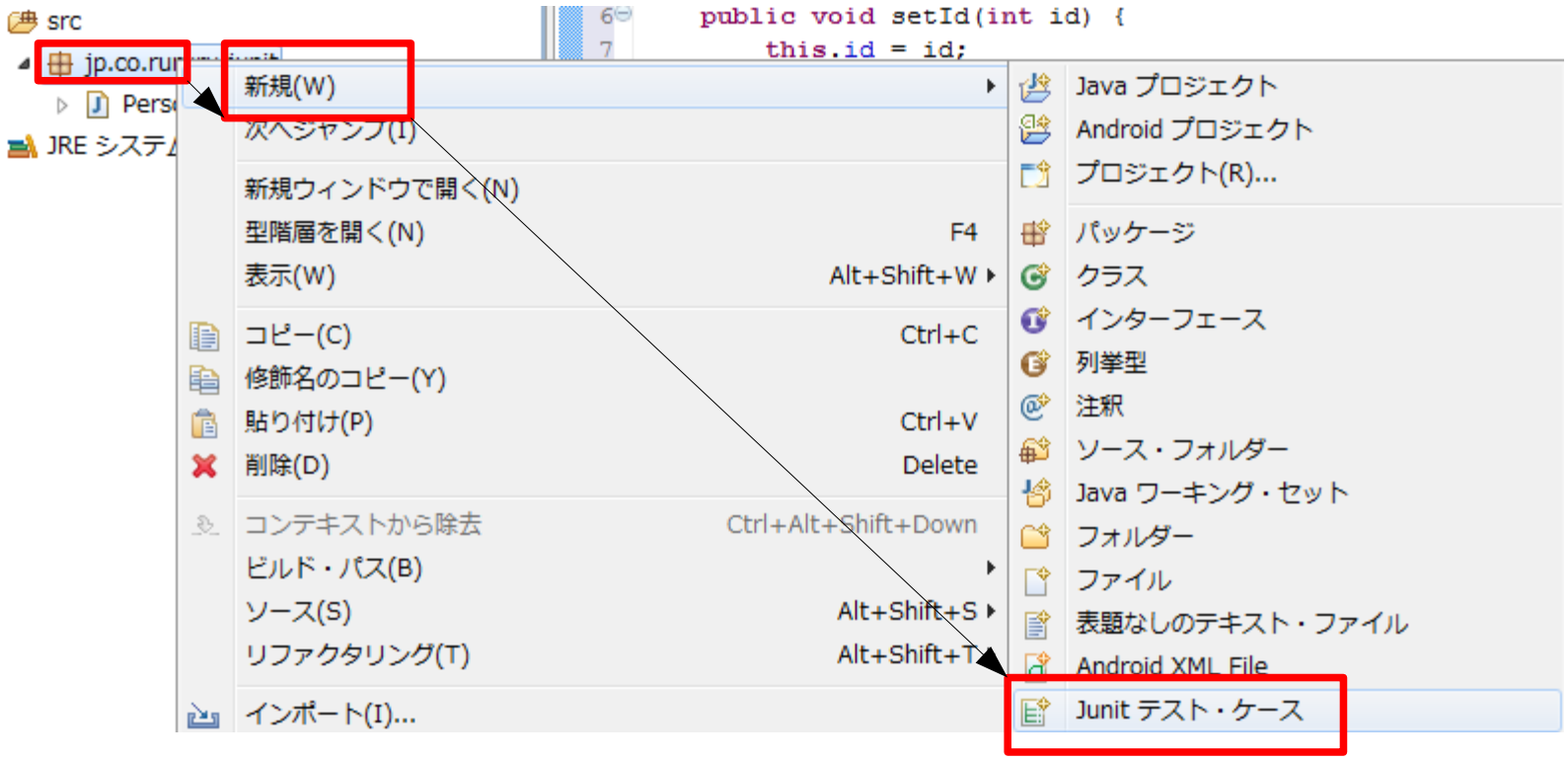
*Person.java

```
1  package jp.co.rururu.junit;
2
3  public class Person {
4      private int id = -1;
5      private String name = null;
6      public void setId(int id) {
7          this.id = id;
8      }
9      public int getId() {
10         return id;
11     }
12     public void setName(String name) {
13         this.name = name;
14     }
15     public String getName() {
16         return name;
17     }
18 }
```



