LINTER VERT POUR L'ÉCOCONCEPTION LOGICIELLE

Exemple des applications Android

Olivier Le Goaër
GreenDays 2019@Anglet

Qui suis-je ?

→Maître de conférences en **Informatique** (Génie Logiciel)
http://olegoaer.perso.univ-pau.fr/

→Initiateur du projet **GREEN Pauware**
http://green.pauware.com

→Responsable du cours **Android** en Master 2
https://fr.slideshare.net/OlivierLeGoar/android-34623833 (+110k vues)

Écoconception logicielle

→**Doublement** de la consommation mondiale d'électricité pour le numérique à l'horizon 2030 (Rapport Greenpeace 2017)

→Les **logiciels verts** correspondent à des applications et services numériques moins gourmands en énergie

Data-center & Cloud

Web

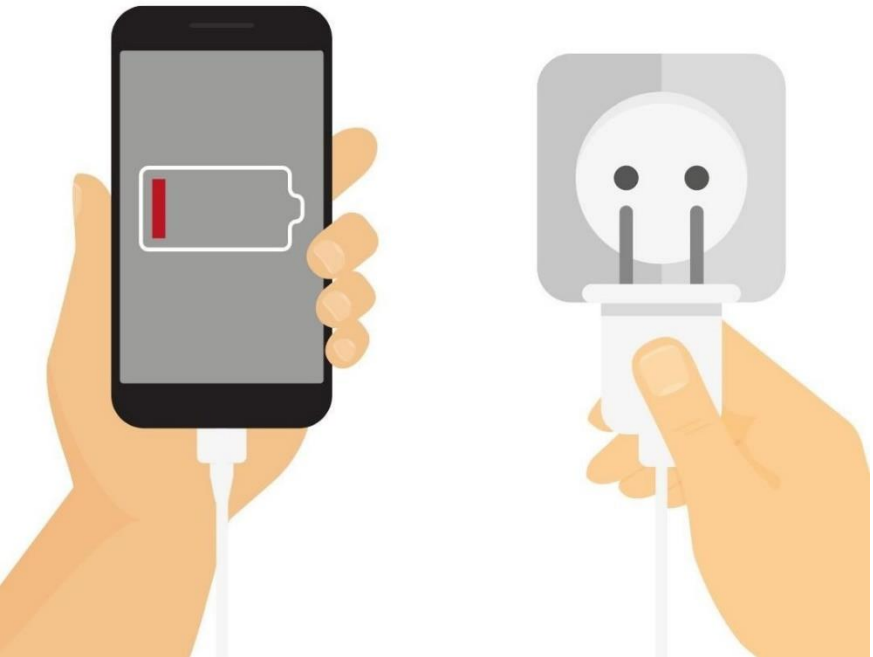
Mobile

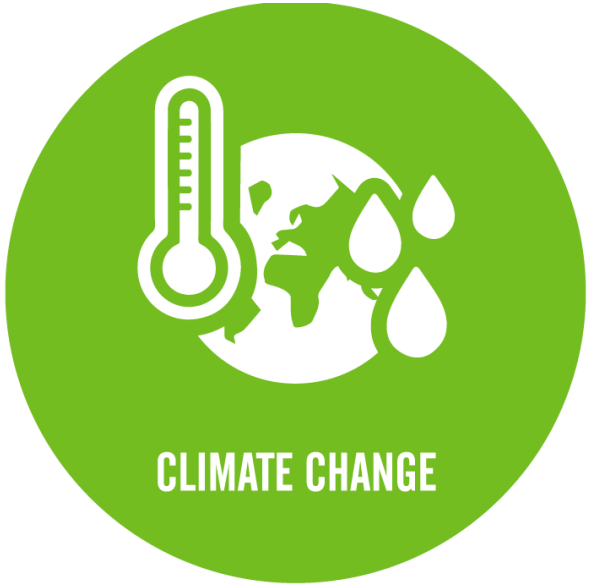
Apps mobiles : l'effet papillon

→**5 milliards** de possesseurs de smartphones dans le monde en 2020

→**194 milliards** d'applications téléchargées en 2018

→Les applications **mal conçues** vident excessivement les **batteries**





Qualité du code

→Clean code

Écrire du code plus « propre » pour améliorer la maintenance

→**Green code**

Écrire du code plus « vert » pour améliorer l'efficacité énergétique

→Inciter les développeurs à le faire au plus tôt, pour éviter d'avoir à rembourser une **dette technique** plus tard

