Android-II

Cours Android II

9-14 janvier 2017 Cesi Nanterre

WEB SERVICES

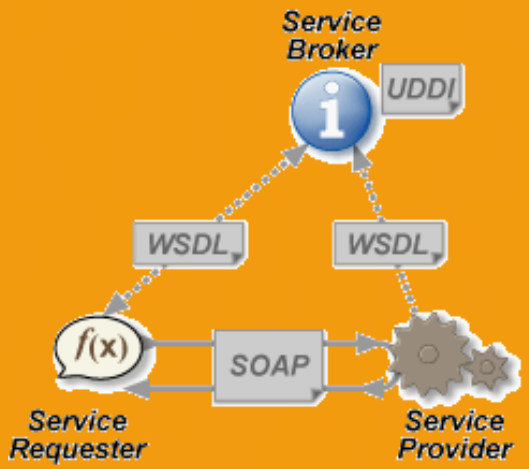
WEB SERVICES

- Généralités
  - Une méthode de communication entre deux systèmes électroniques au travers du Web.
  - une fonction/ressource logicielle aaccessible à une adresse réseau au travers du Web ou d'un Cloud.
- Le W3C définit les Web Services Architecture
  - Un service Web dispose d'une interface dans un format interprétable par une machine.
  - Les systèmes interagissent avec un Web Service conformément à ses modes opératoires en utilisant des messages de type SOAP (Simple Access Protocol) ou autres , typiquement au format XML et véhiculés en HTTP en conjonction avec l'utilisation d'autres standards Web.
- La plupart des Web services ne mettent pas en oeuvre des architectures complexes.
  - Deux grandes classes de Web services:
    - REST- Web services / REST – Compatibles Web Services
    - Arbitrary Web services, dans lesquels le service expose un ensemble d'opérations arbitraires .

WEB SERVICES

- Les Web Services sont un ensemble de moyens permettant d'exposer une API au travers d'un réseau et de façon à s'abstraire des technologies de mise en oeuvre (Technology-Neutral) .
- Les standrads de Web services sont encore en évolution.
  - Des tendances émergent:
    - SOAP pour les services de communication,
    - WSDL pour les services de description,
    - UDDI pour l'enregistrementeet la découverte de services,
    - BPEL(Business Process Execution Language) pour la composition de services .
    - Les tentatives de spécifications WS* sont pléthoriques et abordent l'ensemble du spectre des aspects clés des activites Web Service tels que : fiabilité des messages securité, données privées, coordination, traitement des événements ...
  - SOAP est un standard complexe. Quelques packages existent sous Android.

Architectures Web Service



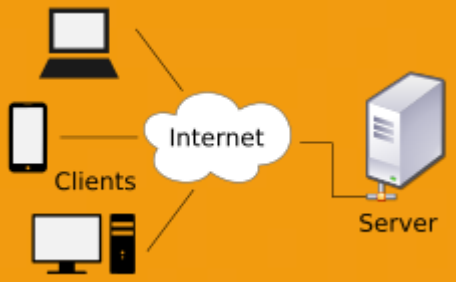
WSDL,
Web Services Description Language
UDDI
Universal Description Discovery and Infrastructure
SOAP
Simple Object Access Protocol

Limitations
Les services Web autres que RestFul sont complexes et construits sur des solutions logicielles non opensource

Problèmes potentiels de performance
Messages au format XML et SOAP/HTTP pour envelopper et transporeter les messages.

SOAP est un protocole et non pas une architecture

Web Service Architectures



RestFull

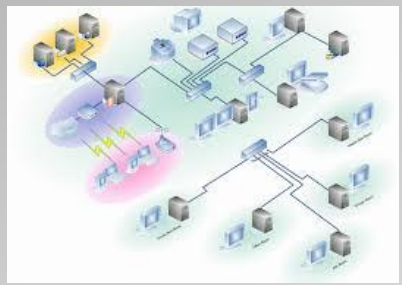
- Client-server
- Stateless
- Cacheable
- Layered system
- Code on demand
- Uniform interface

WEB SERVICES

- Ex de catalogues de web services
  - http://www.publicapis.com/
  - http://www.webservicex.net
  - http://www.programmableweb.com/
  - http://www.webservicelist.com/

WEB SERVICES

- Rappels sur les NetWork
- NetWork: groupe d'ordinateurs interconnectés



- TCP/IP
  - Node/Noeud: chaque équipement adressé sur le réseau est un node
  - Protocoles: ensemble de règles prédéfinies et agrées permettant la communication sur un réseau
  - Protocoles organisésen couches
    - TCP/IP stack
    - Link Layer—Physical device address resolution protocols such as ARP and RARP
    - Internet Layer—IP itself, which has multiple versions, the ping protocol, and
    - ICMP, among others
    - Transport Layer—Different types of delivery protocols such as TCP and UDP
    - Application Layer—Familiar protocols such as HTTP, FTP, SMTP, IMAP, POP, DNS, SSH, and SOAP

WEB SERVICES

- IP (Internet Packet)
  - IP adress ipv4 = 32 bits, ipv6 = 48 bits
- Delivery Protocols
  - TCP UDP (user datagram protocol) delivery protocols used over TCP/IP
- Application Protocols
  - (Simple Mail Transfer Protocol) SMTP , HTTP
- Clients and servers
  - Ports
    - Well-known ports—0 through 1023
    - Registered ports—1024 through 49151
    - Dynamic and/or private ports—49152 through 65535

WEB SERVICES

- Etat du réseau
  - Utiliser ConnectivityManager

```
@Override
public void onStart() {
super.onStart();
ConnectivityManager cMgr = (ConnectivityManager)
this.getSystemService(Context.CONNECTIVITY_SERVICE);
NetworkInfo netInfo = cMgr.getActiveNetworkInfo();
this.status.setText(netInfo.toString());
}
```

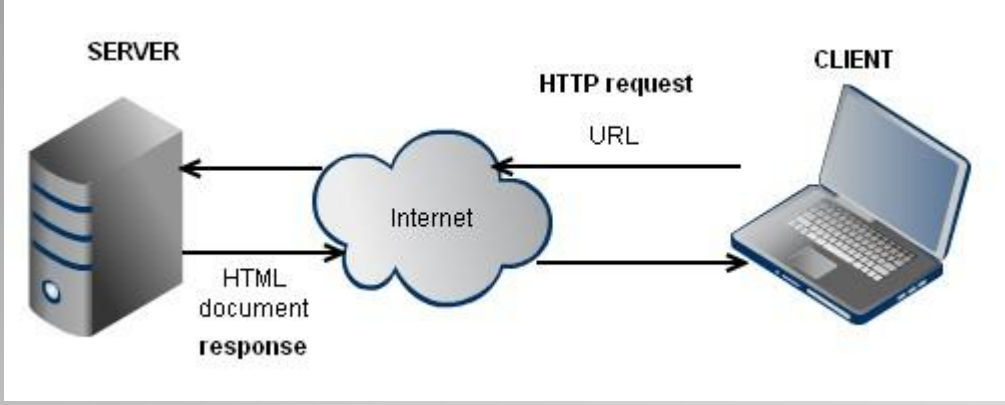
  - Connexion au réseau en préalable de l'accès