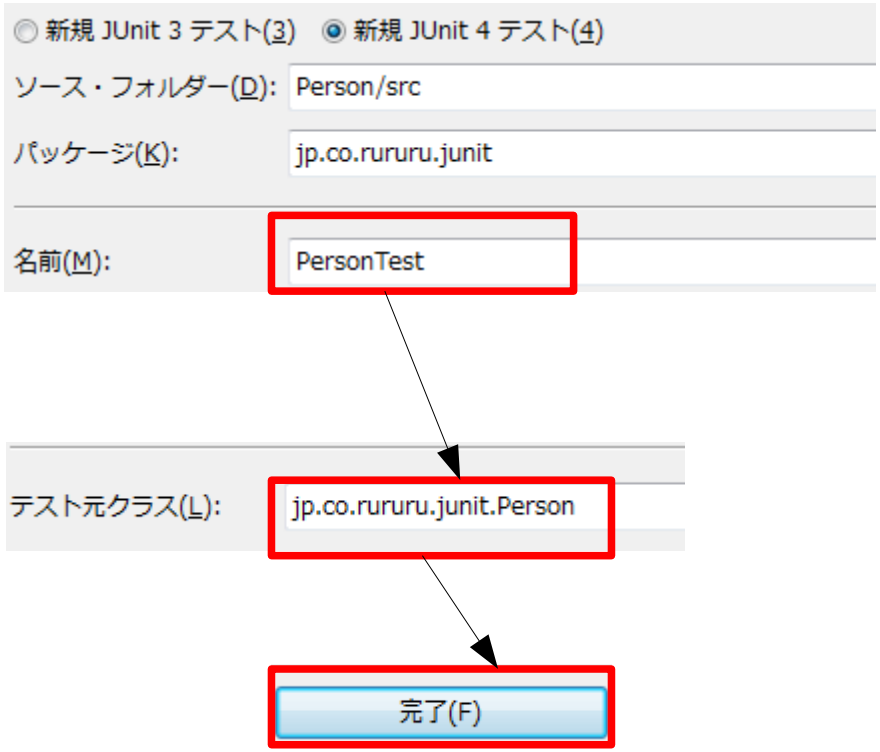
テスト (7/20)

(8)「jp.co.rurururu.junit」を選択し、右クリックメニューの「新規」－「JUnitテスト・ケース」を選択します。



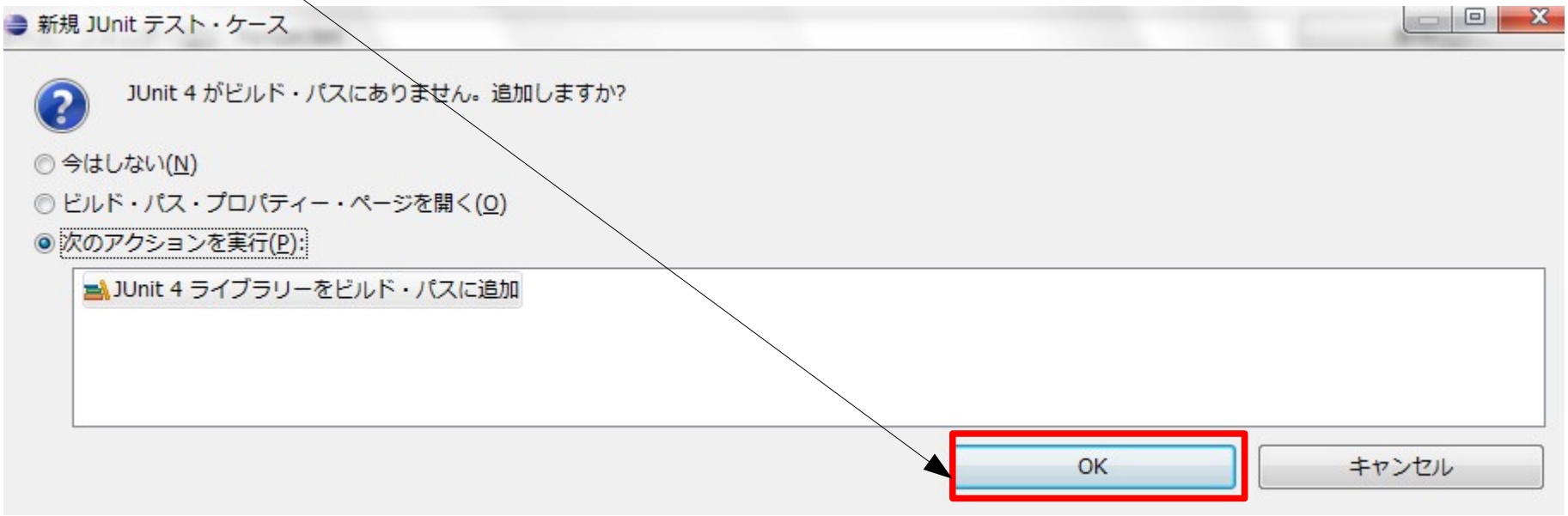
テスト (8/20)