Label de qualité (écolabel)

→Garantir un **standard** de qualité vert pour les apps mobile

Délivré par l'ADEME par exemple

→La pression des géants du secteur

Les sites web sont devenus massivement mobile-friendly à partir du moment où ce critère a été pris en compte par le SEO Google en 2015

Les applications mobiles deviendront **energy-friendly** dès que les magasins (Google Play, Apple Store) prendront ce critère en compte

Linter

→Outil d'**analyse statique** créé à l'origine pour les programmes C

→Pointe des **défauts** dans le code afin de maintenir la **qualité**

Bugs potentiels, mauvaises habitudes, ruptures des conventions, …

→On en trouve pour n'importe quel langage/technologie

ESLint (JavaScript), **Android lint** (Java/Kotlin), Pylint (Python), Golint (Go)

Qualité du code spécifique à Android

→La surconsommation d'énergie réside avant tout dans des défauts étroitement liés au **framework Android**

→Les défauts **orientés objet** jouent un rôle anecdotique

Blob Class, Feature Envy, Long Method, …

→Les défauts **idiomatiques** sont optimisés par la compilation JIT*

`for` vs. « `for-each` », `static final` pour les constantes primitives, wrappers vs. types primitifs, inline `get`/`set`, `System.arraycopy()`…

* Just-In-Time

13 défauts énergétiques Android

Nom	Sévérité	Portée
Everlasting Service	ERREUR	Code source
Sensor Leak	ERREUR	Code source
Internet In The Loop	ERREUR	Code source
Dark UI	AVERTISSEMENT	Manifest, Style, Drawable
Sensor Coalesce	AVERTISSEMENT	Code source
Durable Wake Lock	AVERTISSEMENT	Code source
Uncompressed Data Transmission	AVERTISSEMENT	Code source
Rigid Alarm	AVERTISSEMENT	Code source, Gradle
Dirty Boot	AVERTISSEMENT	Code source, Manifest
Keep Screen On	INFORMATION	Code source, Layout
Keep CPU On	INFORMATION	Code source, Manifest
Battery-Efficient Location	INFORMATION	Code Source, Gradle
Bluetooth Low-Energy	INFORMATION	Code source

Problèmes critiques

Améliorations possibles

Bonnes/Mauvaises pratiques

Sensor Leak

```
public class SensorActivity extends Activity implements SensorEventListener {
    private final SensorManager mSensorManager;
    private final Sensor mAccelerometer;

    public SensorActivity() {
        mSensorManager = (SensorManager)getSystemService(SENSOR_SERVICE);
        mAccelerometer = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
    }

    protected void onResume() {
        super.onResume();
        mSensorManager.registerListener(this, mAccelerometer,
                                        SensorManager.SENSOR_DELAY_NORMAL);
    }

    protected void onPause() {
        super.onPause();
        mSensorManager.unregisterListener(this);
    }

    public void onAccuracyChanged(Sensor sensor, int accuracy) { }
    public void onSensorChanged(SensorEvent event) { }
}
```

*Always make sure to **disable** sensors you don't need, especially when your activity is **paused**. Failing to do so can **drain the battery in just a few hours**. Note that the system will not disable sensors automatically when the screen turns off*. [Doc]

acquisition/libération

cycle de vie écran

Keep Screen On

```java
public class MainActivity extends Activity {
  @Override
  protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON);
  }
}
```

*To **avoid draining the battery**, an Android device that is left idle quickly falls **asleep**. However, there are times when an application needs to wake up the screen **keep it awake** to complete some work*. [Doc]