WEB SERVICES

- Communication à partir d'un server socket
  - Server Socket
    - Stream to read/write raw bytes
    - get local IP using ipconfig

WEB SERVICES

- Communication basée sur le protocole HTTP
  - http



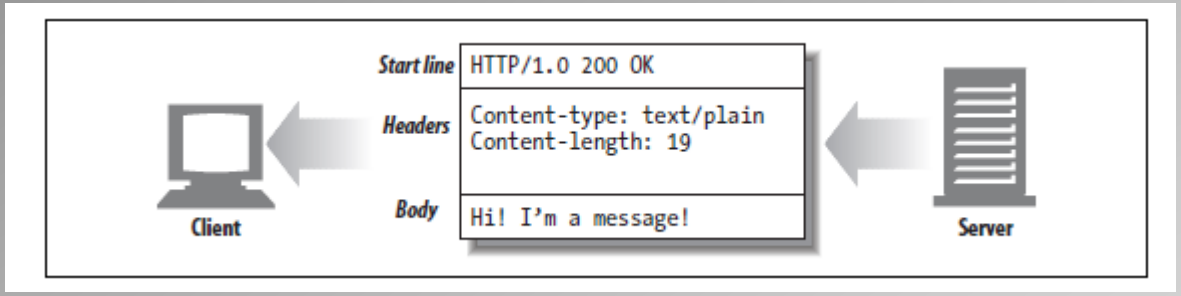
  - Ressources hébergées sur serveurs
    - Db, files, images, html pages, Services …
    - Data types définis par leur type MIME
    - Chaque ressource a un identificateur URI (uniform ressource identifier), URL (uniform ressource locator) qui définit sa localisation

WEB SERVICES

- URL http://www.fast.hardware.com/specials/hammer.gif
  - Scheme : http://
  - Adress : www.fast.hardware.com
  - Ressource : specials/hammer.gif
- Transactions
  - Requetes
  - Reponses
  - Messages
  - Methodes
    - GET — Send named resource from the server to the client.
    - POST — Send client data into a server gateway application.
    - PUT — Store data from client into a named server resource.
    - DELETE — Delete the named resource from a server.
    - HEADER — Send HTTP headers from the response for the named resource.
  - Status codes
    - 200 Ok …

WEB SERVICES

- Messages
- 3 sections



  - VERB URL VERSION (Start line)
  - HEADERS
  - BODY

```
GET / HTTP/1.1
Accept: text/*
Host: www.microsoft.com

HTTP/1.1 200 OK
Date: Thu, 01 Jun 2006 06:34:41 GMT
Server: Microsoft-IIS/6.0
P3P: CP="ALL IND DSP COR ADM CONo CUR CUSo IVAo IVDo PSA PSD TAI TELo OUR SAMo CNT COM INT NAV ONL PHY PRE PUR UNI"
X-Powered-By: ASP.NET
X-AspNet-Version: 2.0.50727
Cache-Control: private
Content-Type: text/html; charset=utf-8
Content-Length: 30430
```

WEB SERVICES

- Utilise le protocole HTTP
- Android supporte deux types de clients Http
- HttpURLConnection

  Ou
- HttpClient (Apache)

- Pattern d'utilisation de HttpURLConnection
  - Methodes Http
    - Get utilisé par default
    - Post si setDoOutput(true)
    - Supporte aussi put delete head trace setRequestMethod(String).

WEB SERVICES

- Performance
  - Problème ANR, Android interdit l'appel d'Http dans le thread principal
  - solution
    - Lancer les requête Http dans un thread séparé / background
  - Vous ne pouvez pas mettre à jour les vues/widgets hors du thread principal
  - Utiliser asynctask ou autre mecanisme de gestion des taches dans des threads séparés

Managing background activities

Executing background tasks

- Rappel
  - Java est multithreading
  - Implementations Android (basées sur Java)
    - asyncTask
    - Handler, messages
    - Download manager
    - IntentService

Executing background tasks

- Java Multithreading
- Two ways to implement a thread
  - Create class to Extend Thread
  - Define run()
  - Start
  - Example

```
class PrimeThread extends Thread {
      long minPrime;
      PrimeThread(long minPrime) {
         this.minPrime = minPrime;
      }

      public void run() {
         // compute primes larger than minPrime
          . . .
      }
   }
//start the thread
PrimeThread p = new PrimeThread(143);
   p.start();

//notes
thread stops
     On exit() call
     After execiting run()
A thread has a priority level
```

Executing background tasks

- Java Multithreading
  - Implement Runnable interface
    - Create class to implement runnable
    - Define run()
    - Start
    - Example

```
class PrimeRun implements Runnable {
      long minPrime;
      PrimeRun(long minPrime) {
         this.minPrime = minPrime;
      }

      public void run() {
         // compute primes larger than minPrime
         . . .
      }
   }
//start it
PrimeRun p = new PrimeRun(143);
   new Thread(p).start();
```

Executing background tasks

- Android
  - asyncTask, used for short operations (few seconds)
    - Allows to perform background operations
    - Publish results in UI thread
    - Uses 3 types :Parameters , Progress, Result
    - Uses 4 steps : onPreExec

```
private class DownloadFilesTask extends AsyncTask<URL, Integer, Long> {
    protected Long doInBackground(URL... urls) {
        int count = urls.length;
        long totalSize = 0;
        for (int i = 0; i < count; i++) {
            totalSize += Downloader.downloadFile(urls[i]);
            publishProgress((int) ((i / (float) count) * 100));
            // Escape early if cancel() is called
            if (isCancelled()) break;
        }
        return totalSize;
    }
    protected void onProgressUpdate(Integer... progress) {
        setProgressPercent(progress[0]);
    }
    protected void onPostExecute(Long result) {
        showDialog("Downloaded " + result + " bytes");
    }
}

//Update UI on progress triggered by publicProgress
```

Executing background tasks Asynctask

- Android
  - asyncTask, 4 steps
  - onPreExecute()
    - invoked on the UI thread before the task is executed. onUpdate
  - doInBackground(Params...)
    - invoked on the background thread immediately after onPreExecute() finishes executing.
    - The result of the computation must be returned by this step and will be passed back to the last step onPostExecute().
    - This step can also use publishProgress(Progress...). Values are published on the UI thread, in the onProgressUpdate(Progress...) step.
  - onProgressUpdate(Progress...), invoked on the UI thread after a call to publishProgress(Progress...).
    - The timing of the execution is undefined.
  - onPostExecute(Result), invoked on the UI thread after the background computation finishes.