(9) 名前に「PersonTest」、テスト元クラスに「jp.co.rururu.junit.Person」を入力後に「完了」ボタンをクリックします。



テスト (9/20)

(10)「OK」ボタンをクリックします。



テスト (10/20)

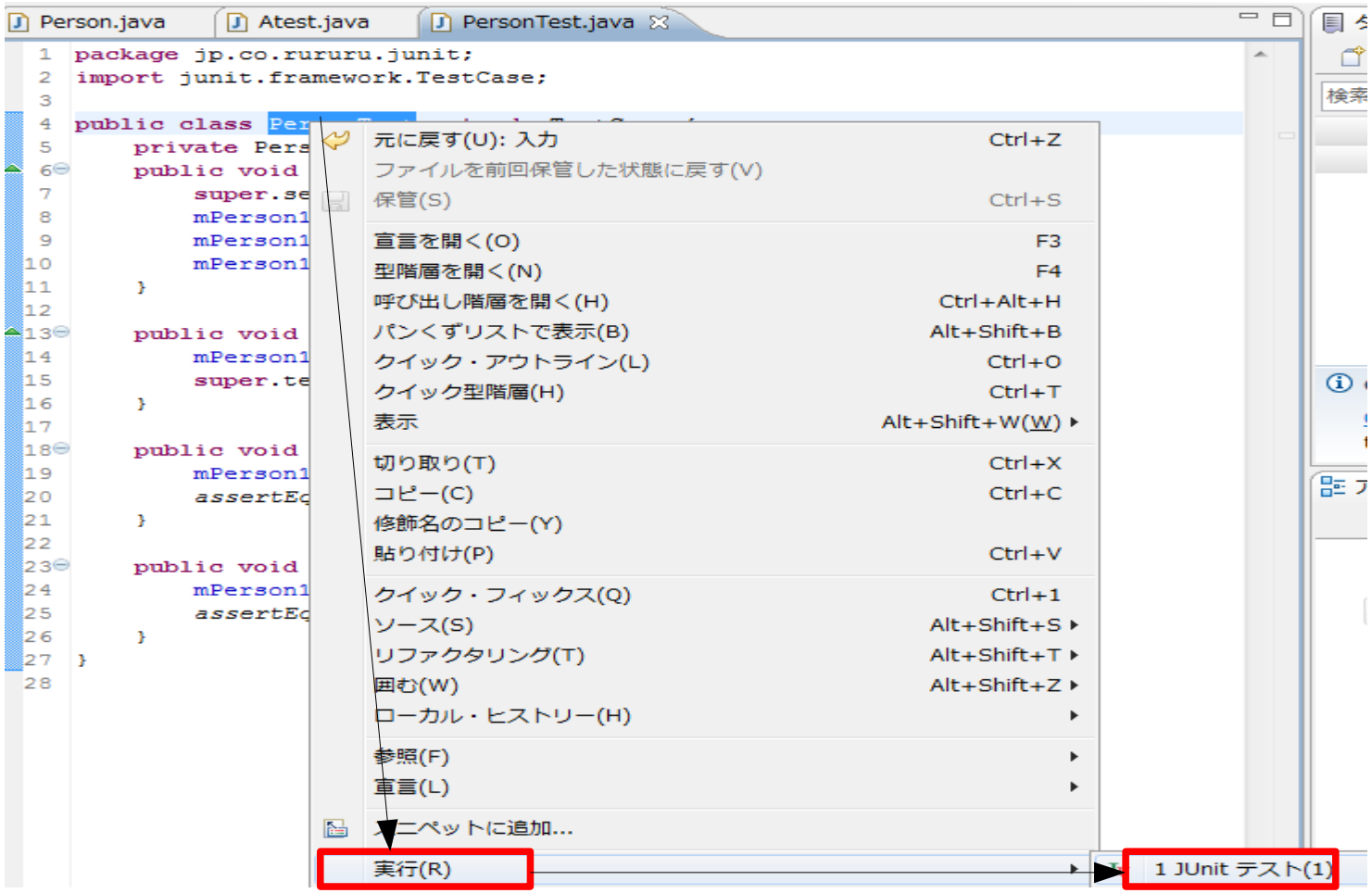
(11) テストクラスにコードを追加

追加

```
package jp.co.rurururu.junit;
import junit.framework.TestCase;

public class PersonTest extends TestCase {
    private Person mPerson1 = null;
    public void setUp() throws Exception  {
        super.setUp();
        mPerson1 = new Person();
        mPerson1.setId(10);
        mPerson1.setName("Taro Test");
    }

    public void tearDown() throws Exception {
        mPerson1 = null;
        super.tearDown();
    }

    public void testSetId() throws Exception {
        mPerson1.setId(20);
        assertEquals(20, mPerson1.getId());
    }

    public void testSetName() throws Exception {
        mPerson1.setName("Taro Test");
        assertEquals("Taro Test", mPerson1.getName());
    }
}
```

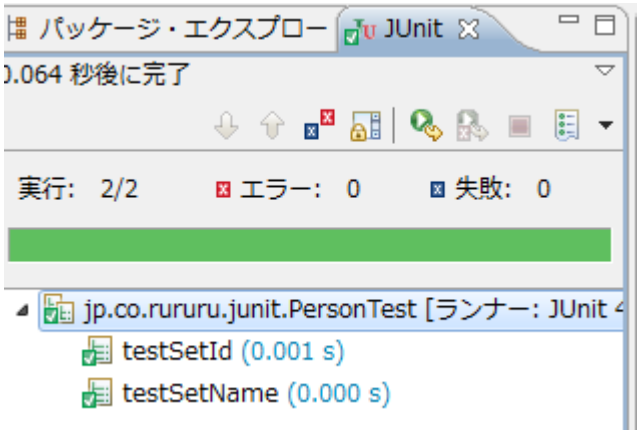
テスト (11/20)

(12) クラス名「PersonTest」を選択し、右クリックメニューの「実行」－「JUnit テスト」を選択します。



テスト (12/20)

(13) 以下の画面のように緑色のバーが表示された場合は全てのテストケース（今回は２つ）が成功しています。



テスト (13/20)

■**参考情報**

ソフトウェア技術ドキュメントを勝手に翻訳
