Cégep Ste-Foy – Hiver 2017
Programmation Mobile I – 420-V41-SF

Énoncé du Travail Pratique 1
- Partie 2 -

22 février 2017

Préparé par
Daniel Huot et Benjamin Lemelin

1 Résumé du travail

L'application à développer devra permettre de calculer le poids d'un composé chimique à partir de sa formule écrite. Votre application devra fonctionner sur deux plateformes : en mode console sur la plateforme Java SE et en mode graphique sur la plateforme Android.

2 Conditions de réalisation du travail

Valeur sur la note finale	Type	Durée	Nombre de remises
10 %	Individuel	2 semaines	2

3 Travail à effectuer – Remise 2

Cette section vise à vous présenter les différentes étapes du travail à effectuer pour cette première remise.

3.1 Importer et comprendre le code

Vous trouverez avec cet énoncé deux projets de base à utiliser pour cette remise. Le premier est un petit projet Java où vous implémenterez l'application en mode console. Le second est un projet Android où vous implémenterez l'application sur une plateforme mobile.

3.2 Importez le code sur GitHub, GitLab ou BitBucket

Créez deux nouveaux dépôts privés, l'une pour votre application Java, l'autre pour votre application Android. Importez les deux projets dans ces dépôts.

3.3 Incorporer la librairie de la première remise aux projets

Compilez la librairie faite à la remise 1 en un fichier « .jar » et placez-en une copie dans le dossier « libs » de chacun de vos projets.

Faites ensuite un « Sync » Gradle en cliquant sur le bouton «  » dans l'onglet « Gradle » de IntelliJ IDEA et Android Studio.

3.4 Implémenter l'application Java en mode console

En utilisant votre librairie, faites une application Java en mode console qui demande une formule chimique et calcule le poids total.

```
Entrez un formule chimique :
H2O
Le poids de H2O est 18,015280 g/mol.
```

S'il y a erreur, vous devez l'afficher.

```
Entrez un formule chimique :
H2(
Il y a une parenth├¿se ouvrante sans sa parenth├¿se fermante.
```

(*Notez que les accents s'affichent incorrectement sous Windows. C'est normal.*)

Il n'y a pas de menu dans cette application. Vous saisissez une entrée, elle vous donne une réponse, l'application arrête.

Votre application devra être en français. Voici les chaines de caractères à utiliser :

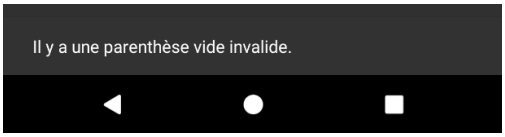
`Entrez une formule chimique : \n`
`Le poids de %s est %f g/mol.\n`
`Le caractère \"%s\" est illégal dans une formule chimique.\n`
`La formule saisie est vide.\n`
`Un exposant invalide est placé avant même un élément chimique ou une parenthèse.\n`
`Il y a une parenthèse vide invalide.\n`
`L'élément chimique \"%s\" est inconnu.\n`
`Il y a une parenthèse ouvrante sans sa parenthèse fermante.\n`
`Il y a une parenthèse fermante sans sa parenthèse ouvrante.\n`

Votre application console devra utiliser un style architectural MVC. Vous devrez donc créer une vue et un contrôleur. Aussi, le projet qui vous est donné contient des tests d'acceptation. Assurez-vous qu'ils passent.

3.5 Implémenter l'application Android

En utilisant votre librairie, faites une application Android faisant les mêmes choses que l'application console. À vous de concevoir une interface mobile qui respecte les principes vus en classe.

N'oubliez pas non plus d'afficher les messages d'erreur. Par contre, vous devrez afficher les messages d'erreur avec « Snackbar » obligatoirement.



Votre application devra être en français et en anglais. Vous avez déjà les chaines de caractères en anglais, il ne reste qu'à créer celles en français. Encore une fois, votre application devra utiliser un style architectural MVC. Aussi, le projet qui vous est donné contient des tests d'acceptation. Assurez-vous qu'ils passent.