Executing background tasks

- Android
  - Handler et Messages
    - Permet d'émettre /Send et traiter/exécuter des messages ou des Runnable
    - File d'attente de messages
    - Traitement dans un thread séparé
    - Echéancier possible de traitement des Messages et des runnable depuis la file d'attente
    - Vous envoyer / send a message dans la file d'attente
    - Vous postez / post des runnable
    - Le handler associé à un thread/fille d'attente est accessible depuis tout autre thread (Intra process)

Loaders

- Available to every Activity and Fragment.
- Provide asynchronous loading of data.
- Monitor the source of their data and deliver new results when the content changes.
- Automatically reconnect to the last loader's cursor when being recreated after a configuration change.

Loaders

- Loaders run on separate threads to prevent janky or unresponsive UI.
- Loaders simplify thread management by providing callback methods when events occur.
- Loaders persist and cache results across configuration changes to prevent duplicate queries.
- Loaders can implement an observer to monitor for changes in the underlying data source.

Download Manager

- System service that handles long-running HTTP downloads.

Structure des données

- XML
- JSON
- Voir cours I

WEBSERVICES

- Data representation/description
  - Languages
    - XML
    - JSON
    - Conçus pour être
      - Auto descriptif
      - Traité sur tout type de plateforme
    - Utilisés comme format d'échange et de stockage de données (NoSql) (ex Firebase)
    - Choix fréquent dans la représentation de données de web services

JSON

- Structure

```
{"menu": {
  "id": "file",
  "value": "File",
  "popup": {
    "menuitem": [
      {"value": "New", "onclick": "CreateNewDoc()"},
      {"value": "Open", "onclick": "OpenDoc()"},
      {"value": "Close", "onclick": "CloseDoc()"}
    ]
  }
}}
```

- Parsing

Même chose en XML:

```
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

JSON

- Structure

```
{"widget": {
    "debug": "on",
    "window": {
        "title": "Sample Konfabulator Widget",
        "name": "main_window",
        "width": 500,
        "height": 500
    },
    "image": {
        "src": "Images/Sun.png",
        "name": "sun1",
        "hOffset": 250,
        "vOffset": 250,
        "alignment": "center"
    },
    "text": {
        "data": "Click Here",
        "size": 36,
        "style": "bold",
        "name": "text1",
        "hOffset": 250,
        "vOffset": 100,
        "alignment": "center",
        "onMouseUp": "sun1.opacity = (sun1.opacity / 100) * 90;"
    }
}}
```

The same text expressed as XML:

```
<widget>
    <debug>on</debug>
    <window title="Sample Konfabulator Widget">
        <name>main_window</name>
        <width>500</width>
        <height>500</height>
    </window>
    <image src="Images/Sun.png" name="sun1">
        <hOffset>250</hOffset>
        <vOffset>250</vOffset>
        <alignment>center</alignment>
    </image>
    <text data="Click Here" size="36" style="bold">
        <name>text1</name>
        <hOffset>250</hOffset>
        <vOffset>100</vOffset>
        <alignment>center</alignment>
        <onMouseUp>
            sun1.opacity = (sun1.opacity / 100) * 90;
        </onMouseUp>
    </text>
</widget>
```

WEBSERVICES-XML

- Data representation/description
  - XML document structure
    - Organized as a tree structure starting from root

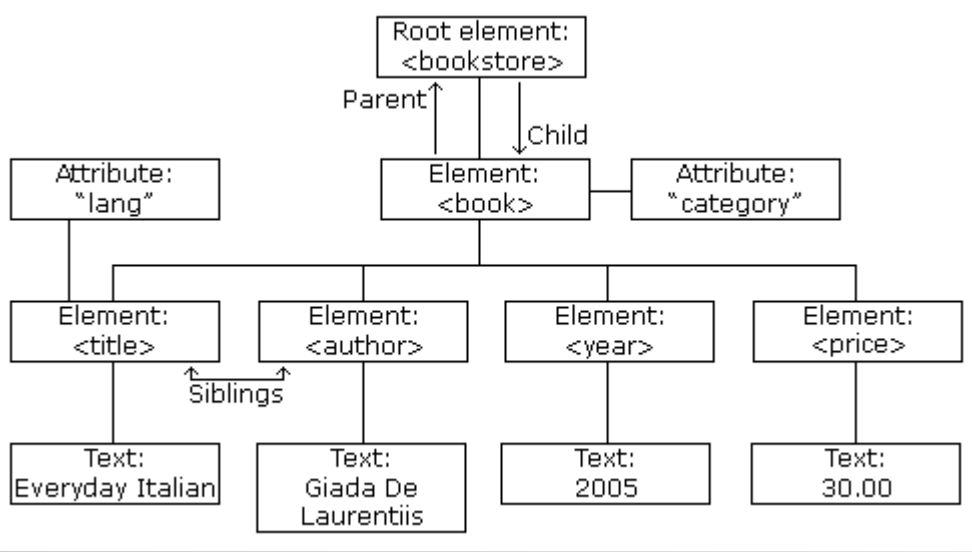
```
<?xml version="1.0" encoding="UTF-8"?>
<note>
 <dest>Marc</dest>
 <de>Jeanne</de>
 <sujet>Rappel</sujet>
 <cont>C'est l'anniversaire de Julie ce week-end!</cont>
</note>
```

  - Contains root element <note>
  - Contains 4 child elements : dest, de, sujet,cont

```
<root>
 <child>
  <subchild>.....</subchild>
 </child>
</root>
```

WEBSERVICES-XML

- Data representation/description
  - Elements can have text content and attributes
  - Parent, children, sibling



WEBSERVICES-XML

- Data representation/description
  - Elements

