LO21 - Printemps 2015 - Projet.

ProjectCalendar

Dans ce projet on doit développer l'application `ProjectCalendar` destinée à mixer des fonctionnalités d'un agenda électronique traditionnel et d'un outil de gestion de projet.

Fonctions principales

1. Une partie de l'application doit permettre d'afficher l'emploi du temps hebdommadaire de l'utilisateur, c'est à dire une vue synthétique des 7 jours d'une semaine particulière avec les évènements qui y ont été programmés.

2. Une autre partie de l'application doit permettre à l'utilisateur de renseigner l'ensemble des projets qu'il a en charge. Un projet est défini comme un ensemble de tâches à effectuer. Une tâche ne peut être associée qu'à un seul projet. Une tâche peut être une simple tâche (on parle de *tâche unitaire*) ou représenter un ensemble de sous-tâches (on parle de *tâche composite*). Chaque sous-tâche peut à son tour être une tâche unitaire ou composite. Une tâche i (unitaire ou composite) est caractérisée par un titre (permettant d'en saisir la nature) et par un ensemble de tâches (unitaires ou composites) qui doivent avoir été terminées avant de commencer la tâche i (on parle de *contraintes de précédence*). Une tâche unitaire est aussi caractérisée par une durée. Certaines tâches unitaires peuvent être préemptées (on peut commencer la tâche, l'interrompre, et la reprendre plus tard), alors que d'autres ne peuvent pas l'être (une fois commencées, elles doivent être exécutées jusqu'à leur fin). Une tâche qui peut être préemptée peut l'être plusieurs fois. La durée d'une tâche unitaire qui ne peut pas être préemptée ne peut pas excéder 12h. Un projet ou une tâche (composite ou unitaire) peut aussi avoir une date de disponibilité (avant laquelle on ne peut pas commencer le projet ou la tâche) et une échéance (avant laquelle le projet ou la tâche doit être terminé). L'échéance d'un projet est une borne supérieure de l'ensemble des échéances des tâches de ce projet.

3. L'utilisateur doit pouvoir ajouter des évènements dans son agenda. Un évènement est soit la programmation d'une activité traditionnelle (dont il faudra déterminer les caractéristiques) comme un rendez-vous ou une réunion, soit la programmation d'une tâche (unitaire) liée à un projet. Dans le cas d'une tâche unitaire qui peut être préemptée, cela peut aussi être la programmation d'une partie seulement de cette tâche (l'autre partie restant à être programmée plus tard). Dans le cas de la programmation d'une tâche, l'application doit interdire une programmation qui ne respecterait pas l'ensemble des contraintes liées à cette tâche (contraintes de disponibilité, d'échéance, de précédence). Dans cette application, il n'est pas possible de programmer deux évènements en même temps.

4. Dans la partie liée à la gestion des projets, l'utilisateur doit pouvoir aisément repérer ou choisir les tâches unitaires qui peuvent (et doivent encore) être ordonnancées lorsqu'il cherche à ajouter des événements dans son agenda. Les dépendances entre tâches unitaires, tâches composites et projets doivent être clairement affichées (par ex avec un tree-view).

5. L'application doit permettre l'export des programmations d'une semaine particulière de l'agenda ou l'export de l'ensemble des programmations liées à un projet particulier. On se contentera d'exports dans des fichiers texte brut et dans des fichiers XML. Néanmoins, il faudra s'assurer que l'application soit facilement maintenable de façon à ce que d'autres exports soient possibles dans le futur. Par exemple une synchronisation avec un calendrier en ligne, ou l'export sous forme d'un diagramme de Gantt d'un projet, etc.