3.6 Complément d'information – Lire un fichier sous Android

La liste de tous les éléments chimiques du tableau périodique se trouve dans le fichier « res/raw/chemicalelements.txt ». Comme il se trouve dans le dossier « res », il a donc un identifiant dans la classe « R ».

Pour lire votre fichier, vous devez appeler la méthode « openRawResource ». Cela vous retourne un « InputStream » que vous pouvez lire comme d'habitude. Par exemple :

```
InputStream inputStream = getResources().openRawResource(R.raw.chemical_elements);
```

Comme toujours, n'oubliez pas de le fermer après avoir tout lu et de gérer les exceptions potentielles. Par contre, si la lecture échoue, faites planter l'application tout simplement, car ce n'est pas une situation normale et qui nécessite un message d'erreur.

Enfin, ce fichier est déjà en français et n'a donc pas besoin d'être traduit ni d'être traduit en anglais (mais c'est un BONUS par contre).

3.7 BONUS – Traduire le fichier « chemicalElements.txt »

Créez une variante en anglais du fichier « chemicalElements.txt ». Cela n'aura aucun impact sur votre affichage, mais cela vous permettra de constater que toute ressource peut être remplacée par un changement de configuration (langue, orientation d'écran, taille d'écran, etc…).

Il n'y a pas de tests à écrire pour cela, car cela n'a aucun impact.

3.8 BONUS – Mieux afficher la formule écrite

En temps normal, les chiffres d'une formule chimique sont affichés sous forme d'indice. Pour l'instant, votre affichage ressemble à ceci :

$$H2O$$

Il devrait plutôt ressembler à ceci :

$$H_2O$$

Faites en sorte que si l'utilisateur tape un nombre, il soit affiché en tant qu'indice. Vous pouvez soit utiliser un « InputFilter » pour faire cela, ou modifier l'interface et ajouter un « TextView » qui, soit dit en passant, peut afficher du « HTML ». Dans les deux cas, faites une recherche pour savoir comment procéder.

Notez que vous n'avez pas à écrire de tests pour cette fonctionnalité, car vous ne savez pas encore comment procéder pour écrire des tests d'interface. Par contre, rien ne vous empêche de faire une petite recherche et d'écrire ces tests quand même (ce qui n'est pas si difficile que cela en fait).

3.9 Finaliser l'application

Voici vos « user stories », vos critères d'acceptabilité et vos directives techniques pour ce travail.

3.9.1 User Stories

No	Description	OK
1	En tant qu'utilisateur, je veux pouvoir saisir une formule chimique et en obtenir le poids en g/mol.	
2	En tant qu'utilisateur, je veux être notifié de toute erreur de syntaxe que j'aurais pu commettre afin de pouvoir la corriger.	

3.9.2 Critères d'acceptabilité de la conception

No	Description	OK
1	L'interface console doit être en français.	
2	L'interface Android doit être en français et en anglais.	
3	Les tests d'acceptation fournis avec les projets doivent passer.	
4	L'interface console doit être identique à ce qui est demandé.	
5	L'interface Android doit être conforme aux principes des plateformes mobiles.	
6	Aucune donnée ne doit être perdue après un changement d'orientation.	

3.9.3 Directives techniques

No	Description	OK
1	L'application doit utiliser le style architectural « MVC ».	
2	Le package de l'application doit être « ca.csf.mobile1.tp1 ».	
3	Les messages d'erreur sous Android doivent utiliser « Snackbar ».	
4	Vous devez utiliser un dépôt de code sous Git et en donner l'accès à votre professeur.	
5	Vous devez placer les « User Stories » dans la section « Issues » de votre dépôt Git.	

4 Modalités de remise

Remettez sur LÉA à l'heure et à la date indiquée par votre professeur un fichier texte comportant l'adresse vers vos dépôts Git. Ces adresses seront utilisées pour effectuer un « git clone ». Ajoutez à ce fichier votre nom, votre matricule ainsi que tout commentaire pertinent à la correction. Enfin, le nom du fichier texte doit être votre matricule suivi de « .txt ».

Une seule journée de retard est tolérée entraînant alors une pénalité de 15 % de la note. Au-delà de ce délai, le travail est refusé et la note « 0 » est attribuée.

