Boards

Remise Zippée du projet sur http://foad2.unicaen.fr/moodle/course/view.php?id=20911

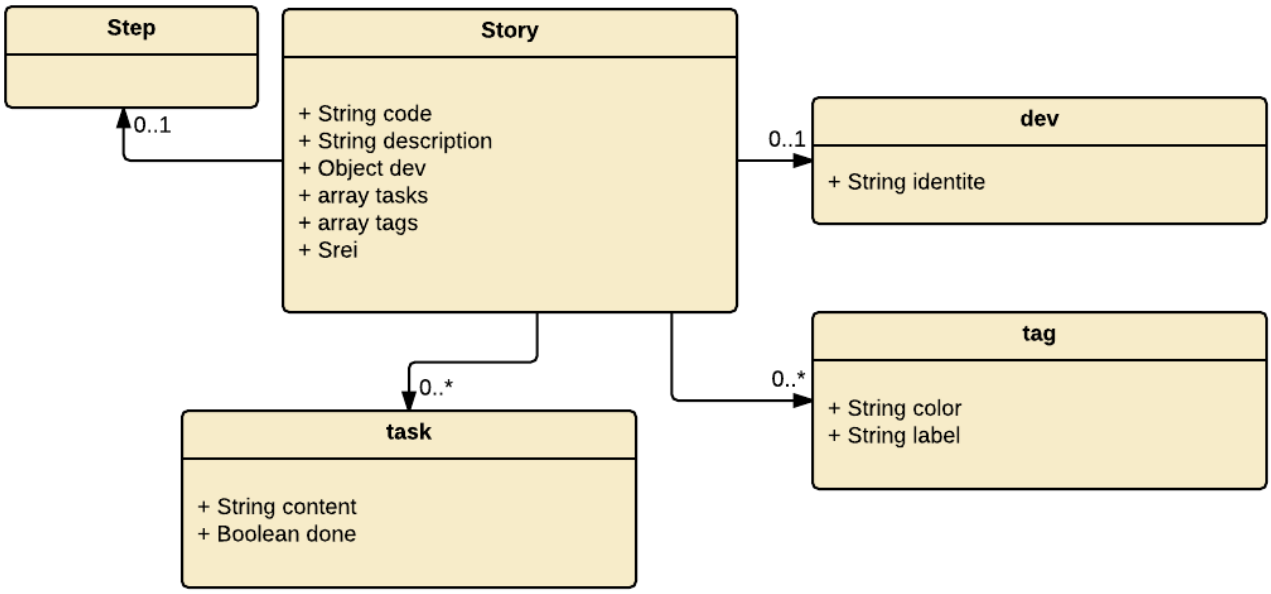
Attention au nommage **prenom.nom** !

-- Contexte

Dans le cadre de l'évaluation du potentiel d'Angular pour les projets de votre entreprise, vous travaillez sur un projet Exemple permettant de gérer les User stories de projets Scrum.

En voici les principales caractéristiques :

- Chaque User story [**story**] a un code et un descriptif.
- Il est possible de lui apposer des tags [**tags**], composés d'une couleur et d'un label.
- Elle peut être affectée à un développeur[**dev**] (qui a juste une identité).
- Elle peut contenir une liste de tâches [**tasks**], à réaliser ou réalisées.
- Une story peut appartenir à une étape [**step**]



-- Eléments à implémenter

- **Strict respect des normes, nommages et consignes** données ⇒ **1 point**
- Usage de **Browserify** pour créer un unique bundle ⇒ **1 point**

-- Squelette de l'application, routage (3 points)

//TODO 2.1.1

Le dossier root de votre application devra être de la forme : **prenom.nom**

Créer la structure suivante :

Emplacement	**Fichier**	**Rôle**
/	index.html	Fichier principal
/js		Fichiers js
	app.js	Fichier principal de l'application (module)
	router.js	Fichier de routage
/js/controllers		contrôleurs
	story.js	Définit le contrôleur **storyController**
	stories.js	Définit le contrôleur **storiesController**
/js/directives		directives
/js/services		services
/templates		templates HTML
/css		Feuille(s) de styles

//TODO 2.1.2

Implémenter le routage suivant :