```
<bookstore>
 <book category="COOKING">
  <title lang="en">Everyday Italian</title>
  <author>Giada De Laurentiis</author>
  <year>2005</year>
  <price>30.00</price>
 </book>
 <book category="CHILDREN">
  <title lang="en">Harry Potter</title>
  <author>J K. Rowling</author>
  <year>2005</year>
  <price>29.99</price>
 </book>
 <book category="WEB">
  <title lang="en">Learning XML</title>
  <author>Erik T. Ray</author>
  <year>2003</year>
  <price>39.95</price>
 </book>
</bookstore>
```

WEBSERVICES-XML

- Data representation/description
- Language is case sensitive
  - Well formed document
    - Must have a root element
    - Open tag must have a closing tag
    - Elements must be properly nested
    - Attributes values must be quoted
    - Predefined entity references

&lt;	<	less than
&gt;	>	greater than
&amp;	&	ampersand
&apos;	'	apostrophe
&quot;	"	quotation mark

Comments

<!-- This is a comment -->

WEBSERVICES-XML

- Data representation/description
  - Well formed document
    - Element names must start with a letter or underscore
    - Element names cannot start with the letters xml (or XML, or Xml, etc)
    - Element names can contain letters, digits, hyphens, underscores, and periods
    - Element names cannot contain spaces

WEBSERVICES-XML

- Data representation/description
  - Attributes are quoted
    - <gangster name='George "Shotgun" Ziegler'>
    - <gangster name="George "Shotgun" Ziegler">
  - Elements versus attribute

```
<person gender="female">
 <firstname>Anna</firstname>
 <lastname>Smith</lastname>
</person>

<person>
 <gender>female</gender>
 <firstname>Anna</firstname>
 <lastname>Smith</lastname>
</person>
```

WEBSERVICES-JSON

- Data representation/description
  - JSON Java Script Open Notation
  - alternative à XML

```
{"employees":[
   {"firstName":"John", "lastName":"Doe"},
   {"firstName":"Anna", "lastName":"Smith"},
   {"firstName":"Peter", "lastName":"Jones"}
]}
```

```
<employees>
   <employee>
      <firstName>John</firstName> <lastName>Doe</lastName>
   </employee>
   <employee>
      <firstName>Anna</firstName> <lastName>Smith</lastName>
   </employee>
   <employee>
      <firstName>Peter</firstName> <lastName>Jones</lastName>
   </employee>
</employees>
```

WEBSERVICES-JSON

- Data representation/description
  - JSON Java Script Open Notation
  - An alternative to XML
  - JSON
    - Data
      - Data: pair name/value
      - "firstName":"John«
      - Data separator ,
    - Object
      - {…}
      - {"firstName":"John", "lastName":"Doe"}
    - Array
      - […]

```
"employees":[
 {"firstName":"John", "lastName":"Doe"},
{"firstName":"Anna", "lastName":"Smith"},
{"firstName":"Peter","lastName":"Jones"}
]
```

MIME type for JSON text is "application/json

Generating XML

- Create XML data

```
public static String writeUsingNormalOperation(Study study) {
    String format =
            "<?xml version='1.0' encoding='UTF-8'?>" +
            "<record>" +
            "   <study id='%d'>" +
            "      <topic>%s</topic>" +
            "      <content>%s</content>" +
            "      <author>%s</author>" +
            "      <date>%s</date>" +
            "   </study>" +
            "</record>";
    return String.format(format, study.mId, study.mTopic, study.mContent,
study.mAuthor, study.mDate);
}
```

Generating XML

- Create XML data using DOM

```
public static String writeUsingDOM(Study study) throws Exception {
    Document doc = DocumentBuilderFactory.newInstance().newDocumentBuilder().newDocument();
    // create root: <record>
    Element root = doc.createElement(Study.RECORD);
    doc.appendChild(root);

    // create: <study>
    Element tagStudy = doc.createElement(Study.STUDY);
    root.appendChild(tagStudy);
    // add attr: id =
    tagStudy.setAttribute(Study.ID, String.valueOf(study.mId));

    // create: <topic>
    Element tagTopic = doc.createElement(Study.TOPIC);
    tagStudy.appendChild(tagTopic);
    tagTopic.setTextContent(study.mTopic);

    // create: <content>
    Element tagContent = doc.createElement(Study.CONTENT);
    tagStudy.appendChild(tagContent);
    tagContent.setTextContent(study.mContent);

    // create: <author>
    Element tagAuthor = doc.createElement(Study.AUTHOR);
    tagStudy.appendChild(tagAuthor);
    tagAuthor.setTextContent(study.mAuthor);

    // create: <date>
    Element tagDate = doc.createElement(Study.DATE);
    tagStudy.appendChild(tagDate);
    tagDate.setTextContent(study.mDate);

    // create Transformer object
    Transformer transformer = TransformerFactory.newInstance().newTransformer();
    StringWriter writer = new StringWriter();
    StreamResult result = new StreamResult(writer);
    transformer.transform(new DOMSource(doc), result);

    // return XML string
    return writer.toString();
}
```

Generating XML

- Create XML data using XML serializer

```
public static String writeUsingXMLSerializer(Study study) throws Exception {
    XmlSerializer xmlSerializer = Xml.newSerializer();
    StringWriter writer = new StringWriter();

    xmlSerializer.setOutput(writer);
    // start DOCUMENT
    xmlSerializer.startDocument("UTF-8", true);
    // open tag: <record>
    xmlSerializer.startTag("", Study.RECORD);
    // open tag: <study>
    xmlSerializer.startTag("", Study.STUDY);
    xmlSerializer.attribute("", Study.ID, String.valueOf(study.mId));

    // open tag: <topic>
    xmlSerializer.startTag("", Study.TOPIC);
    xmlSerializer.text(study.mTopic);
    // close tag: </topic>
    xmlSerializer.endTag("", Study.TOPIC);

    // open tag: <content>
    xmlSerializer.startTag("", Study.CONTENT);
    xmlSerializer.text(study.mContent);
    // close tag: </content>
    xmlSerializer.endTag("", Study.CONTENT);

    // open tag: <author>
    xmlSerializer.startTag("", Study.AUTHOR);
    xmlSerializer.text(study.mAuthor);
    // close tag: </author>
    xmlSerializer.endTag("", Study.AUTHOR);

    // open tag: <date>
    xmlSerializer.startTag("", Study.DATE);
    xmlSerializer.text(study.mDate);
    // close tag: </date>
    xmlSerializer.endTag("", Study.DATE);

    // close tag: </study>
    xmlSerializer.endTag("", Study.STUDY);
    // close tag: </record>
    xmlSerializer.endTag("", Study.RECORD);
```

Generating JSON

- Use JSON class
  - Data
  - Object
  - Array

