UNIVERSITÉ de Cergy-Pontoise

LP WAM - DU II

POO et Java

Projet

1 Organisation

1.1 Les groupes

Constitution des groupes :
- Groupe 1 : **Saman**, Benjamin, Maxime
- Groupe 2 : **Ulysse**, Amadou, Nikola
- Groupe 3 : **Enzo**, Baptiste, Rodolphe
- Groupe 4 : **Esteban**, Thierno, Mouloud

Chaque membre du groupe assumera à tour de rôle la responsabilité de chef de projet pendant un mois.

1.2 Planning et conseils pour la présentation

Planning
- identifier les principales sous-tâches du projet à réaliser, la répartition des rôles au sein du groupe ainsi que le planning correspondant pour chaque tache (période et durée). Vous pouvez créer un diagramme de GANTT [1] de votre projet. Dans tous les cas, le diagramme de Gantt est à rendre au format png (capture ou export) avant le samedi 25 janvier 2020 : 1 point
- deux points d'avancement (fin janvier et fin février) : 3 points
- soutenance (5 min) et démonstration (5 min) : 6 points : fin mars

Soutenance : 5 diapositives maximum, 5 minutes maximum
- la diapositive de titre présentera le groupe, le contexte, le sujet. (i.e. : la page de garde sera compacte),
- les autres diapositives doivent présenter les spécificités de réalisation de l'équipe projet, donc aucune information "évidente" (ex. détail du sujet, progression personnelle, ...) ne devra être mentionnée,
- la diapositive de conclusion mettra en évidence le niveau d'achèvement du projet (points traités et non traités du cahier des charges et extensions s'il y en a),
- vous devrez prévoir une version pdf de votre diaporama au cas où.

Chaque membre du groupe devra intervenir lors de la présentation. Les informations suivantes devront être présentes : la répartition des tâches, les principaux éléments de conception.

NB : à éviter ABSOLUMENT : les diagrammes de classes UML illisibles (trop chargés, ...), les programmes (code Java), les captures d'écran (puisqu'il y a aussi une démonstration), la liste des outils (ex. Eclipse, etc.)...

Important : pour la soutenance, vous devez avoir votre machine portable de l'université avec votre présentation et vos programmes ET une clé USB contenant votre diaporama (en PDF).

Démonstration : 5 minutes maximum
- mode console,
- mode graphique.

Vous veillerez à prévoir un scénario pour la démonstration.

2 Projet

2.1 Contexte

L'objectif du projet est de créer une petite application permettant de manipuler, dans une première phase, des cartes de visites (virtual card) au format numérique et dans un second temps des événements d'agenda

1. https ://www.ganttproject.biz/download/free

(rendez-vous, tâche à réaliser, etc.). Ce type de carte de visite est utilisé dans un contexte professionnel dans les carnets d'adresses des clients de messagerie ou en élément de la signature lors des échanges par mail par exemple, et ce type de calendrier d'événements est utilisé, par exemple, dans l'ENT pour vos emplois du temps.

Les principales actions de votre logiciel sont :

— la lecture et l'affichage (mise en forme) d'un fichier de carte de visite (vcf) ou d'un fichier de calendrier (ics),
— la possibilité de modifier partiellement le contenu via une interface graphique et d'enregistrer ces modifications,
— la possibilité de transformer ce fichier en un fragment de code HTML exploitant les microformats correspondants.

En mode console (terminal) « CLI [2] » : les paramètres attendus sur la ligne de commande sont : en entrée, le nom du fichier (de type vcf ou ics) à traiter et en sortie, le nom du fichier (sérialisation ou html) à produire. Si le fichier de sortie n'est pas spécifié, le programme doit afficher directement dans la console le contenu du fichier mis en forme. Si aucun paramètre n'est indiqué, le programme affiche les options possibles. Une des options permet d'afficher la liste des fichiers vcf et ics disponibles sur la machine en les recherchant, à partir du dossier spécifié, dans l'ensemble de ses sous- répertoires.

En mode graphique « GUI [3] » : l'exploration d'une arborescence quelconque (dossiers et sous-dossiers) de fichiers permettant de lister uniquement les fichiers de type vcf et ics et leur emplacement est attendue. La sélection d'un de ces fichiers provoquera son affichage (mais on ne recherche pas nécessairement un affichage complet), la possibilité de modifier les principaux champs (là encore, on ne cherche pas à traiter tous les cas particuliers lors des modifications) et leur enregistrement, et enfin l'option de génération du fragment HTML correspondant (seuls les éléments les plus significatifs sont à exporter).

Il est donc possible de lancer une exécution en mode console, d'enregistrer le résultat en utilisant la sérialisation puis de lancer une exécution en mode graphique et d'ouvrir directement la sauvegarde précédente (sans relire le fichier « vcf / ics » initial).