URL	**Composante**	**Valeur**
/	Description	Affiche la liste des user stories
	template	templates/stories.html
	contrôleur	storiesController
	alias	storiesCtrl
/story/:code	Description	Affiche la user story correspondant à **code**
	template	templates/story.html
	contrôleur	storyController
	alias	storyCtrl
/	Route par défaut	

-- Service (2 points)

//TODO 2.2

Le service **dataService** simule la connexion à un web service ; il est défini de la façon suivante :

stories, **tags** et **devs** seront définis “en dur” à l'intérieur du service dans l'équivalent de variables privées :

```
    var steps=["todo","In progress","done"];
    var stories=[
                    {"code":"B22","description":"En tant que créateur, je veux
ajouter et gérer les réponses d'une question [methods] .",
"tags":[{"color":"red","label":"bug"},{"color":"orange","label":"Admin"}],
                        "dev":{"identite":"James Gosling"},
                        "tasks":[{"content":"get
```

```
reponse/all","do":false},{"content":"get reponse/:id","do":false},{"content":"put
reponse","do":false},{"content":" post reponse/:id","do":false}],
                        "step": "todo"
                    },
                    {"code":"E120","description":"En tant que créateur, je veux
créer / Modifier des quiz [methods]",
                        "tasks":[{"content":"get questionnaire/:id","do":false}],
                        "tags":[]
                    },
                    {"code":"E140","description":"En tant que créateur, je souhaite
gérer les utilisateurs [methods]",
                        "tasks":[{"content":"get
user/all","do":true},{"content":"get user/:id","do":true},{"content":"put
user","do":true}],
                        "dev":{"identite":"Rod Johnson"},
                        "tags":[{"color":"orange","label":"Admin"}],
                        "step":"done"
                    }
    ];
    var
tags=[{"color":"red","label":"bug"},{"color":"#cc317c","label":"question"},{"color"
:"#159818","label":"help
wanted"},{"color":"#cccccc","label":"duplicate"},{"color":"#0052cc","label":"todo"}
];
    var devs=[{"identite":"James Gosling"},{"identite":"Rod
Johnson"},{"identite":"Linus Torvalds"}];
```

Service	**datas.js** (dataService)
Variables privées	stories Tableau des user stories
	steps Tableau des steps existantes
	tags Tableau des tags disponibles
	devs Tableau des développeurs
Méthodes publiques	getStories() retourne **stories**
	getSteps() retourne **steps**
	getTags() retourne **tags**
	getDevs() Retourne **devs**
	getStory(code) Retourne la user story correspondant au **code** donné, **null** si le code est inexistant
	setStoryAvancement(story) ajoute et valorise le membre **avancement** de la **story** passée en paramètre (avancement=nb tâches réalisées/ nb tâches). Une tâche est réalisée si son membre done=true. A l'issue du passage **story.avancement** est existant et correctement valorisé

Créez dataService, implémentez les méthodes publiques.
Il faudra ensuite injecter **dataService** aux 2 contrôleurs **storiesController** et **StoryController**

-- Url /stories ou / (4 points)

//TODO 2.3

Affiche la liste des user stories.