7.1 Eclipse による ADT を使ったテスト
https://sites.google.com/a/techdoctranslator.com/jp/android/developing/testing/testing_eclipse
ソフトウェア技術ドキュメントを勝手に翻訳
l.1 テストの基礎
http://www.techdoctranslator.com/android/guide/testing/testing_android
AndroidアプリケーションをJUnitでテストする
http://itinfo.main.jp/tan/?p=35
JUnit/テストアプリケーションの作成
http://wikiwiki.jp/android/?JUnit%2F%A5%C6%A5%B9%A5%C8%A5%A2%A5%D7%A5%EA%A5%B1%A1%BC%A5%B7%A5%E7%A5%F3%A4%CE%BA%EE%C0%AE
Androidアプリのユニットテスト
http://d.hatena.ne.jp/masa-pon/20090313/1236905132
Androidアプリケーション　Junitテストの作り方　～その1～
http://edywrite.blogspot.com/2010/05/androidjunit1.html
JUnitチュートリアル
http://www.次世代創造機構.jp/android/androidLecture/JUnit/JUnit.html
AndroidでのUnitTestの始め方
http://www.yaunix.com/2011/01/08/androidでのunittestの始め方/
初めてのAndroidプロジェクト作成
http://blog.livedoor.jp/fivegangsters/archives/3444853.html
Androidのユニットテストを倒す（PureJavaライブラリ編）
http://d.hatena.ne.jp/esmasui/20100910/1284080432
Android Eclipse から ユニットテストを実行する
http://android-a-yan.com/blog/?p=322
Androidアプリケーションの作成とエミュレータでの実行 (はじめてのAndroidアプリケーション)
http://www.ipentec.com/document/document.aspx?page=android-first-application-build-and-execute