5 Évaluation

Ce travail sera évalué avec une correction normale et une correction négative. Les points seront retirés en fonction de la gravité de la faute, de la qualité globale du travail et de l'effort mis par l'étudiant. Le tableau suivant vise à résumer les éléments pouvant réduire la note, jusqu'à atteindre la note de 0 %.

Éléments		Pondération
Fonctionnalités :		5 % Correction normale
• Support des symboles d'éléments chimiques dans les formules.	(0176-1, 0176-2)	
• Support des parenthèses dans les formules.	(0176-1, 0176-2)	
• Support des exposants dans les formules.	(0176-1, 0176-2)	
• Exception lancée lorsqu'une parenthèse vide est rencontrée.	(0176-1, 0176-2)	
• Exception lancée lorsqu'un caractère illégal est rencontré.	(0176-1, 0176-2)	
• Exception lancée lorsqu'une parenthèse illégale est rencontrée.	(0176-1, 0176-2)	
• Exception lancée lorsqu'une parenthèse non fermée est rencontrée.	(0176-1, 0176-2)	
• Exception lancée lorsqu'un exposant mal placé est rencontré.	(0176-1, 0176-2)	
• Exception lancée lorsqu'un élément chimique inconnu est rencontré.	(0176-1, 0176-2)	
• En mesure de saisir une formule chimique dans l'application.	(0176-1, 0176-2)	
• En mesure de calculer le poids de la formule chimique saisie.	(0176-1, 0176-2)	
• Affichage de messages d'erreur s'il y a erreur dans la formule chimique.	(0176-1, 0176-2)	
Interface utilisateur (incluant, mais sans s'y limiter) :		15 % Correction négative
• Interface console malpropre.	(017D-4)	
• Non-respect de la grille de 8dp sous Android.	(017D-4)	
• Éléments d'interface mal alignés sous Android.	(017D-4)	
• Non-utilisation de « Snackbar » pour les messages sous Android.	(017D-4)	
• Fichier XML de « Layout » Android malpropre.	(017D-4)	
• « Layouts » Android instables sous différentes tailles et densités d'écran.	(017D-4)	
• Expérience utilisateur désagréable sous Android.	(017D-4)	
• Perte de données lors d'un changement d'orientation.	(017D-3, 017D-4)	
Documentation du code (incluant, mais sans s'y limiter) :		10 % Correction négative
• Classe non documentée ou documentée de manière insuffisante.	(017D-7)	
• Méthode non documentée ou documentée de manière insuffisante.	(017D-7)	
• Paramètres non documenté ou documenté de manière insuffisante.	(017D-7)	
• Valeur de retour non documenté ou documenté de manière insuffisante.	(017D-7)	
• Lien non présent vers une classe référencé dans la documentation.	(017D-7)	
• Documentation malpropre.	(017D-7)	
• Indentation incorrecte de la documentation.	(017D-7)	
• Documentation qui n'est pas en français.	(017D-7)	
• Standard de documentation « JavaDoc » non respecté.	(017D-7)	

