Universal Widget API : concevoir des widgets multiplateformes

Xavier Borderie / netvibes
http://dev.netvibes.com
ParisWeb 2007 workshop
Samedi 17 novembre

Ce que sont les widgets

- Petites applications spécialisées (ou pas)
- Disponibles via le navigateur : Netvibes, iGoogle, Live.com, YourMinis...
- Disponible via un moteur de widget intégré : Vista Gadgets, Apple Dashboard, Yahoo! Widgets, Opera...

Le constat de Netvibes

- MiniAPI fonctionne bien
  - 1000 modules depuis mai 2006
  - Bon retour de la communauté
- Pas de standard de fait
  - Chaque site/moteur utilise son propre format
  - La specification du W3C est toujours en Working Draft

Ce que promet UWA

- Fonctionnement sur les plus grandes plates-formes, sans modification du fichier original
  - Netvibes, iGoogle, Live.com, Opera, Apple Dashboard, Windows Vista, Yahoo! Widgets
- Un fonctionnement proche de MiniAPI, en plus strict
- Une facilité d'apprentissage grâce aux standards : XHTML/XML, JavaScript/Ajax, CSS

Les bases d'UWA

- Un fichier statique XHTML, respectueux de la syntaxe XML
- Encodage UTF-8
- Espace de nom Netvibes obligatoire : xmlns:widget="http://www.netvibes.com/ns/"

Présentation du gabarit de base

- http://dev.netvibes.com/files/uwa-skeleton.html
- Point de départ pour la plupart des développeurs
- Générateur en version test

Gabarit 1 : les en-têtes XHTML

- Tout ce qu'il y a de plus classique
- Ne pas oublier le namespace...

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
  xmlns:widget="http://www.netvibes.com/ns/"  >
  <head>

    <title>Title of the Widget</title>
    <link rel="icon" type="image/png"
      href="http://www.netvibes.com/favicon.ico" />
```

Gabarit II : les balises meta

- Usages variés
- La plupart optionnelles

```
<meta name="author" content="John Doe" />
<meta name="website" content="http://exemple.org" />
<meta name="description" content="A descriptive description" />
<meta name="keywords" content="these, are, key words" />
<meta name="screenshot" content="http://exemple.org/widget-full.png" />
<meta name="thumbnail" content="http://exemple.org/widget-logo.png" />

<meta name="apiVersion" content="1.0" />
<meta name="autoRefresh" content="20" />
<meta name="debugMode" content="true" />
```

Gabarit III : les fichiers d'émulation

- Optionnels mais très pratique pour le test en mode Standalone

```
<link rel="stylesheet" type="text/css"
  href="http://www.netvibes.com/themes/uwa/style.css" />
<script type="text/javascript"
  src="http://www.netvibes.com/js/UWA/load.js.php?
env=Standalone"></script>
```

Gabarit IV : les préférences UWA

- Jeu de balises spécifiques à UWA
- Casse la validation XHTML
- Nécessaire pour des préférences portables

```
<widget:preferences>
  <preference name="url" type="text" label="URL"
    defaultValue="http://feeds.feedburner.com/NetvibesDevBlog" />
  <preference name="pass" type="password" label="Password" />
  <preference name="displayImages" type="boolean" label="Display images?"
    defaultValue="true" />
  <preference name="limit" type="range" label="Number of items to display"
    defaultValue="10" step="1" min="1" max="25" />
  <preference name="category" type="list" label="Category"
    defaultValue="1st" onchange="refresh">
    <option value="all" label="all" />
    <option value="1st" label="First category" />
    <option value="2nd" label="Second category" />
    <option value="3rd" label="Third category" />
  </preference>
  <preference name="search" type="hidden" defaultValue="" />
</widget:preferences>
```

```
<widget:preferences>
  <preference name="url" type="text" label="URL"
    defaultValue="http://feeds.feedburner.com/NetvibesDevBlog" />

  <preference name="password" type="password" label="Password" />

  <preference name="displayImages" type="boolean"
    label="Display images?" defaultValue="true" />

  <preference name="limit" type="range"
    label="Number of items to display"
    defaultValue="10" step="1" min="1" max="25" />

  <preference name="category" type="list" label="Category"
    defaultValue="1st" onchange="refresh">
    <option value="all" label="all" />
    <option value="1st" label="First category" />
    <option value="2nd" label="Second category" />
    <option value="3rd" label="Third category" />
  </preference>

  <preference name="search" type="hidden" defaultValue="" />
</widget:preferences>
```

Gabarit V : code CSS et JavaScript

- Tout simplement dans les balises conçues à cet effet...