テスト (14/20)

3、Activityのテスト

■Activityについて

画面を表示するコンポーネント。ボタンやリストが配置された画面、Webページが表示されている画面、3Dグラフィックスが表示されている画面などはそれぞれがActivityです。ボタンのタッチやキーの入力など、ユーザからの入力を受けて処理を実行し結果を表示するユーザインターフェイス(UI)の役割を担います。

■参考情報

androidの単体テスト(AndroidTestCase)
http://blog.haw.co.jp/android/?p=471
AndroidでのUnitTestの始め方
http://www.yaunix.com/2011/01/08/androidでのunittestの始め方/
[Android]Instrumentationによるユニットテストでは直接ビューを触ってはいけない
http://d.hatena.ne.jp/Kazzz/20100311/p1
android JUnitでActivityのテストを行う
http://d.hatena.ne.jp/lounge1975/20110324/1300937575
Android開発でJUnit導入してみる。 その2 Activityのテスト自動化
http://blog.livedoor.jp/shizuku_kun/archives/51566264.html
Androidで別アクティビティを呼び出したことをユニットテストする方法
http://wp.krks.net/2011/01/androidで別アクティビティを呼び出したこ/
Androidメモ
http://www.saturn.dti.ne.jp/~npaka/android/LaunchMode/index.html
ActivityInstrumentationTestCase2
http://wikiwiki.jp/android/?ActivityInstrumentationTestCase2
Activity Testing
http://maruyama.cloud-market.jp/download/ActivityTesting1.pdf

テスト (15/20)

4、Serviceのテスト

■Serviceについて

バックグラウンドで処理をするコンポーネントです。ユーザがServiceを直接操作することはできません。
代わりにプロセス間通信の仕組みを持っており、AIDL(Android Interface Definition Language)という
書式でインターフェイスを定義することで、他のActivityからServiceに接続して操作することができます。

■参考情報

Serviceのユニットテスト
http://d.hatena.ne.jp/kaw0909/20110123/1295749470
Service Testing
https://sites.google.com/site/androidtestclub/translation/dev-guide/framework-topics/00-framework-topics-testing/service-testing
ソフトウェア技術ドキュメントを勝手に翻訳 -I.4 サービスのテスト-
http://www.techdoctranslator.com/android/guide/testing/service_testing
androidでJUnitを使ってテストケースを書く方法
http://fromnorth.blogspot.com/2008/12/androidjunit.html
[Android]Serviceのライフサイクルの動作確認
http://d.hatena.ne.jp/terurou/20100519/1274252852
Android: 特定の時間に起動するアプリ
http://kurotofu.sytes.net/kanji/fool/?tag=androidアプリ開発
カテゴリー別アーカイブ: Android入門
http://android-a-yan.com/blog/?cat=8
androidでのテスト
http://blog.livedoor.jp/areyard-code/archives/494863.html

テスト (16/20)

5、ContentProviderのテスト

■ContentProviderについて

アプリケーション間で情報を共有するコンポーネント。insert／update／deleteなどのインターフェイスが定義されている。内部にどのような形でデータを保存するかは実装に依存します。
ファイル／ＤＢ／ネットワーク上のストレージでも、他のアプリケーションは気にすることなく情報を取得できます。

■参考情報

AndroidでContentProviderのモックを使ったテストを行う
http://d.hatena.ne.jp/thorikawa/20101021/p1
ContentProviderのテストの仕方
http://abekatsu.blogspot.com/2011/03/contentprovider.html
JUnitチュートリアル
http://www.次世代創造機構.jp/android/androidLecture/JUnit/JUnit.html
Androidテスト入門
http://dev.c-lis.co.jp/nagoya/HansOn-JUnitandMonkey.pdf
Android Tips TextView内の文字列リンクから特定のActivtyを呼び出す
～ContentProviderとLinkifyの理解～
http://edywrite.blogspot.com/2011/03/android-tips-textviewactivty.html