```
"employees":[
 {"firstName":"John", "lastName":"Doe"},
{"firstName":"Anna", "lastName":"Smith"},
{"firstName":"Peter","lastName":"Jones"}
]
JSONObject main = new JSONObject();
JSONArray arrayEmployee = new JSONArray();
for (int i = 0; i<=4; i++)
{
JSONObject employee = new JSONObject();
//La classe de base
try {
employee.put("firstName","Prenom"+"_"+Integer.toString(i));
employee.put("lasttName","Nom"+"_"+Integer.toString(i));
//ajout de la classe à l'array
arrayEmployee.put(employee);
} catch (JSONException e) {
// TODO Auto-generated catch block
e.printStackTrace();
}
}
main.put("Employees", arrayEmployee);
return main.toString();
}
```

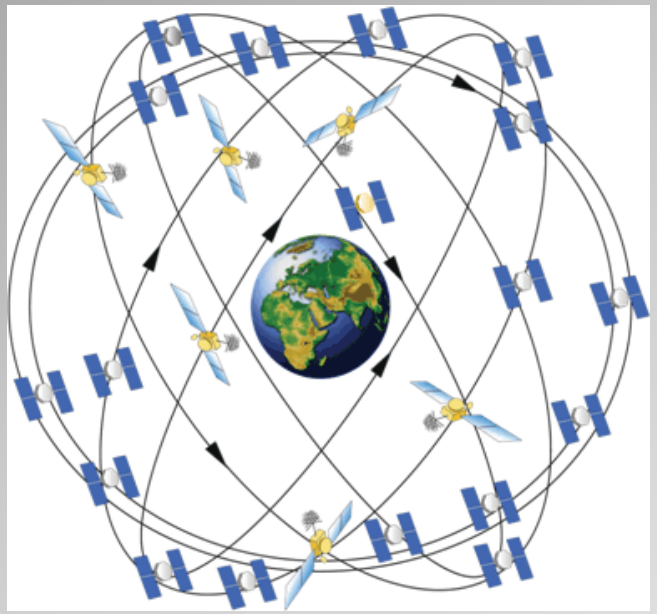
Parsing - XML

- **Dom Parser** - Loads the complete contents of the document and creates its complete hiearchical tree in memory.
- **SAX Parser** - Does not load the complete document into the memory. Parses the document on event based triggers.
- **JDOM Parser** - Parses the document in similar fashion to DOM parser in a easier way
- **XPath Parser** - Parses the XML based on expression.
- **XmlPullParser** recommended by Goggle

ANDROID LOCATION SERVICE

SERVICES DE LOCALISATION

- Rappels
  - Fournisseurs de services de localisation
    - Fournisseur de service GPS



    - 4 satellites visibles sur l'horizon parmi 27 en orbite

SERVICES DE LOCALISATION

- Ephemerides et Almanach utilisés par les recepteurs pour calculer la long/lat
- Obtenir un Almanach est long, facteur limitant l'usage de GPS pour des applications mobiles
  - Contournement
    - » A-GPS (Assisted GPS), Almanach diffué par le réseau terrestre
  - S-GPS (Standard GPS)

– Fournisseur de Network / Réseau

- En Wi-FI
- En utilisant le réseau cellulaire /Cellular Network

SERVICES DE LOCALISATION

- API de Localisation
  - Package android.location
  - Classes:
    - LocationManager : classe donnant l'accès au système de services de localisation
    - LocationProvider : superclass abstraite implémentée par les fournisseurs de services de localisation
    - Location : classe représentant les données de géo localisation
    - Criteria : critères de sélection des fournisseurs
  - Interfaces:
    - LocationListener: Utilisé par les récepteurs pour recevoir les notifications de changement de localisation / position depuis le LocationManager.
- Google location API (recommandé)

LOCATION SERVICE

- Geocoder
  - Geocoding : addresse -> autre format (i.e Lat,Long)
  - Reverse geocoding coord de localisation -> addresse
    - isPresent pour savoir si le backend service existe.

SERVICES DE LOCALISATION

- API à installer
  - Config min : Android > 2.3, AVD > 4.2
  - Dans Android Studio
    - Extras
      - Android Support Repository
      - Android Support Library
    - + Installer
      - Google Repository
      - Google Play Services
  - Récupérer les images Google API pour les émulateurs

SERVICES DE LOCALISATION

- Configuration
  - Modifier build.gradle du projet
    - Insérer
      - Apply plugin: ‘com.android.application’
      - Dependencies {

      Compile ‘com.google.android.gms:play-services:10.0.1’}

      //l’appli est liée à la version référencée
    - Sync le projet avce gradle tools->Android->sync projet avec gradle

Applications

- Mettent en œuvre deux classes
  - LocationServices pour obtenir des coordonnées
  - MapView pour visulaiser des positions sur les cartes

LOCATION SERVICE

- Maps API
  - Mapview : UI to display maps
  - MapActivity : extend this class to use Mapview
  - Overlay : class used to annotate maps
  - MapController : class used to control the map display
  - MyLocationOverlay : layer to display the current position
  - ItemizedOverlay : layer to add drawables to MapViews

LOCATION SERVICE

- Maps API
  - Drawing on a map
  - Markers
  - Window information
  - Shapes
  - Ground overlay (images)
  - StreetView

LOCATION SERVICE

- MapFragment
  - Markers
  - MapType : terrain, hybrid, satellite
  - IndoorMaps
  - Polylines

SERVICE DE LOCALISATION

- Obtention des clès de l'API dans eclipse
- Maps API
  - You need a Key set to use this API
  - Get it from Google : https://console.developers.google.com/project
  - Import Google Play Services into Eclipse
  - Write your application extends mapviewactivity
  - Set-up the manifest with metadata (api key) and use permissions
    - ACCESS_NETWORK_STATE
    - INTERNET
    - WRITE_EXTERNAL_STORAGE
    - ACCESS_COARSE_LOCATION
    - ACCESS_FINE_LOCATION
    - OpenGL ES V2

    - Your terminal (Mobile phone) may need google display services update

SERVICE DE LOCALISATION

- Obtention des clès de l'API Android Studio
- Maps API
  - You need a Key set to use this API
  - Get it from Google : https://console.developers.google.com/project
  - Import Google Play Services into Eclipse
  - Write your application extends mapviewactivity
  - Set-up the manifest with metadata (api key) and use permissions
    - ACCESS_NETWORK_STATE
    - INTERNET
    - WRITE_EXTERNAL_STORAGE
    - ACCESS_COARSE_LOCATION
    - ACCESS_FINE_LOCATION
    - OpenGL ES V2

    - Your terminal (Mobile phone) may need google display services update

LOCATION

– Simulating the location in the emulator

Location Geocoder

- Fournit des adresses à partir de coordonnées (Location)
- Class Geocoder.getFromLocation

FOURNISSEUR DE CONTENU

- Manages access to a central repository of data
- Applications access a provider to handle repository data.
- Applications use a provider client object
- Ex native Android content provider
  - Contact
  - Agenda
- Data is presented to external applications as one or more tables (Relational DB like, row columns)

word	App_id	frequency	locale	_ID
Kodie	user1	100	En_US	1
completude	user2	45	Fr_FR	2

.

CONTENT PROVIDER

- Application get access the data from a provider with a contentresolver client object
- The ContentResolver expose the basic "CRUD" (create, retrieve, update, and delete) methods of persistent storage access.
- Use permission are usually required for the application to use a content provider
  - ex using User Dictionary Provider
    - call ContentResolver.query().
    - Returns a cursor

CONTENT PROVIDER

– Contentresolver.query

- mCursor = getContentResolver().query(
  UserDictionary.Words.CONTENT_URI, // The content URI of the words table
  mProjection, // The columns to return for each row
  mSelectionClause, // Selection criteria
  mSelectionArgs, // Selection criteria
  mSortOrder); // The sort order for the returned rows
- Compared to Sql

query() argument	SELECT keyword/parameter	Notes
Uri	FROM table_name	Uri maps to the table in the provider named table_name.
projection	col,col,col,...	projection is an array of columns that should be included for each row retrieved.
selection	WHERE col = value	selection specifies the criteria for selecting rows.
selectionArgs	(No exact equivalent. Selection arguments replace? placeholders in the selection clause.)	
sortOrder	ORDER BY col,col,...	sortOrder specifies the order in which rows appear in the returned Cursor.

CONTENT PROVIDER

- URI
- content://user_dictionary/words
  - user_dictionary is provider's authority
  - Words is the table to access
  - content:// scheme is allways present , indicates this is a provider
- You append an ID to the URI to get one item
- Otherwise you get the full set of data

- Utils classes to build a URI
  - Uri
  - Uri.Builder
  - ContentUris (to append data to uri)

CONTENT PROVIDER

- To retrieve Data
  - Request read access to provider (use in application manifest)
  - Write code that sens a query to the provider
- Constructing a query
  - ```
    // A "projection" defines the columns that will be returned for each row
    String[] mProjection =
    {
       UserDictionary.Words._ID,    // Contract class constant for the _ID column name
       UserDictionary.Words.WORD,   // Contract class constant for the word column name
       UserDictionary.Words.LOCALE  // Contract class constant for the locale column name
    };
    ```
  - ```
    // Defines a string to contain the selection clause
    String mSelectionClause = null;
    ```
  - ```
    // Initializes an array to contain selection arguments
    String[] mSelectionArgs = {""};
    ```
    ```

