Cibler plusieurs OS avec Titanium : 'best practices'

Meetup Paris Titanium n° 1 – Jeudi 24 mai 2012

Préambule

- Contrairement à des web apps développées en HTML5 et CSS3, Titanium préserve le 'look and feel' de la plate-forme sous-jacente
  - Titanium ce n'est pas : 'Write once, run everywhere'
  - Titanium c'est : 'Write once, adapt everywhere'
  - Dans ces conditions, il est inévitable qu'une partie du code soit spécifique à chaque plate-forme
- À ce jour, Appcelerator n'a pas de 'best practices' complètes et formalisées relatives au développement d'une app ciblant plusieurs OS
- Des développements sont en cours, aussi bien chez Appcelerator (cf. par exemple les nouvelles règles de 'layout') que chez les développeurs
- Taux de réutilisation du code selon Appcelerator :
  - Près de 100% du code non lié à l'interface utilisateur (il ne faut pas oublier que même hors interface utilisateur il existe des spécificités à chaque OS)
  - 80% à 100% du code de l'interface utilisateur (100% pour une app 'immersive', par exemple un jeu)

Définition du problème

- Objectifs
  - Écrire une app ciblant plusieurs OS (iOS et Android par exemple) ou matériels (smartphones et tablettes par exemple) en écrivant un minimum de code
  - Faire en sorte que l'app résultante soit facile à maintenir :
    - Évolutions fonctionnelles
    - Évolutions liées à des évolutions de Titanium
    - Évolutions liées à des évolutions de l'OS sous-jacent

Fonctionnalités de Titanium permettant de cibler plusieurs OS

- Identification de la plate-forme sur laquelle le code s'exécute
- APIs, 'properties' et constantes spécifiques à une plate-forme
- Ressources spécifiques à une plate-forme

Identification de la plate-forme et branchements ad-hoc

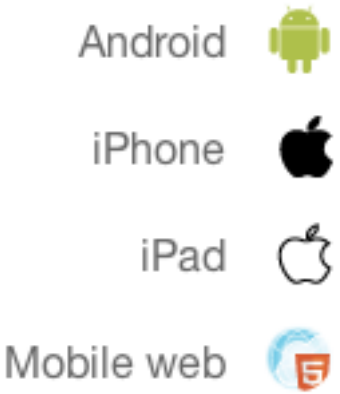
Propriété	Description	Exemple de valeurs
Ti.Platform.name	Retourne le nom de la plate-forme	iPhone OS, Android
Ti.Platform.osname	Retourne un identifiant simplifié de la plate-forme	iphone, ipad, android, mobileweb
Ti.Platform.model	Retourne un identifiant du matériel	iPhone 3GS, iPod Touch 2G, Droid

```
// create a JavaScript alias to the platform-specific property
var osname = Ti.Platform.osname;
// Booleans identifying the platforms are handy too
var isAndroid = (osname === 'android') ? true : false;

if(isAndroid) {
     // do Android specific stuff
} else {
     // do iOS, mobileweb, or other platform stuff
}
```

APIs et propriétés spécifiques à une plate-forme

- Les APIs spécifiques à une plate-forme sont regroupées dans des 'namespaces' dédiés
  - `Ti.UI.iOS[.Toolbar]`
  - `Ti.UI.iPhone[.AnimationStyle]`
  - `Ti.UI.iPad[.SplitWindow]`
  - `Ti.UI.Android[.hideSoftKeyboard()]`
- Certains objets ont des propriétés spécifiques à une ou plusieurs plates-formes
  - Exemple : la propriété `softInputMode` de `Ti.UI.Window` ne s'applique qu'à Android
- Certaines constantes sont spécifiques à une ou plusieurs plates-formes
  - Exemple : la constante `Ti.UI.iOS.ANIMATION_CURVE_EASE_IN` ne s'applique qu'à iOS



Ressources spécifiques à une plate-forme (1/3)

- Ressources concernées images, stylesheets, scripts…
- Règle de base
  - Tout ressource stockée dans un répertoire de spécifique à une plate-forme (exemple : 'Resources/android/img', 'Resources/iphone/img', 'Resources/mobileweb/img') sera traité à la place d'une ressource équivalente stockée dans le répertoire 'Resources/img'

```
var img = Ti.UI.createImageView({
     image: 'logo.png'
     /* 'Resources/android/logo.png' ou 'Resources/iphone/logo.png'
     * ou 'Resources/mobileweb/logo.png' seront utilisés
     * automatiquement s'ils existent lors de la compilation.
     * Sinon 'Resources/logo.png' sera utilisé.
     */
});
```

Ressources spécifiques à une plate-forme (2/3)

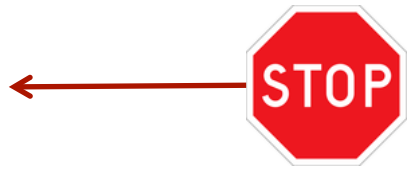
Ressources spécifiques à une plate-forme (3/3)

- Fichiers JSS (≈ CSS)

```
ui.js
var appceleratorLabel = Ti.UI.createLabel({
     text:'Appcelerator',
     id:'appceleratorLabel'
});

ui.jss
#appceleratorLabel {
     width:149;
     text-align:'center';
     color:'#999';
     font-size:'13';
}
```

- Titanium accepte des fichiers spécifiques à une plate-forme
  - Exemples : `ui.iphone.jss` , `ui.android.jss`
  - Attention: `ui.jss` ,est prioritaire sur `ui.iphone.jss` et `ui.android.jss`



Les conseils d'Appcelerator pour cibler plusieurs OS avec Titanium

- Branchements ad-hoc
  - OK s'il y a très peu de code spécifique
- Fichiers JS spécifiques à chaque plate-forme
  - OK si y a beaucoup de code spécifique
  - Limite le nombre de blocs `if … then` dans le code
  - Par contre certaines évolutions devront être répercutées sur toutes les plates-formes, accroissant alors la charge de travail
- Pour des projets relativement simples, et n'utilisant pas CommonJS (`JSS` et `require()` ne fonctionnent pas bien ensemble actuellement)
  - Possibilité d'utiliser des fichiers `JSS` pour créer des 'layouts' spécifiques à chaque plate-forme
- Pour des projets plus complexes
  - Possibilité d'utiliser des fichiers JS intégrant les spécificités de chaque plate-forme lors de la définition des composants
  - Utiliser la technique Tweetanium (objet 'style' global avec branchements selon plate-forme)

Conseil additionnel : utiliser le modèle MVC