```
<style type="text/css">
 /* your CSS rules */
</style>

<script type="text/javascript">
  var YourWidgetName = {};

  YourWidgetName.data = null;

  YourWidgetName.display = function(responseJson) {
    // display code
  }

  widget.onLoad = function() {
    UWA.Data.getFeed(widget.getValue('url'), YourWidgetName.display);
  }

  widget.onRefresh = widget.onLoad;
  widget.refresh = widget.onLoad;
</script>
```

```
<style type="text/css">
  /* your CSS rules */
</style>

<script type="text/javascript">
  var YourWidgetName = {};

  YourWidgetName.data = null;

  YourWidgetName.display = function(responseJson) {
    // display code
    }

  widget.onLoad = function() {
    UWA.Data.getFeed(widget.getValue('url'),
YourWidgetName.display);
    }

  widget.onRefresh = widget.onLoad;
  widget.refresh = widget.onLoad;
</script>
```

Gabarit VI : fin du fichier

- Le corps peut être vide ou pré-rempli - son contenu est libre car modifiable à tout moment
- Les données sont chargées via Ajax et placées via le DOM

```
  </head>
  <body>
    <p>Loading...</p>
  </body>
</html>
```

On va p’tet passer à la pratique, non ?

Utiliser CSS et JavaScript

- Comme dans un fichier HTML classique : <script> et <style>
- Evitez de faire appel à des fichiers externes : mettez tout votre code dans le widget
- CSS : préfixer chaque règle d'une classe au nom du widget, `.monWidget p { ... }`
- CSS : cibler avec des classes plutôt que des id
- JS : placer chaque méthode/valeur dans un objet au nom du widget, `var MonWidget = {};`

Exemples pratiques :
fliptext
Fireplace

Fliptext

- http://nvmodules.typhon.net/maurice/fliptext/
- Fait uniquement avec HTML, CSS et JS - aucun appel extèrieur
- Adaptation dans UWA du code JavaScript original : http://www.fliptext.org/

```
widget.onLoad = function() {
  for (i in Flip.table) {
    Flip.table[Flip.table[i]] = i
  }

  out = '<textarea></textarea><p><button>flip <img
    src="http://nvmodules.typhon.net/maurice/fliptext/refresh.png"
    alt="Flip" /> dılɟ</button></p><textarea></textarea>';
  widget.setBody(out);

  var bt = widget.body.getElementsByTagName('button')[0];
  var textareas = widget.body.getElementsByTagName('textarea');
  bt.onclick = function(){
    var result = Flip.flipString(textareas[0].value.toLowerCase());
    textareas[1].value = result;
  }
}
```

Fireplace

- http://nvmodules.typhon.net/maurice/fireplace/index.html
- Démonstration de l'intégration de Flash
- Le widget génère le code HTML avec JavaScript, mais il est possible de placer directement la balise <object> dans le corps, sans jamais faire appel à JavaScript

```
widget.onLoad = function() {
    var contentHtml = '';

    contentHtml += '<div style="margin: 0 auto;text-
align:center;">';

    contentHtml += '<object type="application/x-shockwave-flash"
data="http://nvmodules.typhon.net/maurice/fireplace/fire.swf"
width="320" height="240" class="flash">';

    contentHtml += '<param name="movie" value="http://
nvmodules.typhon.net/maurice/fireplace/fire.swf" />';
    //contentHtml += '<param name="wmode" value="opaque" />';

    contentHtml += '<embed src="http://nvmodules.typhon.net/
maurice/fireplace/fire.swf" type="application/x-shockwave-flash"
width="320" height="240"></embed>';
    contentHtml += '</object>';

    contentHtml += '</div>';

    widget.setBody(contentHtml);
    widget.onResize();
}
```

Exercice pratique

Créer un widget Zorglangue

- http://ndiremdjian.free.fr/pics/zorglangue.htm
- Mêmes CSS et `widget.onLoad()` que Fliptext
- Code JavaScript placé dans un object `Zorglub`

Le widget Zorglangue

- http://nvmodules.typhon.net/maurice/zorglub/

Méthodes et propriétés JavaScript de UWA

- Eviter `document` et `window`
- Remplacer `document.getElementById('id')` par `widget.body.getElementsByClassName('classe')[0]`
- Elements étendus seulement si créés par `widget.createElement()`. L'extension peut se faire à la main :