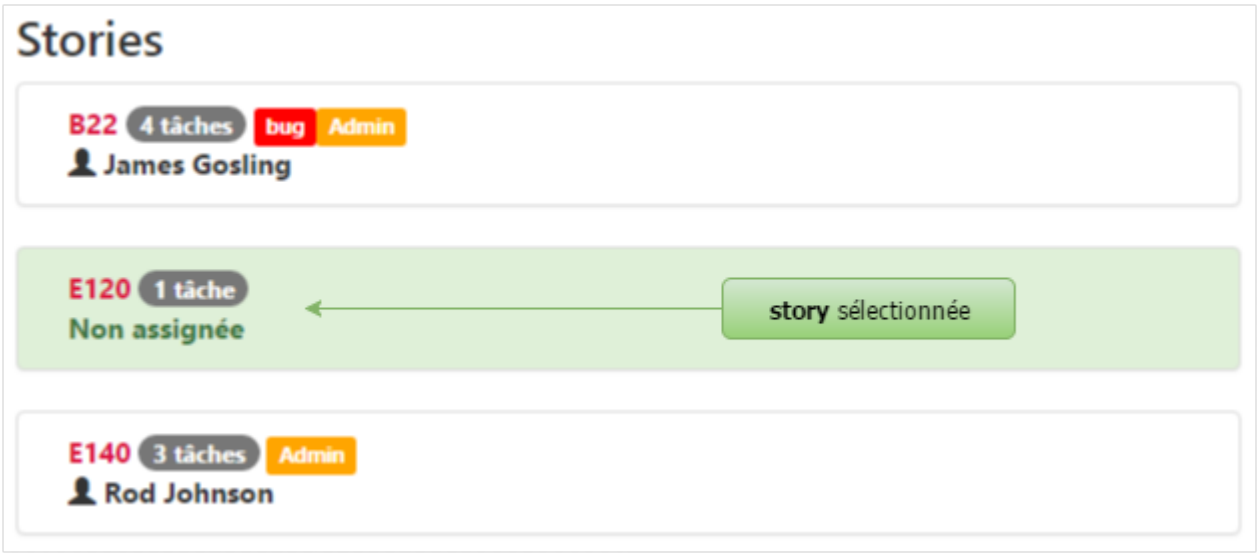


Créer le template **/templates/stories.html**, associé au contrôleur **/js/controllers/stories.js**

Controller	**stories.js** (storiesController)
Variables publique	stories Tableau des user stories
	selected **story** sélectionnée
Méthodes publiques	goto(story) Accède à l'url **story/:code** en utilisant le service **$location**

Comportement de l'interface :

- Le double-clic sur une story permet d'accéder à l'url **/story/:code** où code est celui de la story cliquée
- Le clic sélectionne la story (stockée dans le membre **selected** du contrôleur)
- La classe css de la story sélectionnée est “selected”



-- Directive storyHeader (2 points)

//TODO 2.4



On souhaite intégrer l'entête de chaque story affichée à l'url /stories dans une directive, pour la réutiliser plus facilement dans la page suivante, pour présenter le détail de chaque story.

La directive **storyHeader**, accessible en tant qu'élément, ou attribut, permettra d'afficher le contenu suivant d'une story. Il sera nécessaire de lui passer la variable story (qu'elle est en charge d'afficher) dans son **scope**.

- Déclarer la directive **storyHeader**,
- intégrer la dans l'application,
- créer le template correspondant,
- utilisez la à l'url **/stories**

On respectera la structure suivante :

Emplacement	**Fichier**	**Rôle**
/js/directives		Dossier définissant les directives
	storyHeader.js	Fichier définissant la directive storyHeader
/js/directives/templates		Fichiers templates
	storyHeader.html	Fichier template de la directive storyHeader

-- Url /stories ou / (3 points)

//TODO 2.5

Affiche les **stories** par step.



Filtre sur ng-repeat

On utilisera la directive **ng-repeat** avec un filtre (filter) composé d'une expression (testant le membre **step** de chaque **story**).

Le filtre peut faire appel à une méthode définie dans le contrôleur :

Exemple :

```
<div ng-repeat="item in items | filter: myCtrl.check"></div>
```

```
this.check=function(item){
    return item.code=="AAA";
};
```

La boucle **ng-repeat** affichera dans ce cas les éléments du tableau **items** dont le membre **code** est égal à "AAA"

Disposition des éléments

On utilisera le GridSystem Bootstrap pour disposer correctement les conteneurs de step sur la page :

- La grille Bootstrap est composée de 12 colonnes, chaque classe css de type **col-md-n** attribuée à un élément permet de lui donner une largeur de n/12.
- Pour nos steps, si nous en avons 3, chacune d'entre elles devra avoir une largeur de **12/3=4** et donc avoir la classe **col-md-4**

Modifier le template **stories.html** et le contrôleur **storiesController** :

Controller	**stories.js** (storiesController)
Méthode publique à ajouter	getColClass() Retourne la classe css à attribuer aux colonnes (col-md-x) en fonction du nombre de **steps**

-- Url /story/:code (3 points)

//TODO 2.6

Affiche la **story** correspondant au **code** passé dans l'url.