Quelques scénarios d'exécution (exemples fictifs de lancement de vos 2 programmes) :

```
java -jar cli.jar
java -jar cli.jar -d .
java -jar cli.jar -i edt.ics
java -jar cli.jar -i edt.ics -o edt.ser
java -jar cli.jar -i edt.ics -h fragment.html
java -jar gui.jar
```

Explications :

— la première ligne affiche les modes d'utilisation de votre logiciel en mode console (i.e. les options possibles et leur rôle) ;
— la deuxième ligne liste tous les fichiers ics ou vcf à partir du dossier spécifié (« -d » = directory) [ici le dossier courant (« . »)] en mode console ;
— la troisième ligne prend en entrée le ficher « edt.ics » (« -i » = input) et affiche à l'écran son contenu mis en forme en mode console ;
— la quatrième ligne prend en entrée le ficher « edt.ics » et sauvegarde le contenu structuré dans un fichier « edt.ser » (« -o » = output) en mode console ;
— la cinquième ligne prend en entrée le ficher « edt.ics » et génère le fichier « fragment.html » (« -h » = html) en mode console. Vous pouvez prévoir une option supplémentaire « -p » (« - p » = page) permettant de créer un squelette de page HTML valide contenant le fragment ;
— la dernière ligne correspond au lancement de l'interface graphique.

Extensions possibles : vous pouvez prévoir des améliorations à votre solution mais uniquement si tout le reste est complet. Vous privilégierez donc la qualité de la réalisation à la quantité de ses fonctionnalités.

2.2 Le jeu de données

Pour ce projet, vous devrez rechercher des exemples simples sur le web pour les fichiers de départ `.vcf` et `.ics` en vous appuyant, par exemple, sur les liens ci-dessous :

— https ://icalendar.org/
— https ://www.w3.org/2002/12/cal/vcard-notes.html
— http ://microformats.org/wiki/h-event
— http ://microformats.org/wiki/h-card

2. CLI : Command Line Interface
3. GUI : Graphical User Interface

— https ://support.office.com/en-us/article/Send-and-save-contacts-as-vCards-vcf-files-94A17A6F-105F-46C7-9308-33658C1C2690
— https ://support.mozilla.org/en-US/kb/how-use-virtual-card-vcard

Pour vos tests et votre démonstration, vous utiliserez des cas d'utilisation concrets comme par exemple : une carte de visite générée à partir d'un client mail ou de votre webmail, votre emploi du temps sur l'ENT, un agenda utilisé sur votre poste ou en ligne. N.B. : il vous appartient d'utiliser les validateurs (icalendar / vcard / html) pour vérifier la qualité de vos fichiers aussi bien en entrée qu'en sortie :

— https ://icalendar.org/validator.html
— https ://validator.w3.org/

2.3 Résultats attendus

Complétude et qualité du projet : 10 points (fichier « readme.txt », code java, javadoc, fichiers jar).
Les livrables sont à déposer sur la plate-forme pédagogique avant le vendredi 27 mars 2020 (à midi).

— fichier « readme.txt » contenant les noms / prénoms / groupe TD / des membres du projets ainsi que les informations spécifiques utiles : 1 point
— rapport de projet (minimum 5 pages, maximum 15 pages) : 3 points (le fond et la forme seront évalués). Les 2 fichiers suivants sont à rendre : 1) le document de traitement de texte (docx ou odt), 2) la version pdf de votre rapport
— ensembles des fichiers sources du projet (.java) : 2 points
— la javadoc : 2 points
— les 2 fichiers jar en version compatible java 1.8 : 2 points

Important : Tous les fichiers et sous-dossiers à remettre doivent être placés dans un répertoire unique portant les 3 noms du groupe (sous la forme NOM1_NOM2_NOM3) à compresser en un seul fichier au format zip qui sera déposé sur la plate-forme pédagogique de cours.

2.3.1 Quelques indications pour la réalisation

L'objectif du projet est de vous permettre de mettre en œuvre, dans le cadre d'une réalisation concrète, les notions de POO et Java abordées au cours du module. Il n'est pas nécessaire de vouloir être exhaustif dans le traitement des nombreuses situations présentes dans les fichiers manipulés. De même, il est possible d'utiliser des bibliothèques externes correspondant à vos besoins.

Première partie Pour démarrer le projet, il est recommandé de traiter dans un premier temps le cas (plus simple) des cartes de visite : fichier vCard et export au format h-card. Si vous ne parvenez pas à générer le fragment HTML, passez néanmoins à la seconde partie.

Seconde partie Dans un deuxième temps, vous pourrez aborder les fichiers de calendrier (« .ics »).

2.3.2 Quelques indications pour le rapport

Ce document rédigé à l'aide d'un traitement de texte (Word ou LibreOffice par exemple) doit permettre de fournir un compte-rendu complet et un bilan de votre travail et de son aboutissement. Vous y placerez en particulier le diagramme de classes UML de votre application. Les informations de planning et de répartition des tâches sont attendues. Des explications sur les aspects particuliers de votre solution. Vous pouvez ajouter quelques captures d'écran représentatives mais n'en abusez pas. Vous détaillerez le niveau d'aboutissement de votre réalisation avec également un regard critique sur les points forts et les points faibles que vous aurez identifiés.