テスト (17/20)

6、Androidのテストツール「Monkey(モンキー)」

■Monkeyについて

Android SDKに付属しているテストツール。クリックやタッチなどのユーザーの操作イベントやシステムのイベントをランダムに発生させることができます。 このツールを利用してストレステストを行えます。

■参考情報

Monkey
http://wikiwiki.jp/android/?Monkey
AndroidのテストツールMonkey
http://www.taosoftware.co.jp/blog/2009/04/android_monkey.html
monkeyテストツールを使う
http://d.hatena.ne.jp/siso9to/20101215/1292434449
Monkey(モンキー)
http://fancyfree.sakura.ne.jp/pg/index/android_intermediate026/
ソフトウェア技術ドキュメントを勝手に翻訳
Monkey - UI / アプリケーション エクササイザ
http://www.techdoctranslator.com/android/developing/tools/monkey
09 > Developing > Tools > Monkey
http://sites.google.com/site/androidtestclub/translation/dev-guide/developing/tools/09-developing-tools-monkey

テスト (18/20)

7、CIツール「Hudson(ハドソン)」

■Hudsonについて

川口 耕介さんが作ったJavaベースのCI(Continuous Integration：継続的インテグレーション)ツール。
Maven(メイヴァン／メイヴィン。Java用プロジェクト管理ツール)やAnt(アント。ビルドツール)などで記述されたビルドを定期的に実行し、ビルドの結果を監視。開発規模が大きい場合、Hudsonを利用してチームメンバがこまめに成果物を早い段階から統合することで、開発の効率化や品質の向上に役立ちます。
豊富なプラグインで機能拡張する仕組みもあります。

■参考情報

Hudson
http://hudson-ci.org/
Hudson CI -java.net-
http://java.net/projects/hudson/
Hudson -Wikipedia-
http://ja.wikipedia.org/wiki/Hudson
Hudson -日本語Wiki-
http://wiki.hudson-ci.org/display/JA/Hudson
Hudsonを使ったアジャイルな開発入門
http://gihyo.jp/dev/feature/01/hudson
イマドキのIDE事情
CIツールとIDEの連携 - EclipseからHudsonを利用する
http://journal.mycom.co.jp/column/ide/043/index.html
Java/Hudson
http://www.masatom.in/pukiwiki/Java/Hudson/
Gitコミット時にHudson自動ビルド
http://unicus.jp/skmk/archives/41
Hudsonで継続的インテグレーション
http://www.junglejava.jp/archives/2009/05/entry_1058.html
Hudson - ジョブの自動監視や自動テストを行えるCIツール
http://www.syboos.jp/opensource/bookmark/detail/hudson.html
Hudsonの使い道
http://forza.cocolog-nifty.com/blog/2009/02/hudson-a9dc.html
米Oracle、「Hudson」をEclipse Foundation下のプロジェクトとすることを提案
http://sourceforge.jp/magazine/11/05/06/0445201

テスト (19/20)

8、CIツール「Jenkins(ジェンキンス)」

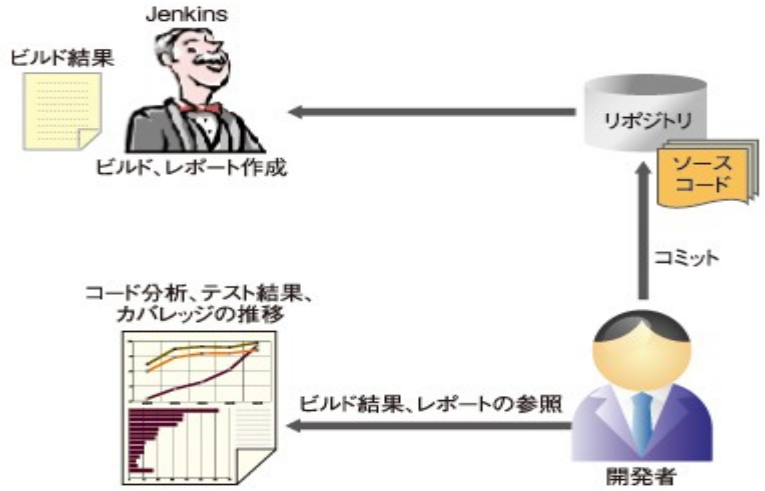
■Jenkinsについて

Oracle が「Hudson」の商標権およびプロジェクトの管理を主張したため、Hudson 開発コミュニティの投票の結果「Jenkins」への改名され開発がすすめられています。JenkinsがHudsonの後継と言われているので、今後はHudsonよりJenkinsの利用が多くなると思われます。