Comportement de l'interface :

- Une task **done** est barrée
- Les cases à cocher permettent de faire passer une tâche de non réalisée (done=false) à réalisée (done=true) et inversement

Créer le template **/templates/story.html**, associé au contrôleur **/js/controllers/story.js**

Controller	**story.js** (storyController)
Variables privées	story user story à afficher
Méthodes publiques	toggleDone(task) Bascule de vrai à faux et inversement le membre **done** de la tâche **task** passée en paramètre

-- A poursuivre...

//TODO 2.7

Fonctionnalités supplémentaires à implémenter :

Dans la page /stories ou / :

- Ajouter/modifier/supprimer des steps
- Changer l'affectation d'une story dans une step

Super-bonus (20 garantis) :

- Mettre en place le drag & drop des stories entre les steps avec un module Angular

-- Ressources

HTML/CSS

```
<!DOCTYPE html>
<html>
<head>
    <base href="http://127.0.0.1/yoursite/">
    <meta charset="UTF-8">
    <link rel="stylesheet"
href="http://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/css/bootstrap.min.css">
    <link rel="stylesheet" href="css/styles.css">

    <script type="text/javascript"
src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.2/jquery.min.js"></script>
    <script async type="text/javascript"
src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.6/js/bootstrap.min.js"></script>


    <script
src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.9/angular.min.js"></script>
    <script
src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.9/angular-route.min.js"></s
cript>

    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
</head>
<body>
<div class="container">
    <div class="panel panel default">
        <div class="panel-body">

        </div>
    </div>
</div>
</body>
</html>
```

```
<IfModule mod_rewrite.c>
    Options +FollowSymlinks
    RewriteEngine On
    RewriteBase /yoursite/
    RewriteCond %{REQUEST_FILENAME} !-f
    RewriteCond %{REQUEST_FILENAME} !-d
    RewriteCond %{REQUEST_URI} !.*\.(css|js|html|png|jpg|jpeg|gif|txt|ttf|woff)
    RewriteRule (.*) index.html [L]
</IfModule>
```

```
body{
```

```
    font-family: 'Segoe UI', Frutiger, 'Frutiger Linotype', 'Dejavu Sans',
'Helvetica Neue', Arial, sans-serif;
}
.story-panel{

}
.story{
    padding: 10px;
    cursor: pointer;
}
.story-code{
    font-weight: bold;
    color: crimson;
}

.story-panel span{
    vertical-align: inherit;
}
.assign-to{
    font-weight: bold;
}
.selected{
    background-color: #dff0d8;
    color: #3c763d;
}
.selected::after{
    content: '';
    display: block;
    clear: both;
}
```

Bootstrap

Done

```
  <span class="glyphicon glyphicon-tasks" aria-hidden="true"></span> Done
```

3 tâches

```
<span class="badge">3 tâches </span>
```

Admin

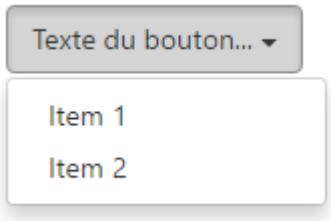
```
<span class="label" style="color: orange">Admin</span>
```

Texte du bouton

```

```

```
<button type="button" class="btn btn-default" aria-label="Left Align">
  <span class="glyphicon glyphicon-align-left" aria-
hidden="true"></span> Texte du bouton
</button>
```




```
<div class="btn-group">
    <button type="button" class="btn btn-default dropdown-toggle" data-
toggle="dropdown" aria-haspopup="true" aria-expanded="false">
        Texte du bouton... <span class="caret"></span>
    </button>
    <ul class="dropdown-menu">
        <li>
            <a href="#">
                Item 1
            </a>
        </li>
        <li>
            <a href="#">
                Item 2
            </a>
        </li>
    </ul>
</div>
```




```
<div class="panel panel-default">
    <div class="panel-heading">
        En-tête
    </div>
    <div class="panel-body">Body</div>
    <ul class="list-group">
        <li class="list-group-item">
            Item 1
        </li>
        <li class="list-group-item">
```

```
            Item 2
        </li>
    </ul>
</div>
```

From:
http://slamwiki2.kobject.net/ - **SlamWiki 2.1**

Permanent link:
http://slamwiki2.kobject.net/slam4/richclient/angularjs/boards-2?rev=1458695498

Last update: **2019/08/31 14:40**