Version programmatique

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

Version déclarative

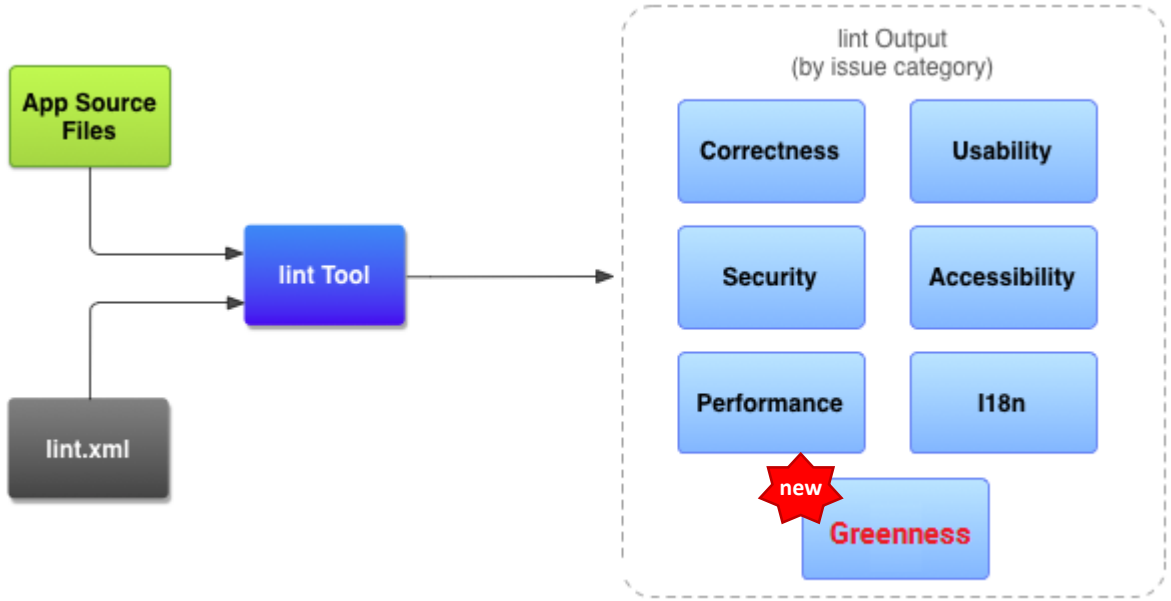
Android lint se met au vert

→Android lint est intégré à l'IDE **Android Studio**

Analyse à la volée, rapport d'inspection, quick fix

→Android lint est livré avec ~ 350 vérifications dans 6 catégories

Seulement **3** traitent réellement des économies d'énergie



Preuve de concept

→Télécharger http://green.pauware.com/android-lint/greenchecks.jar

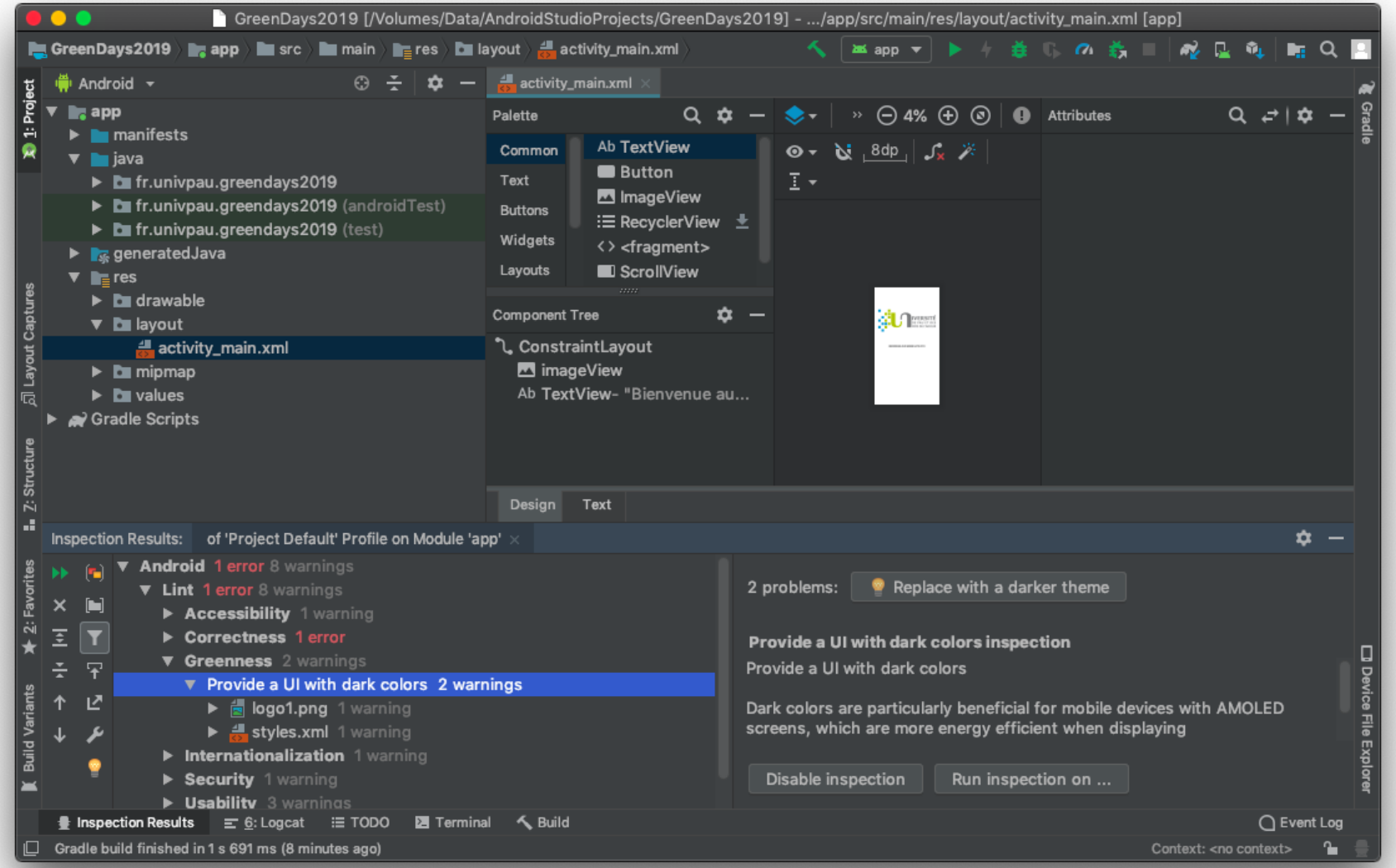
→Placer l'archive dans `~/.android/lint/`

→Lancer Android Studio. Ouvrir un projet.

→Menu *Analyze > Inspect Code**