■Jenkinsの特徴

・ソースコードの統合・テストの自動化
・定期的なビルドによるコンパイルエラー、バグの早期発見
・メトリクスの自動取得
　コードチェックやカバレッジを取得するプラグインを導入すれば、ビルドごとのコードの統計を見ることができるようです。
・ビルド完了時の背景が変えることができる「Persona」プラグインなどが提供されている。

■Jenkinsの利用パターン



@ITの「「Hudson」改め「Jenkins」で始めるCI（継続的インテグレーション）入門」の図を引用

テスト (20/20)

■**参考情報**

Jenkins
http://jenkins-ci.org/
日本Jenkinsユーザー会
http://build-shokunin.org/
日本Jenkinsユーザー会 —Googleグループ—
http://groups.google.com/group/jenkinsci-ja
Jenkins -日本語 Wiki-
https://wiki.jenkins-ci.org/display/JA/Jenkins
第2回Jenkins勉強会」活動報告
http://gihyo.jp/news/report/2011/03/0901
第2回Jenkins勉強会
https://wiki.jenkins-ci.org/pages/viewpage.action?pageId=54723198
「Hudson」改め「Jenkins」で始めるCI（継続的インテグレーション）入門
http://www.atmarkit.co.jp/fjava/rensai4/devtool21/devtool21_1.html
jenkins エレガントなコードを書くための執事のすすめ
http://hiroki.jp/2011/03/09/1681/
Jenkins-CIとEUnitで自動テストを構築
http://erlang.dressingroom.jp/article/44040184.html
Jenkinsの最初のリリースとHudsonへのサポート
http://www.infoq.com/jp/news/2011/02/jenkins-hudson
元Hudsonチーム、米Oracleから独立後初の「Jenkins」リリース
http://sourceforge.jp/magazine/11/02/04/0328213
OracleからスピンアウトするHudsonコミュニティ、Jenkinsが実質的に後継
http://journal.mycom.co.jp/news/2011/01/31/080/index.html
継続的インテグレーションツール Hudson、Jenkins 名称に変更決定
http://slashdot.jp/it/article.pl?sid=11/02/01/0157228
Hudson、Jenkinsに改名
http://www.infoq.com/jp/news/2011/02/jenkins
CIツールのHudson，Oracleと交渉決裂しJenkinsに改名へ
http://gihyo.jp/dev/clip/01/orangenews/vol61/0004

Androidのソースコードをダウンロード

■**Androidのソースコードについて**

Androidはオープンソースなのでソースコードをダウンロードして参照することやビルドすることが可能です。
Windows環境ではCygwinをインストールして、repoを使用して取得することになります。

■**参考情報**

Androidソースコードをダウンロード・ビルドするには
http://www.adakoda.com/android/000118.html
Androidのソースコードをgitリポジトリからダウンロードする方法
http://magpad.jugem.jp/?eid=83
Android2.0,Eclairブランチがgitリポジトリに
http://www.swingingblue.net/mt/archives/002730.html
@ Repo って何だろ? -- 複数 git リポジトリーのためのツール
http://at-aka.blogspot.com/2009/02/repo-git.html
Androidのrepoコマンドでデフォルトではダウンロードされないリポジトリをダウンロードする
http://typex2.wordpress.com/2009/02/10/androidのrepoコマンドでデフォルトではダウンロードされ/
OHA版Androidのソース一式をGitHub上で公開
http://sites.google.com/site/devcollaboration/github
AndroidのソースコードをEclipseに取り込んで見た
http://www.cliph.net/wordpress/archives/780
Eclipse Git プラグイン ～ インストールからコミット、履歴の比較まで
http://android-a-yan.com/blog/?cat=7
Git使ってAndroidソースをダウンロードしてみた。
http://handalab.com/blog/android%E9%96%8B%E7%99%BA/223/
Repo と Git の使い方 (Using Repo and Git)
http://darutk-oboegaki.blogspot.com/2011/01/repo-git-using-repo-and-git.html