Content Provider

- Contacts provider
- Entities

Content Provider

- Create a content provider

Data Persistence

- Shared Preferences
  - Allows to save/read key/values pairs into/from a file
- Content Provider

Telephony

- Terms
  - GSM based on Time Division Multiple Access(TDMA)
  - Use a Suscriber Identity Module (SIM) card to store important user/carrier data
  - *Integrated Circuit Card Identifier (ICCID)*—Identifies a SIM card; also known as a SIM Serial Number, or SSN.
  - *International Mobile Equipment Identity (IMEI)*—Identifies a physical device. ThevIMEI number is usually printed underneath the battery.
  - *International Mobile Subscriber Identity (IMSI)*—Identifies a subscriber (and thevnetwork that subscriber is on).
  - *Location Area Identity (LAI)*—Identifies the region within a provider network that's occupied by the device.
  - *Authentication key (Ki)*—A 128-bit key used to authenticate a SIM card on a provider network.

Telephony

- Terms
  - CDMA other technology used in USA and Asia countries
  - No SIM card
  - Mobile Equipment Identifier (MEID)—Identifies a physical device. It corresponds to GSM's IMEI.
  - Electronic Serial Number (ESN)—The predecessor to the MEID, this number is shorter and identifies a physical device.
  - Pseudo Electronic Serial Number (pESN)—A hardware identifier, derived from the MEID, that's compatible with the older ESN standard. The ESN supply was exhausted several years ago, so pESNs provide a bridge for legacy applications built around ESN. A pESN always starts with 0x80 in hex format or 128 in decimal format.
  - If your application needs telephony the manifest.xml file should include
  - <uses-feature android:name="android.hardware.telephony"="true"/>

Telephony

- TelephonyManager
  - Get telephony information
  - Phone Network state
  - Attach a PhoneStateListener event listener to be aware of state changes
- Retrieve telephony properties
  - TelephonyManager telMgr = (TelephonyManager) getSystemService(Context.TELEPHONY_SERVICE);

```
String callStateString = "NA";
int callState = telMgr.getCallState();
switch (callState) {
case TelephonyManager.CALL_STATE_IDLE:
callStateString = "IDLE";
break;
case TelephonyManager.CALL_STATE_OFFHOOK:
callStateString = "OFFHOOK";
break;
case TelephonyManager.CALL_STATE_RINGING:
callStateString = "RINGING";
break;
}
```

Telephony

```
– CellLocation cellLocation = (CellLocation)telMgr.getCellLocation();
String cellLocationString = null;
if (cellLocation instanceof GsmCellLocation)
{
cellLocationString = ((GsmCellLocation)cellLocation).getLac()
+ " " + ((GsmCellLocation)cellLocation).getCid();
}
else if (cellLocation instanceof CdmaCellLocation)
{
cellLocationString = ((CdmaCellLocation)cellLocation).
getBaseStationLatitude() + " " +
((CdmaCellLocation)cellLocation).getBaseStationLongitude();
}
String deviceId = telMgr.getDeviceId();
String deviceSoftwareVersion =
telMgr.getDeviceSoftwareVersion();
String line1Number = telMgr.getLine1Number();
String networkCountryIso = telMgr.getNetworkCountryIso();
String networkOperator = telMgr.getNetworkOperator();
String networkOperatorName = telMgr.getNetworkOperatorName();
String phoneTypeString = "NA";
int phoneType = telMgr.getPhoneType();
switch (phoneType) {
case TelephonyManager.PHONE_TYPE_GSM:
phoneTypeString = "GSM";
break;
case TelephonyManager.PHONE_TYPE_CDMA:
phoneTypeString = "CDMA";
break;
case TelephonyManager.PHONE_TYPE_NONE:
phoneTypeString = "NONE";
```

Telephony

- Attach listener.

```
final TelephonyManager telMgr =
(TelephonyManager) getSystemService(
Context.TELEPHONY_SERVICE);
PhoneStateListener phoneStateListener =
new PhoneStateListener() {
public void onCallStateChanged(
int state, String incomingNumber) {
telMgrOutput.setText(getTelephonyOverview(telMgr));
}
};
telMgr.listen(phoneStateListener,
PhoneStateListener.LISTEN_CALL_STATE);
String telephonyOverview = getTelephonyOverview(telMgr);
telMgrOutput.setText(telephonyOverview);
```