2.4 Quelques exemples [4]

Vcard

```
BEGIN:VCARD
VERSION:3.0
N:Gump;Forrest;;Mr.;
FN:Forrest Gump
ORG:Bubba Gump Shrimp Co.
TITLE:Shrimp Man
PHOTO;VALUE=URI;TYPE=JPEG:http://www.example.com/dir_photos/my_photo.jpg
TEL;TYPE=WORK,VOICE:(111) 555-1212
TEL;TYPE=HOME,VOICE:(404) 555-1212
ADR;TYPE=WORK,PREF:;;100 Waters Edge;Baytown;LA;30314;United States of America
LABEL;TYPE=WORK,PREF:100 Waters Edge Baytown, LA 30314 United States of America
ADR;TYPE=HOME:;;42 Plantation St.;Baytown;LA;30314;United States of America
LABEL;TYPE=HOME:42 Plantation St.Baytown, LA 30314 United States of America
EMAIL:forrestgump@example.com
REV:2008-04-24T19:52:43Z
END:VCARD
```

fragment HTML hCard

```
<div class="vcard">
  <a class="fn org url" href="http://www.commerce.net/">CommerceNet</a>
  <div class="adr">
    <span class="type">Work</span>:
    <div class="street-address">169 University Avenue</div>
    <span class="locality">Palo Alto</span>,
    <abbr class="region" title="California">CA</abbr>
    <span class="postal-code">94301</span>
    <div class="country-name">USA</div>
  </div>
  <div class="tel">
    <span class="type">Work</span> +1-650-289-4040
  </div>
  <div>Email:
    <span class="email">info@commerce.net</span>
  </div>
</div>
```

4. Source : https ://en.wikipedia.org/ et http ://microformats.org/wiki/hcard

Principales propriétés du format hCard

Properties

Common hCard properties (inside class **vcard**)

- **fn** - name, formatted/full. required
- **n** - structured name, container for:
  - **honorific-prefix** - e.g. Ms., Mr., Dr.
  - **given-name** - given (often first) name
  - **additional-name** - other/middle name
  - **family-name** - family (often last) name
  - **honorific-suffix** - e.g. Ph.D., Esq.
- **nickname** - nickname/alias, e.g. IRC nick
- **org** - company/organization
- **photo** - photo, icon, avatar
- **url** - home page for this contact
- **email** - email address
- **tel** - telephone number
- **adr** - structured address, container for:
  - **street-address** - street #+name, apt/ste
  - **locality** - city or village
  - **region** - state or province
  - **postal-code** - postal code, e.g. U.S. ZIP
  - **country-name** - country name
- **bday** - birthday. ISO date.
- **category** - for tagging contacts
- **note** - notes about the contact

```
<div class="vcard">
   <span class="fn">Sally Ride</span>
   (<span class="n">
      <span class="honorific-prefix">Dr.</span>
      <span class="given-name">Sally</span>
      <abbr class="additional-name">K.</abbr>
      <span class="family-name">Ride</span>
      <span class="honorific-suffix">Ph.D.</span></span>),
   <span class="nickname">sallykride</span> (IRC)
   <div class="org">Sally Ride Science</div>
   <img class="photo" src="http://example.com/sk.jpg"/>
   <a class="url" href="http://sally.example.com">w</a>,
   <a class="email" href="mailto:sally@example.com">e</a>
   <div class="tel">+1.818.555.1212</div>
   <div class="adr">
      <div class="street-address">123 Main st.</div>
      <span class="locality">Los Angeles</span>,
      <abbr class="region" title="California">CA</abbr>,
      <span class="postal-code">91316</span>
      <div class="country-name">U.S.A</div></div>
   <time class="bday">1951-05-26</time> birthday
   <div class="category">physicist</div>
   <div class="note">1st American woman in space.</div>
</div>
```

iCalendar

```
BEGIN:VCALENDAR
PRODID:-//XYZproduct//EN
VERSION:2.0
BEGIN:VEVENT
URL:http://conferences.oreillynet.com/pub/w/40/program.html
DTSTART:20051005
DTEND:20051008
SUMMARY:Web 2.0 Conference
LOCATION:Argent Hotel\, San Francisco\, CA
END:VEVENT
END:VCALENDAR
```

Fragment HTML hCalendar

```
<div class="vevent">
  <a class="url"
    href="http://conferences.oreillynet.com/pub/w/40/program.html">
    http://conferences.oreillynet.com/pub/w/40/program.html
  </a>
```

```
  <span class="summary">Web 2.0 Conference</span>:
  <abbr class="dtstart" title="2005-10-05">October 5</abbr>-
  <abbr class="dtend" title="2005-10-07">7</abbr>, at the
  <span class="location">Argent Hotel, San Francisco, CA</span>
</div>
```