Tests unitaires (incluant, mais sans s'y limiter) : • Altération d'un test unitaire en dehors de ce qui est permis. (0176-4, 0177-3) • Suppression d'un test unitaire sans autorisation. (0176-4, 0177-3) • Vérification échouant dans un test unitaire. (0177-1, 0177-2, 0177-3) • Classe publique non testée. (0176-4, 0177-1) • Méthode publique non testée. (0176-4, 0177-1) • Classe de test qui n'est pas dans le dossier « src/test/java ». (0177-2, 0177-3) • Vérification manquante dans un test. (0176-4, 0177-1) • Test vérifiant plus qu'une fonctionnalité. (0177-1) • Utilisation de la mauvaise vérification « JUnit ». (0177-2)	10 % Correction négative
Qualité générale de l'architecture (incluant, mais sans s'y limiter) : • Contrôleur effectuant de la logique réservée à la couche modèle. (017D-3, 017D-5) • Séparation incorrecte entre la vue et le modèle. (017D-3, 017D-5) • Abstractions non utilisées où cela aurait été nécessaire. (016T-1) • Référence concrète où une référence abstraite aurait été suffisante. (016T-3) • Pas d'usage de l'héritage où cela aurait été nécessaire. (016T-1, 016T-3) • Non-respect du diagramme de classe. (016T-2) • Méthode ayant trop de responsabilités pas découpées convenablement. (016T-3) • Classe ayant trop de responsabilités pas découpées convenablement. (016T-3) • Découpage en « package » insuffisant. (016T-2) • Nom d'un « package » imprécis. (016T-3) • « dot.case » non utilisé pour un nom de « package ». (016T-3)	30 % Correction négative
Qualité générale du code (incluant, mais sans s'y limiter) : • Logique applicative « patchée ». (016T-3) • Nom d'une variable imprécis. (016T-3) • Nom d'une classe imprécis. (016T-3) • Nom d'une interface imprécis. (016T-3) • Nom d'une constante imprécis. (016T-3) • Nom d'une énumération imprécis. (016T-3) • Nom d'une méthode imprécis. (016T-3) • « camelCase » non utilisé pour un nom de variable ou de méthode. (016T-3) • « PascalCase » non utilisé pour un nom d'interface, de classe ou d'énumération. (016T-3) • « UPPER_CASE » non utilisé pour un nom de constante. (016T-3) • Manque d'une constante ou cela aurait été nécessaire. (016T-3) • Méthode sans attribut de visibilité. (016T-3) • Méthode publique qui devrait être privée. (016T-3) • Méthode statique qui ne devrait pas l'être. (016T-3) • Méthode protégée qui ne devrait pas l'être. (016T-3) • Attribut de classe qui n'est utilisé que dans une seule méthode. (016T-3) • Commentaire inutile ne compensant pas le manque d'expressivité du code. (016T-3) • Erreur à la compilation. (016T-3, 016T-5) • Avertissement à la compilation. (016T-3, 016T-5)	20 % Correction négative
Qualité générale du travail (incluant, mais sans s'y limiter) : • Corruption de la configuration Gradle des projets. (016T-5) • Non-respect des consignes de remise. (016T-5) • Fichiers inutiles remis avec les projets. (016T-5) • Travail non remis via l'outil gestion de code source Git. (0176-3)	10 % Correction négative

Notez que la qualité écrite de la langue française fait partie intégrante de l'évaluation de ce travail. Étant donné que l'accès aux ressources de la langue française est permis, chaque faute de français retirera donc 0,5 % à la note finale jusqu'à concurrence de 20 % retirés. Sont corrigées les fautes d'orthographe, les fautes de grammaire, les fautes syntaxiques et les fautes de ponctuation.

6 Compétences reliées au travail pratique

Voici la liste des compétences évaluées, évaluées partiellement ou non-évaluées par ce travail. Vous trouverez les explications sur ces choix en dessous de cette liste.

- 016T-1 Créer un modèle objet : Évalué.
- 016T-2 Perfectionner le modèle objet : Évalué.
- 016T-3 Procéder à la codification d'une classe : Évalué.
- 016T-5 Générer la version exécutable du programme : Évalué.
- 0176-1 Analyser les caractéristiques de l'application : Évalué.
- 0176-2 Analyser la nature des améliorations à apporter à l'application : Évalué.
- 0176-3 Ajouter et modifier des fonctionnalités à une application : Évalué.
- 0176-4 Valider le fonctionnement de l'application : Évalué.
- 0177-1 Planifier les tests : Évalué.
- 0177-2 Procéder à l'exécution des différents tests : Évalué.
- 0177-3 Valider la qualité de l'application : Évalué.
- 017D-2 Établir le cadre technologique : Non-évalué.
- 017D-3 Préparer le travail de développement : Évalué.
- 017D-4 Produire le prototype de présentation : Évalué.
- 017D-5 Produire le prototype de communication : Évalué.
- 017D-6 Développer l'application : Non-évalué.
- 017D-7 Produire la documentation relative à l'application : Évalué.

La compétence 017D-2 sera évaluée au travail pratique 2 lorsque l'étudiant sera amené à effectuer des requêtes à un service web à partir de son application mobile. La compétence 017D-6 sera évaluée au travail pratique 3 lorsque l'étudiant devra utiliser une base de données relationnelle pour son application mobile.