Telephony

- Interacting with phone
  - Using intents to make calls
    - Use Intent.ACTION_CALL action and the tel: Uri.
    - Use Intent.ACTION_DIAL action

```
dialintent = (Button) findViewById(R.id.dialintent_button);
dialintent.setOnClickListener(new OnClickListener() {
public void onClick(View v) {
Intent intent =
new Intent(Intent.DIAL_ACTION,
Uri.parse("tel:" + NUMBER));
startActivity(intent);
}
});
```

Telephony

- Interacting with phone
- Android permissions

android.permission.CALL_PHONE
Initiates a phone call without user confirmation in dialer
android.permission.CALL_PRIVILEGED
Calls any number, including emergency, without confirmation in dialer
android.permission.MODIFY_PHONE_STATE
Allows the application to modify the phone state: for example, to turn the radio on or off
android.permission.PROCESS_OUTGOING_CALLS
Allows the application to receive broadcast for outgoing calls and modify
android.permission.READ_PHONE_STATE
Allows the application to read the phone

Telephony

- Interacting with phone
  - To parse numbers
    - Use PhoneNumberUtils class
  - Intercepting outbounds calls

```
public class OutgoingCallReceiver extends BroadcastReceiver {
public static final String ABORT_PHONE_NUMBER = "1231231234";
@Override
public void onReceive(Context context, Intent intent) {
if (intent.getAction().equals(
Intent.ACTION_NEW_OUTGOING_CALL)) {
String phoneNumber =
intent.getExtras().getString(Intent.EXTRA_PHONE_NUMBER);
if ((phoneNumber != null)
&& phoneNumber.equals(
OutgoingCallReceiver.ABORT_PHONE_NUMBER)) {
Toast.makeText(context,
"NEW_OUTGOING_CALL intercepted to number "
+ "123-123-1234 - aborting call",
Toast.LENGTH_LONG).show();
abortBroadcast();
}
}
}
}
```

Telephony

- Interacting with phone
  - Working with SMS (Short Message Service)
  - Receiving SMS

```
public class SmsReceiver extends BroadcastReceiver {
private static final String SMS_REC_ACTION =
"android.provider.Telephony.SMS_RECEIVED";
@Override
public void onReceive(Context context, Intent intent) {
if (intent.getAction().
equals(SmsReceiver.SMS_REC_ACTION)) {
StringBuilder sb = new StringBuilder();
Bundle bundle = intent.getExtras();
if (bundle != null) {
Object[] pdus = (Object[])
bundle.get("pdus");
for (Object pdu : pdus) {
SmsMessage smsMessage =
SmsMessage.createFromPdu
((byte[]) pdu);
sb.append("body - " + smsMessage.
getDisplayMessageBody());
}
}
Toast.makeText(context, "SMS RECEIVED - "
+ sb.toString(), Toast.LENGTH_LONG).show();
}
}
}
```

Telephony

- Interacting with phone
  - Working with SMS (Short Message Service)
  - Sending SMS

```
private Button smsSend;
private SmsManager smsManager;
@Override
public void onCreate(Bundle icicle) {
super.onCreate(icicle);
setContentView(R.layout.smsexample);
// . . . other onCreate view item inflation omitted for brevity
smsSend = (Button) findViewById(R.id.smssend_button);
smsManager = SmsManager.getDefault();
final PendingIntent sentIntent =
PendingIntent.getActivity(
this, 0, new Intent(this,
SmsSendCheck.class), 0);
smsSend.setOnClickListener(new OnClickListener() {
public void onClick(View v) {
String dest = smsInputDest.getText().toString();
if (PhoneNumberUtils.
isWellFormedSmsAddress(dest)) {
smsManager.sendTextMessage(
smsInputDest.getText().toString, null,
smsInputText.getText().toString(),
sentIntent, null);
Toast.makeText(SmsExample.this,
"SMS message sent",
Toast.LENGTH_LONG).show();
} else {
Toast.makeText(SmsExample.this,
"SMS destination invalid - try again",
Toast.LENGTH_LONG).show();
}
}
});
}
```

Ermissions

```
<uses-permission android:name="android.permission.RECEIVE_SMS" />
<uses-permission android:name="android.permission.READ_SMS" />
<uses-permission android:name="android.permission.WRITE_SMS" />
<uses-permission android:name="android.permission.SEND_SMS" />
```

Notifications and alarms

- Alarm
  - Simple mecanism for time based operations
    - Fire intents
    - Start services … using broadcast receivers
    - Works outside of your application, when device is asleep
    - Inside your application use a timer and handler
    - Alarm specifics
      - Alarm Type: elapsed time or realtime
      - Trigger time
      - Alarm interval
      - Pending intent

Notifications et alarmes

- Notification
  - visible dans la toolbar de l'application
  - Champs d'une notification
    - obligatoire
      - titre
      - icone
      - message
    - Optionnel
      - action
      - ….
  - Généralement une notification déclenche une activité
- Pattern d'utlisation
  - Ask for a notificationManager (getSystemService)
  - instanciate NotificationCompat.Builder to build the notification
  - notificationManager.notify to send the notification

Cle API

- Cle debug
- Récupérer le md5 d'éclipse
- Aller sur google console
- Obtenir la cle API
- Importer google play dans le workspace
- Creer l'appli
- Mettre à jour le manifeste de l'appli avec la cle API
  - <!-- Goolge Maps API Key -->
  - <meta-data

android:name="com.google.android.maps.v2.API_KEY"
android:value="AIzaSyBZMlkOv4sj-M5JO9p6wksdax4TEjDVLgo" />

Cle API

- Dans Android studio
- Récupérer le fichier google_maps_api.xml dans res
  - \Projet\app\src\release\res\values\google_maps_api.xm
- Récupérer le path et se connecter
  - Vous obtenez une clé spécifique au package de votre application
  - Ex:
  - AIzaSyCBTbop8Ms_HjsGcs0avSVG4oEwBSP7esI
- Mettre à jour la clé dans le fichier google_maps_api_xml
  - **<string name="google_maps_key" translatable="false" templateMergeStrategy="preserve">**
    AIzaSyCBTbop8Ms_HjsGcs0avSVG4oEwBSP7esI
    </string>

URI

- URI.builder
- Construire
  - http://lapi.transitchicago.com/api/1.0/ttarrivals.aspx?key=[redacted]&mapid=`value`.
  - Uri.Builder builder = new Uri.Builder();
    builder.scheme("http")
    .authority("www.lapi.transitchicago.com")
    .appendPath("api") .appendPath("1.0")
    .appendPath("ttarrivals.aspx")
    .appendQueryParameter("key", "[redacted]")
    .appendQueryParameter("mapid", value);

