Ressources, outils et astuces

- Ressources
  - Statistiques sur les terminaux
  - Reconnaissance du terminal / bases de données des terminaux
    - Exemple d'utilisation "basique" de WURFL 1.0.1 :
  - Tests et Emulateurs
  - Bibliothèques de développement pour mobile
  - Documents de référence
- Outils et astuces
  - Géolocalisation
  - Auto configuration par le web
  - Guide pour le développement rapide d'une application pour mobiles : JSF (Trinidad) - Spring

Ressources

Statistiques sur les terminaux

Ici, exemple pour les navigateurs en France sur un an depuis avril 2010 : http://gs.statcounter.com/#mobile_browser-FR-monthly-201004-201104

On peut aussi avoir des stats sur les terminaux eux mêmes, sur des périodes ou des régions différentes.

Reconnaissance du terminal / bases de données des terminaux

UAProf, base de l'Open Mobile Alliance : http://www.openmobilealliance.org/Technical/schemas.aspx

Liste des terminaux : http://deviceatlas.com/devices

WURFL, base "collaborative", avec API : http://wurfl.sourceforge.net/

Exemple d'utilisation "basique" de WURFL 1.0.1 :

WURFL est maintenant en version 1.2, avec notamment une utilisation via Spring prévue : http://wurfl.sourceforge.net/njava/

En pièce jointe, la classe de gestion du WURFL : MyWurflService.java

Lors de l'initialisation de l'application :

```
    // Initialisation du service WURFL
    this.myWurflService = new MyWurflService();
```

Dans mon wrapper, à la création, pour savoir si l'utilisateur a un terminal mobile :

```
this.isMobile = Application.getAppli().getMyWurflService().isMobileDevice(this.sess, context.request().
toString());
```

On peut avoir des infos sur le terminal :

```
Device device = Application.getAppli().getMyWurflService().getWurflHolder().getWURFLManager().
getDeviceForRequest(request.toString());
String browser = device.getCapability(MyWurflService.MOBILE_BROWSER);
```

Pour nettoyer le fichier WURFL et n'avoir que les propriétés voulues :
http://www.tera-wurfl.com/wiki/index.php/WURFL_Customizer

Le filtre utilisé :

```
public static $CAPABILITY_FILTER = array(
    "mobile_browser",
    "device_os",
    "mobile_browser_version",
    "model_name",
    "device_os_version",
    "is_wireless_device",
    "device_claims_web_support",
    "brand_name",
);
```

On passe de 15 à 3Mo en utilisant le filtre.

Tests et Emulateurs

W3C mobileOK : http://validator.w3.org/mobile

Bibliothèques de développement pour mobile

iUI (google code)
JSF Trinidad (cf docs de YD)

Titanium Appcelerator (utilisé par Jasig) : http://www.appcelerator.com/products/titanium-mobile-application-development/

Documents de référence

Mobile Web Application Best Practices

Outils et astuces

Géolocalisation

HTML5 a une API qui peut permettre de géolocaliser si le terminal est capable de le faire : http://dev.w3.org/geo/api/spec-source.html

Voir par exemple http://www.paperblog.fr/2550120/geolocalisation-de-vos-visiteurs-en-javascript-grace-a-html-5/

Il y a plusieurs moyens pour géolocaliser : par IP, par GPS, par utilisation de la géolocalisation des points d'accès WiFi détectés à proximité.... Hormis le GPS, ça passe toujours par l'utilisation de base de données d'IP ou de bornes WiFi, récoltées de manière plus ou moins légale...

(rem : Gears, c'est fini : ~~http://code.google.com/intl/fr/apis/gears/api_geolocation.html~~ http://gearsblog.blogspot.com/2011/03/stopping-gears.html)

Auto configuration par le web

http://pau.edu.tr/eduroam/sayfa2844.aspx

http://www.apple.com/DTDs/PropertyList-1.0.dtd

http://www.iphone-notes.de/mobileconfig/

Guide pour le développement rapide d'une application pour mobiles : JSF (Trinidad) - Spring

Voir le Guide WTP JSF