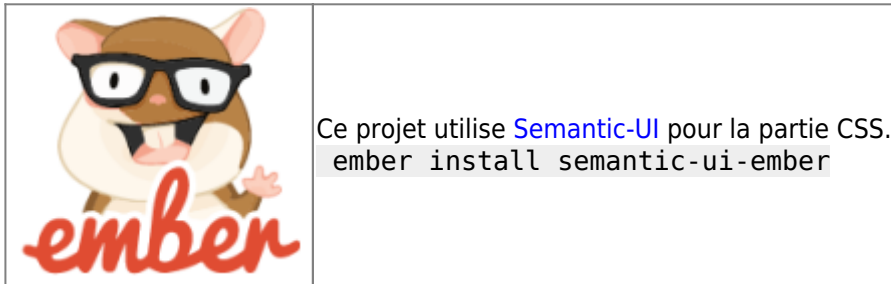


TD n°4



- Projet **boards**
- Application gestion de projets SCRUM

Objectifs

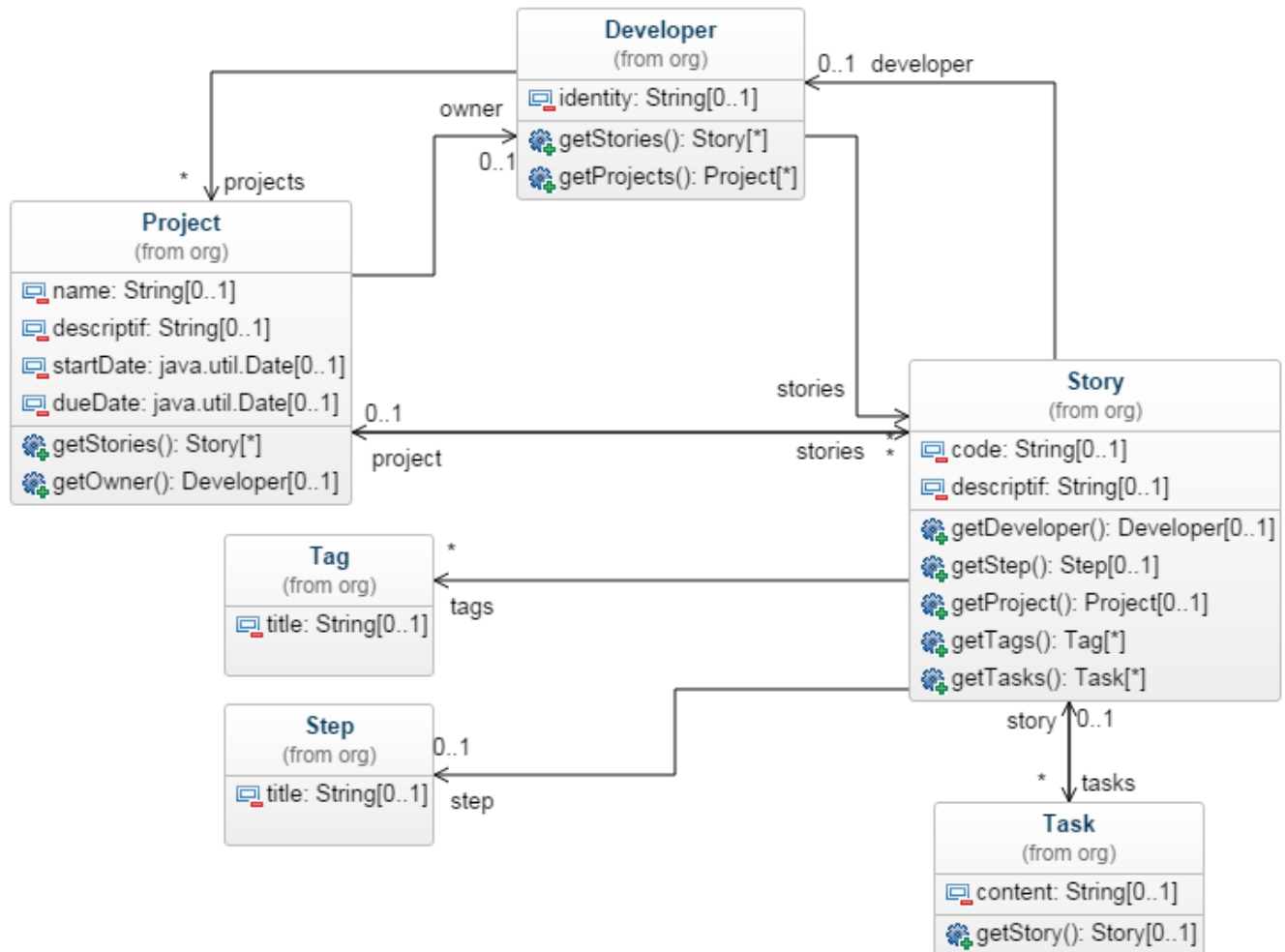
1. Se connecter à un service web externe
2. Créer des composants
3. Manipuler les models

Contexte

Vous travaillez sur un outil permettant de gérer des projets.

Voici les principales caractéristiques du système d'information :

- Chaque projet [**project**] possède un nom, un descriptif, une date de début et de fin, et un propriétaire (owner, qui est un développeur)
- L'équipe est constituée d'un ensemble de développeurs [**developer**].
- Chaque User story [**story**] a un code et un descriptif, et appartient à un projet.
- 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.