Annexes

SUPPORT D'ECRANS MULTIPLES

- Comment définir les layouts dépendants de la taille de l'écran
- Les qualificateurs de groupe de taille d'écran sont abandonnés. Small, x-large, xx-large …
  - Qualificateurs de taille d'écrans
  - sw<n> :
    - smallest width. Ne change pas avec l'orientation de l'écran. Ex sw600dp, res/layout-sw750dp/
  - w<n>:
    - Width. Min width.Change avec l'orientaion de l'écran en passant du mode portrait à landscape. Ex res/layout-w1024dp/
  - h<n> :
    - Height.Min height. Change avec l'orientation de l'écran.
- Ex
  - res/layout/main_activity.xml # For handsets (smaller than 600dp available width)
    res/layout-sw600dp/main_activity.xml # For 7" tablets (600dp wide and bigger)
    res/layout-sw720dp/main_activity.xml # For 10" tablets (720dp wide and bigger)
- Le manifeste doit être mis à jour en cohétence
  - <manifest ... >
    <supports-screens android:requiresSmallestWidthDp="600" />
    ...
    </manifest>

CAPTEURS

- Dépend de la confirgde votre device
  - Types:
    - TYPE_ACCELEROMETER
    - TYPE_AMBIENT_TEMPERATURE
    - TYPE_GRAVITY
    - TYPE_GYROSCOPE
    - TYPE_LIGHT
    - TYPE_LINEAR_ACCELERATION
    - TYPE_MAGNETIC_FIELD
    - TYPE_ORIENTATION
    - TYPE_PRESSURE
    - TYPE_PROXIMITY
    - TYPE_ROTATION_VECTOR
    - TYPE_TEMPERATURE

CAPTEURS

- Gestion
  - Utiliser le sensor framework package android.hardware
  - Classes
    - sensorManager
    - Sensor
    - SensorEvent
    - SensorListener

CAPTEURS

- Identifier les capteurs de votre device

```
private SensorManager mSensorManager;
   ...
   mSensorManager = (SensorManager)
   getSystemService(Context.SENSOR_SERVICE);
//OBTENIR LA LISTE
List<Sensor> deviceSensors =
   mSensorManager.getSensorList(Sensor.TYPE_ALL);
```

CAPTEURS

- Identifier les capteurs de votre device

```
private SensorManager mSensorManager;
        private Sensor mSensor;

        ...

        mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);

        if (mSensorManager.getDefaultSensor(Sensor.TYPE_GRAVITY) != null){
          List<Sensor> gravSensors = mSensorManager.getSensorList(Sensor.TYPE_GRAVITY);
          for(int i=0; i<gravSensors.size(); i++) {
            if ((gravSensors.get(i).getVendor().contains("Google Inc.")) &&
               (gravSensors.get(i).getVersion() == 3)){
              // Use the version 3 gravity sensor.
              mSensor = gravSensors.get(i);
            }
          }
        }
        else{
          // Use the accelerometer.
          if (mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER) != null){
            mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER);
          }
          else{
            // Sorry, there are no accelerometers on your device.
            // You can't play this game.
          }
        }
```

CAPTEURS

- Evénements
  - Callback
    - onAccuracyChanged
    - onSensorChanged

CAPTEURS

- Evénements

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

        @Override
        public final void onCreate(Bundle savedInstanceState) {
          super.onCreate(savedInstanceState);
          setContentView(R.layout.main);

          mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
          mLight = mSensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
        }

        @Override
        public final void onAccuracyChanged(Sensor sensor, int accuracy) {
          // Do something here if sensor accuracy changes.
        }

        @Override
        public final void onSensorChanged(SensorEvent event) {
          // The light sensor returns a single value.
          // Many sensors return 3 values, one for each axis.
          float lux = event.values[0];
          // Do something with this sensor value.
        }

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

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

Connectivité BlueTooth

- Discovery
- Pairing

Permissions

- API 23 +
  - Les permissions sont validées au run et plus seulement au chargement d'un APK
  - Normal et dangerous permissions
  - Les permissions doivent toujours être déclarées dans le manifeste et les « dangerous »
  - Vous pouvez
    - Vérifier les permissions
      - int permissionCheck = ContextCompat.checkSelfPermission(thisActivity, Manifest.permission.WRITE_CALENDAR);
    - Demander des permissions
    - Refuser des permissions
    - Ne pas être à nouveau sollicité

Permissions

- https://material.io/guidelines/patterns/permissions.html#

- https://inthecheesefactory.com/blog/things-you-need-to-know-about-android-m-permission-developer-edition/en
- android.permission.ACCESS_LOCATION_EXTRA_COMMANDS
  android.permission.ACCESS_NETWORK_STATE
  android.permission.ACCESS_NOTIFICATION_POLICY
  android.permission.ACCESS_WIFI_STATE
  android.permission.ACCESS_WIMAX_STATE
  android.permission.BLUETOOTH
  android.permission.BLUETOOTH_ADMIN
  android.permission.BROADCAST_STICKY
  android.permission.CHANGE_NETWORK_STATE
  android.permission.CHANGE_WIFI_MULTICAST_STATE
  android.permission.CHANGE_WIFI_STATE
  android.permission.CHANGE_WIMAX_STATE
  android.permission.DISABLE_KEYGUARD
  android.permission.EXPAND_STATUS_BAR
  android.permission.FLASHLIGHT
  ~~android.permission.GET_ACCOUNTS~~
  android.permission.GET_PACKAGE_SIZE
  android.permission.INTERNET
  android.permission.KILL_BACKGROUND_PROCESSES
  android.permission.MODIFY_AUDIO_SETTINGS
  android.permission.NFC
  android.permission.READ_SYNC_SETTINGS
  android.permission.READ_SYNC_STATS
  android.permission.RECEIVE_BOOT_COMPLETED
  android.permission.REORDER_TASKS
  android.permission.REQUEST_INSTALL_PACKAGES
  android.permission.SET_TIME_ZONE
  android.permission.SET_WALLPAPER
  android.permission.SET_WALLPAPER_HINTS
  android.permission.SUBSCRIBED_FEEDS_READ
  android.permission.TRANSMIT_IR
  android.permission.USE_FINGERPRINT
  android.permission.VIBRATE
  android.permission.WAKE_LOCK
  android.permission.WRITE_SYNC_SETTINGS
  com.android.alarm.permission.SET_ALARM
  com.android.launcher.permission.INSTALL_SHORTCUT
  com.android.launcher.permission.UNINSTALL_SHORTCUT