```
var foo = UWA.$element
(widget.body.getElementsByClassName
('bar')[0];
foo.setStyle('backgroundColor', 'red');
```

Méthodes et propriétés JavaScript de UWA

- En remplacement de `document` et `window`, deux objets spécifiques à UWA : `widget` et `UWA`.
  - `widget` : méthodes habituelles des *frameworks* JavaScript
  - `UWA` : surtout pour `UWA.Data`, les méthodes Ajax
- Voir la *cheat-sheet* :)

Méthodes Ajax

- 4 méthodes “rapides” :
  - `UWA.Data.getText(url, callback);`
  - `UWA.Data.getXml(url, callback);`
  - `UWA.Data.getJson(url, callback);`
  - `UWA.Data.getFeed(url, callback);`
- Toutes reposent sur une méthode-mère :
  - `UWA.data.request(url, request);`

UWA.Data.request

- Autorise les requêtes plus complexes : `POST` + paramètres, authentification, gestion du cache serveur...
- `UWA.Data.request(url, request)` :
  - `request = { method:'post', proxy:'ajax', cache:3600, parameters:'lastname=Bond&id=007', onComplete:MonWidget.display };`

Exemples pratiques :

récupérer les images d'un flux : FFFFOUND

exploiter RSS/Atom : Skyblog

getText sur parsing : DeMets

getText sur parsing : LinkedIn

Récupérer les images d'un flux : FFFFOUND

- http://nvmodules.typhon.net/maurice/uwa-ffffound/
- JS : récupération du flux avec getFeed(), extraction du lien des images (<link> du flux JSON), génération à la volée du code
- CSS : placement des éléments
- JSON Feed Format : http://dev.netvibes.com/doc/uwa_specification/uwa_json_feed_format

Exploiter RSS/Atom : Skyblog

- http://nvmodules.typhon.net/maurice/skyblog/
- Préférences : nom du blog et nombre d'articles à afficher
- JS : parcours du flux JSON et construction du HTML avec le DOM plutôt que directement dans une chaîne
- A noter : utilisation de la préférence `limit`
- A noter : `UWA.Utils.setTooltip()`

getText sur parsing : DeMets

- http://nvmodules.typhon.net/maurice/uwa-demets/
- getText sur un script PHP perso, qui se charge de nettoyer le plus gros de la page
- quelques regexp pour retirer encore plus de contenu
- récupération et affichage du code de la carte

getText sur parsing : LinkedIn

- http://florent.solt.googlepages.com/linkedin-pretty.html
- getText direct sur la page à parcourir
- Affichage direct de la section choisie, avec un peu de RegExp pour simplifier/remanier le contenu

Modèles et contrôleurs

- Pour une meilleure intégration au thème Netvibes
- Pour faciliter la conception de certaines applications

Modèles

- Des classes CSS
  - Data grid
  - E-mail list
  - Feed list
  - Rich list
  - Thumbnailed list

Contrôleurs

- Classes CSS + comportements JavaScript
  - TabView
  - Pager
  - Navigation Highlight

Exemples pratiques :

getFeed et FeedList : RSSReader
JSON request et Pager : Twitter
getJson et TabView : TwitterKing
parsing getText et TabView+Pager+... :
Rugby World Cup

RSS Reader

- http://www.netvibes.com/api/uwa/examples/rssreader.html
- getFeed transcrit n'importe quel type de flux en un format JSON interne normalisé
- http://dev.netvibes.com/doc/uwa_specification/uwa_json_feed_format
- Partant de là, récupérer les informations est une simple question de connaître le format en question

Twitter

- http://dev.netvibes.com/files/examples/twitter-05-auth-getpost.html
- UWA.Data.request sur un flux JSON authentifié
- Composition du HTML avec le DOM et les méthodes UWA (setHTML, etc.)

TwitterKing

- http://www.my-widget.com/twitter_widget.html
- Comme Twitter, mais en mode ++

Rugby World Cup

- http://nvmodules.typhon.net/romain/rugbyworldcup/
- getText direct sur une page tierce
- Parcours du code, RegExp, recomposition des liens originaux, Pager, TabView... la totale

Merci !

- http://dev.netvibes.com
- http://dev.netvibes.com/doc/
- http://dev.netvibes.com/forum
- http://dev.netvibes.com/blog/
- http://eco.netvibes.com
- On embauche ! :)
  http://dev.netvibes.com/doc/jobs