Création de la base de données

Démarrer **mongoDb** :

```
mongod
```

Démarrer le serveur **restheart** :

```
java -jar restheart.jar
```

En cas de problème de port avec mongoDb, exécuter **mongod** sur un autre port (27017 par défaut) :

```
mongod --port 27117
```

En cas de problème de port (8080 par défaut) avec restheart, modifier le fichier de configuration **etc/restheart.yml** :

1. Modifier le port 8080
2. commenter la partie sécurité à partir de la ligne 165)
3. Exécuter **restheart.jar** en spécifiant le fichier de configuration modifié

```
java -jar restheart.jar etc/restheart.yml
```

Exécuter **mongo** (client en ligne de commande pour mongoDb) :

Créer les collections suivantes dans la base de données mongoDb **boards** :

- projects
- developers
- steps
- stories
- tags
- tasks

Exemple : Pour créer une collection dans la console mongo :

```
use boards
db.createCollection('developers',{})
```

Création du projet, intégration des composants

Créer le projet **boards**

Ajouter le plugin ember-cli-uuid pour la génération d'uuid côté client :

```
ember install ember-cli-uuid
```

Adapter & serializer

Générer l'adapter pour l'application :

```
ember g adapter application
```

Modifier l'adapter comme ci-dessous :

```
import DS from 'ember-data';
import { pluralize } from 'ember-inflector';

var Adapter=DS.RESTAdapter.extend({
  ajaxOptions: function(url, type, options) {
    var hash = this._super(url, type, options);
    if (type == 'POST' || type=='PUT') {
      hash.dataType = 'text';
    }
    return hash;
  },
  host: 'http://127.0.0.1:8080',
  namespace: 'boards',
  urlForDeleteRecord(id, modelName) {
    modelName=pluralize(modelName);
    return
    this.get('host')+'/' +this.get('namespace')+'`/${modelName}/*?filter={_id:'${id}'}';
```

```

    }
  });

export default Adapater;

```

Générer le serializer pour l'application :

```
ember g serializer application
```

Modifier le serializer comme ci-dessous :

```

import DS from 'ember-data';

export default DS.JSONSerializer.extend({
  primaryKey: '_id',
  isNewSerializerAPI: true,
  extractId: function (modelClass, resourceHash) {
    if(resourceHash._id)
      return (resourceHash._id.$oid || resourceHash._id);
  },
  normalizeResponse(store, primaryModelClass, payload, id, requestType) {
    if(requestType==='createRecord')
      return this._super(store, primaryModelClass, {}, id, requestType);
    if(requestType==='updateRecord')
      return this._super(store, primaryModelClass, {}, id, requestType);
    if(requestType==='deleteRecord')
      return this._super(store, primaryModelClass, null, id, requestType);
    if (payload._embedded)
      return this._super(store, primaryModelClass, payload._embedded, id, requestType);
    return this._super(store, primaryModelClass, payload, id, requestType);
  },
  serializeId (snapshot, json) {
    let id = snapshot.id;
    json['_id'] = id;
  }
});

```

Autoriser la connexion de l'application ember au serveur **restheart** :

```

...
ENV.contentSecurityPolicy = {
  // ... other stuff here
  'connect-src': "'self' http://127.0.0.1:8080"
}
...

```

Partie Administration

Créer les models **project** et **developer**, ajouter les membres.

```
ember g model project
ember g model developer
```

Route developers

```
ember g route developers
```

Home

Projects

Developers

 **models\Developer**
Data administration

+ Add a new models\Developer...

Identity	
Evan You	
Fabien Potencier	
John Resig	
Kris Selden	
Yehuda Katz	

Route developers/new

```
ember g route developers/new
```

 **models\Developer**
New object creation

Identity

Evan YOU

 **Validate**

 **Cancel**

Confirmation avant suppression

A ajouter pour la route **developers**



Confirmez la suppression de `Evan You`?

 **Confirmer**

 **Annuler**



Route projects

ember g route projects

Home

Projects

Developers

 **models\Project**
Data administration

 **Add a new models\Project...**

Name	Descriptif	StartDate	DueDate		
Boards-EmberJS	Gestion de projet SCRUM avec EmberJS	2018-02-20	2018-02-28		
phpMyBenchmarks	Benchmarks PHP	2018-02-20	2018-03-21		
Cloud 66 for Rails	Build, deploy, and maintain your Rails apps on any cloud or server	2017-07-22	2017-08-01		
Codecov	Group, merge, archive and compare coverage reports	2017-10-09	0000-00-00		
ZenHub	Agile Task Boards, Epics, Estimates and Reports, all within GitHub's UI	2016-11-14	0000-00-00		

Route projects/new

```
ember g route projects/new
```



models\Project
Editing an existing object

Name

Codecov

Descriptif

Group, merge, archive and compare coverage reports

StartDate

2017-10-09

DueDate

0000-00-00

Owner

Kris Selden

✓ Validate

⊗ Cancel

Gestion des stories par projet

- Créer au besoin les collections **tags** et **stories** dans MongoDB
- Ajouter au projet ember les models **tag** et **story**, définir les membres et relations (belongsTo et hasMany)

Ajout d'un adapter pour la sérialisation des stories d'un projet

Créer le serializer suivant pour le model **project** :

```
import DS from 'ember-data';
import ApplicationSerializer from './application';

export default ApplicationSerializer.extend(DS.EmbeddedRecordsMixin, {
  attrs: {
    stories: {
```

```
    deserialize: false,  
    serialize: 'ids'  
  },  
  tags: {  
    serialize: 'ids'  
  }  
}  
});
```

Route /project/:project_id

Elle permet de visualiser les caractéristiques d'un projet, ainsi que ses stories.

```
ember g route project
```

From:

<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:

<http://slamwiki2.kobject.net/richclient/emberjs/td4?rev=1520903411>

Last update: **2019/08/31 14:38**

