Enyo

Enyo est un framework open-source qui utilise JavaScript et HTML. Il est cross-plateforme, en plus d'être un framework mobile et tablette il permet de créer des applications pour téléviseur.

Installation

Il y a deux solutions pour installer Enyo :

- La première consiste à télécharger Node JS et NPM et à installer Enyo via NPM avec la commande suivante : npm install -g enyo-dev
- La seconde consiste à télécharger directement les sources se trouvant sur GitHub à l'adresse suivante : git@github.com:enyojs/enyo.git

Notre solution fut de télécharger les sources de Bootplate sur GitHub et de travailler dessus. Bootplate fournit un projet de démarrage complet et facilite la création d'applications cross-plateformes. Pour télécharger Bootplate :

git clone --recursive --branch 2.4.0 https://github.com/enyojs/bootplate-moonstone.git <projet>

Ces informations peuvent être trouvées sur le site web d'Enyo :http://enyojs.com/get-enyo/

Création d’une calculatrice

Nous allons maintenant vous présenter un tutoriel permettant la création d’une calculatrice simple.

Pour initialiser un projet, il suffit d’utiliser la commande : enyo init nom-du-projet

Si Enyo a été installé en téléchargeant les sources présentes sur Github le projet est déjà initialisé.

Il suffit, à présent, de se déplacer dans le dossier du projet. L’arborescence du projet dépendra de la méthode d’installation et de la présence ou non de plugins mais dans tous les cas les dossiers et fichiers suivants devraient être visibles :

.enyoconfig	// fichier de configuration pour Enyo
.gitignore	// fichier de configuration pour Github
index.js	// fichier source principal
lib	// Enyo et ses bibliothèques
package.json	// fichier de configuration de l’application

Maintenant, s’ils n’existent pas encore, il faudra créer les dossiers "assets" qui contiendra les ressources de l’application et "src" qui contiendra les sources. C’est sur ce dossier que nous nous concentrerons pour la suite du tutoriel.

Avant de s’intéresser aux sources il convient de modifier le fichier **app.js**. Remplacez le contenu du fichier avec :

```
enyo.kind({
        name: "App",
        kind: "Panels",
        fit: true,
        components: [
            {kind: "myapp.MainView"},
            {kind: "myapp.SubView"},
            {kind: "myapp.SubView2"}
        ]
});
new App().write();
```

Le paramètre "name" correspond simplement au nom de la fonction, le paramètre "kind" indique de quel type de fonction il s’agit. Ici "Panels" correspond à un pattern pour le fonctionnement des transitions entre les différentes vues de l’application. "fit" est une variable qui permet à l’application de savoir si l’on souhaite que les dimensions s’adaptent à tous les écrans.

"components" contient simplement tous les composants de la fonction. Ici, on peut trouver trois vues appelées "myapp.MainView", "myapp.SubView" et "myapp.SubView2". Ces vues sont rangées dans le dossier "views". Pour passer d'une vue à l'autre il suffit de faire glisser le panel sur le côté avec le curseur.

Désormais, il s'agit de remplir les vues pour avoir quelque chose à afficher en lançant l'application. Dans view.js :

```
var symbol = 0;
enyo.kind({
        name: "myapp.MainView",
        kind: "FittableRows",
        fit: true,
        components:[
                {kind: "onyx.Toolbar", content: "Main View"},
                {kind: "enyo.Scroller", fit: true, components: [
                        {name: "main", classes: "nice-padding", allowHtml: true}
                ]},
                {kind: "onyx.Toolbar", components: [
                        {kind: "onyx.Button", content: "1", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "2", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "3", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "4", ontap: "thingTap"}
                ]},
                {kind: "onyx.Toolbar", components: [
                        {kind: "onyx.Button", content: "5", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "6", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "7", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "8", ontap: "thingTap"}
                ]},
                {kind: "onyx.Toolbar", components: [
                        {kind: "onyx.Button", content: "9", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "0", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "+", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "-", ontap: "thingTap"}
                ]},
                {kind: "onyx.Toolbar", components: [
                        {kind: "onyx.Button", content: "*", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "/", ontap: "thingTap"},
                        {kind: "onyx.Button", content: "=", ontap:
"calculationTap"},
                        {kind: "onyx.Button", content: "R", ontap: "resetTap"}
        ]},
```

"symbol" est une variable globale utilisée plus loin, "name" doit correspondre au nom des composants se trouvant dans **app.js**. "FittableRows" est un type de template pour l'affichage des éléments, ceux-ci apparaissent alignés horizontalement sur la page et chaque composant se trouve en dessous du précédent.
Les composants ("components") sont divisés en six parties, avec un objet "Scroller" et cinq objets "Toolbar". L'objet "Scroller" permet de scroller (étonnamment) si le contenu dépasse la taille de l'écran car l'application ne le permet pas automatiquement. Les objets "Toolbar" sont faits pour contenir d'autres objets, la première toolbar ne contient que du texte qu'elle affichera. Les autres toolbar contiennent des boutons, les boutons de la calculatrice. Chaque bouton contient un caractère et est lié à une fonction qui s'active lorsque le bouton est cliqué.
Ces fonctions sont les suivantes :

```
thingTap: function(inSender, inEvent) {
                this.$.main.addContent(inSender.content);
                if(inSender.content == "+" || inSender.content == "-" ||
inSender.content == "*" || inSender.content == "/"){
                        var sym = inSender.content;
                        symbol++;
                        if(symbol > 1){
                                symbol = 1;
                                var text = this.$.main.getContent();
                                text = text.slice(0, -1);
                                var calc;
                                var tab;
```

```
                                if(text.indexOf("*") != -1){
                                        tab = text.split("*");
                                        calc = parseInt(tab[0]) *
parseInt(tab[1]);
                                }
                                if(text.indexOf("/") != -1){
                                        tab = text.split("/");
                                        calc = parseInt(tab[0]) /
parseInt(tab[1]);
                                }
                                if(text.indexOf("+") != -1){
                                        tab = text.split("+");
                                        calc = parseInt(tab[0]) +
parseInt(tab[1]);
                                }
                                if(text.indexOf("-") != -1){
                                        tab = text.split("-");
                                        calc = parseInt(tab[0]) -
parseInt(tab[1]);
                                }
                                this.$.main.setContent(calc);
                                this.$.main.addContent(sym);
                        }
                }
        },
```