その他(1/4)

■Titanium(タイタニウム)

公式サイト

http://www.appcelerator.com/products/titanium-mobile-application-development/

HTML＋JavaScriptでiPhone/Androidアプリを作れるTitanium Mobileとは

http://www.atmarkit.co.jp/fsmart/articles/titanium01/01.html

iPhoneやAndroid用のネイティブアプリがJavaScriptで作れる「Titanium Mobile」がすごいらしいです

http://ke-tai.org/blog/2010/10/29/titaniummobile/

スマートフォン開発についての2つの発表があった「はてな技術勉強会 #2」の資料と動画が公開されています

http://ke-tai.org/blog/2010/11/29/hatenatech201011/

Titanium Mobileで作る！ iPhone／Androidアプリ

http://gihyo.jp/dev/serial/01/titanium

■The M Project

公式サイト

http://the-m-project.net/

iOS/Androidに特化したJavaScriptのMVCフレームワーク「The M Project」

http://blog.verygoodtown.com/2011/01/html5-javascript-mobile-application-development-framework-the-m-project/

■quickconnect

クイックコネクト

http://sourceforge.net/projects/quickconnect/

http://sourceforge.jp/projects/sfnet_quickconnect/

その他(2/4)

■PhoneGap(フォンギャップ)

公式サイト

http://electronegative/

PhoneGapフレームワークを使ったAndroidアプリをXperiaで動かしてみた！

http://d.hatena.ne.jp/esperia/20100914/1284482824

AndroidアプリのためのPhoneGap環境

http://d.hatena.ne.jp/speg03/20110414/1302789804

PhoneGapでAndroidアプリを作る

http://shokai.org/blog/archives/5451

PhoneGap Demo Application：HTML+CSS+JavaScriptでアプリを開発！Androidアプリ279

http://octoba.net/archives/tag/phonegap-demo-application

HTML＋CSS＋JavaScriptでiPhone/Androidアプリ開発「PhoneGap」

http://journal.mycom.co.jp/articles/2010/12/03/phonegap/index.html

■JavaScriptで作るiPhoneアプリケーション「Big Five」

http://www.moongift.jp/r/2008/11/big_five/

その他(3/4)

■Unity(ユニティ)

公式サイト

http://unity3d.com/

日本語訳

http://unity3d.com/japan/

Unity入門のための情報源

http://d.hatena.ne.jp/shuichi_h/20101228/1293558764

iOSアプリのAndroid移植も簡単なUnityの基礎知識

http://www.atmarkit.co.jp/fsmart/articles/unity01/01.html

■Flash Builder 4.5

Flash Builder 4.5でAndroidアプリを作ってみた

http://www.atmarkit.co.jp/fsmart/articles/fb4_5_android/01.html

Flash Builder 4.5でAndroidアプリ開発　～概要から作成手順の基本まで最速解説！

http://codezine.jp/article/detail/5872

Android女子部が初体験～Flash Builder 4.5で作るAIR for Androidアプリ

http://codezine.jp/article/detail/5895

その他(4/4)

■App Inventor(アップ インベンター)

App Inventor

http://appinventor.googlelabs.com/about/

「App Inventorでアプリ開発はどこまでできるのか」

http://anticlimactically/fsmart/index/appinventor.html

App Investorがすごいみたい

http://mc-alto.blogspot.com/2010/07/app-investor.html

Androidアプリ開発ツール「App Investor」

http://episode730.blog102.fc2.com/blog-entry-110.html

素人でもAndroidアプリを開発できる・・・かも　Googleの開発ツールApp Investor

http://tamayura.tumblr.com/post/806376505/android-google-app-investor

Google App InventorでAndroidアプリの作り方

http://www.infinity-dimensions.com/blog/archives/google-appinventor-development.html

「Google App Inventor」が一般公開されたそうです。

http://x68stage.ddo.jp/casper/sunbbs/sunbbs.cgi?2042

これで誰でも開発者？Androidアプリ簡単作成ソフト

http://freesoftweb.blog61.fc2.com/blog-entry-374.html

「Google App Inventor」の始めかた

http://x68stage.ddo.jp/casper/sunbbs/sunbbs.cgi?2043