*au besoin, sélectionner uniquement votre module App

Screenshot (*Dark UI*)



Conclusion

→Apps mobiles « vertes par conception » (**green-by-design**)

→**Praticable/acceptable** dans un scénario de développement réel

→Seul le développement **natif** Android est concerné pour l'instant

→Beaucoup d'autres défauts énergétiques restent à **identifier**…

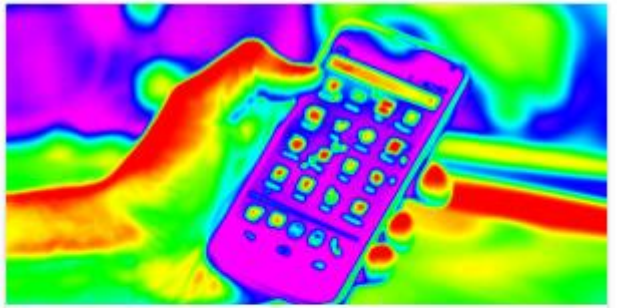
Quelques réflexions…

Génie logiciel vert & dette technique verte

Olivier Le Goaër sur LinkedIn

Vers un étiquetage énergétique des logiciels ?

Olivier Le Goaër sur LinkedIn



Diagnostic de Performance Énergétique Logiciel (DPE-L)

Olivier Le Goaër sur LinkedIn

MERCI

Avez-vous des questions ?