La fonction "thingTap" affiche le caractère sélectionné à l'écran et si un deuxième caractère d'opération devait être utilisé la fonction lance le calcul automatiquement et affiche le résultat suivi du symbole précédemment choisi.

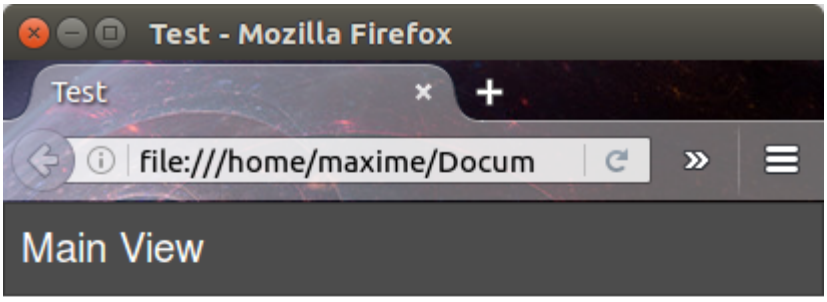
```
calculationTap: function(inSender, inEvent){
                var text = this.$.main.getContent();
                var calc;
                var tab;

                if(text.indexOf("*") != -1){
                        tab = text.split("*");
                        calc = parseInt(tab[0]) * parseInt(tab[1]);
                }
                if(text.indexOf("/") != -1){
                        tab = text.split("/");
                        calc = parseInt(tab[0]) / parseInt(tab[1]);
                }
                if(text.indexOf("+") != -1){
                        tab = text.split("+");
                        calc = parseInt(tab[0]) + parseInt(tab[1]);
                }
                if(text.indexOf("-") != -1){
                        tab = text.split("-");
                        calc = parseInt(tab[0]) - parseInt(tab[1]);
                }
                this.$.main.setContent(calc);
        },
```

"calculationTap" calcule le résultat de l'opération affichée lorsque le bouton "=" est cliqué. La fonction récupère la chaîne de caractères à l'écran et effectue un traitement pour séparer les nombres et les symboles pour ensuite calculer le résultat.

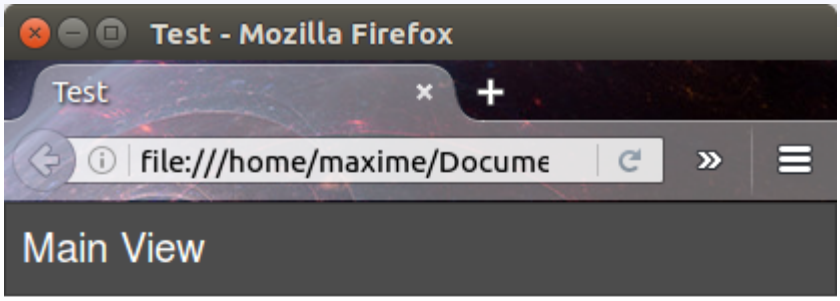
```
resetTap: function(inSender, inEvent) {
                symbol = 0;
                this.$.main.setContent("");
        }
});t(calc);
        },
```

"resetTap" est une fonction qui va simplement réinitialiser l'affichage lorsque le bouton "R" est cliqué.
Une fois l'application lancée l'affichage ressemblera à peu près à ça :



Le design est assez simpliste mais aucun CSS n'a encore été appliqué. Pour modifier le CSS d'un élément il suffit d'ajouter une balise style lors de sa création, comme ceci :

```
{kind: "onyx.Button", content: "1", style:"background-color: red;", ontap:
"thingTap"}
```



Bien sûr, avant de pouvoir réellement afficher cette application il est nécessaire de créer les deux autres vues pour ne pas créer d'erreurs. Creéez donc deux autres fichiers dans **views** que vous appellerez **view1.js** et **view2.js**. Remplissez-les simplement avec le code suivant :

view1.js :

```
enyo.kind({
        name: "myapp.SubView",
        kind: "FittableRows",
        fit: true,
        components:[
                {kind: "onyx.Toolbar", content: "Sub View"}
        ]
});
```

view2.js :

```
enyo.kind({
        name: "myapp.SubView2",
        kind: "FittableRows",
        fit: true,
        components:[
                {kind: "onyx.Toolbar", content: "Sub View 2"}
        ]
});
```

Ensuite ouvrez le fichier **package.js** se trouvant dans **views** et écrivez :

```
enyo.depends(
        "view.js",
        "view1.js",
        "view2.js"
);
```

Et voilà ! L'application est créée et fonctionnelle, les deux autres vues ne contiennent presque rien pour l'instant mais sont accessibles. Libre à vous de les modifier à présent.