Livrable attendu

Le livrable est composé des éléments suivants :

- **Code source :** l'ensemble du code source du projet. Attention, ne pas fournir d'exécutable ou de fichier objet.
- **Documentation** : une documentation complète en `html` générée avec `Doxygen`.
- **Video de présentation** : une courte video de présentation dans laquelle vous filmerez votre logiciel afin d'en démontrer le bon fonctionnement (maximum 5 min). Pour réaliser cette video, vous pourrez vous servir des logiciels CamStudio (Windows), Jing (Windows, Mac OS), RecordMyDesktop (Linux). Ces logiciels sont mentionnés uniquement à titre d'exemple.
- **Rapport :** Un rapport en format `.pdf` composé de 2 parties :

- la description de votre architecture;
- une argumentation détaillée où vous montrez que votre architecture permet facilement des évolutions.

Vous pouvez ajouter en annexe de ce rapport des instructions à destination de votre correcteur si nécessaire (présentation des livrables, instructions de compilation, ...). Ce rapport ne devra pas dépasser 10 pages (annexes comprises).

L'ensemble des livrables est **à rendre pour le dimanche 14 juin 23h59 (fuseau horaire de Paris) au plus tard**. Les éléments du livrable doivent être rassemblés dans une archive `.zip`. L'archive doit être envoyée par mail aux chargés de TD suivant votre séance :

- mardi matin : Ronan Bocquillon (`ronan.bocquillon@utc.fr`).
- mardi après midi : Taha Arbaoui (`taha.arbaoui@utc.fr`).
- mercredi matin : Sohaib Afifi (`sohaib.afifi@utc.fr`).
- jeudi matin : Antoine Jouglet (`antoine.jouglet@utc.fr`).
- jeudi après midi : Ahmed Lounis (`ahmed.lounis@utc.fr`).
- vendredi après midi : Lucile Callebert (`lucile.callebert@utc.fr`).

Évaluation

Le barême de l'évaluation du projet est comme suit :

- **Couverture des fonctionnalités demandées** : 5 points
- **Choix de conception et architecture** : 5 points. En particulier sera évaluée la capacité de l'architecture à s'adapter aux changements.
- **Evaluation des livrables** : 6 points (code source, documentation, vidéo, exemples de fichiers, rapport)
- **Respect des consignes sur les livrables** : 2 points (échéance, présence de l'ensemble des livrables et respect des consignes sur les livrables).
- **Présence** aux séances de TD : 2 points

Remarque : il est rappellé qu'une note inférieure ou égale à 6/20 au projet est éliminatoire pour l'obtention de l'UV.

Consignes

- Le projet est à effectuer en binôme (un seul monôme ou trinôme toléré par groupe de TD).
- Vous êtes libres de réutiliser et modifier les classes déjà élaborées en TD pour les adapter à votre architecture.
- En plus des instructions standards du `C++/C++11`, vous pouvez utiliser l'ensemble des bibliothèques standards du `C++/C++11` et le framework `Qt` à votre convenance.
- Il n'y a pas de contraintes concernant les éléments d'interface (à part le fait d'avoir une vue hebdomadaire de l'agenda) et d'ergonomie. Soyez créatifs. Il devrait y avoir autant d'interfaces différentes que de binômes.

Conseils

- La partie difficile du projet est la conception de votre architecture : c'est dessus qu'il faut concentrer vos efforts et passer le plus de temps.
- Il est conseillé d'étudier au moins les design patterns suivants qui pourraient être utiles pour élaborer l'architecture de votre projet : decorator, factory, abstract factory, builder, brigde, composite, iterator, template method, adapter, visitor, strategy, facade, memento. En plus de vous donner des idées de conception, cette étude vous permettra de vous approprier les principaux modèles de conception.
- Pour la persistance des informations, vous êtes libre d'élaborer vos formats de fichier. Il est tout de même conseillé d'utiliser XML (comme pour l'export) et d'utiliser les outils XML de `Qt`.
- Au lieu d'utiliser des fichiers, vous pouvez utiliser un SGBD comme SQLite.
- Le design de l'application ne sera pas pris en compte dans la notation. Soyez avant tout fonctionnels.