BâtiBloc

1. Consignes à propos du projet

- Le travail doit être réalisé en équipe de 5.
- Votre équipe doit être constituée sur le portail des cours avant la date limite prévue à cet effet.
- Le projet doit être réalisé en Java avec l'environnement de développement NetBeans (gratuit) ou IntelliJ IDEA Community (gratuit). Chacun a ses avantages et inconvénients : coder est généralement plus agréable sur IntelliJ, mais la conception de l'interface utilisateur est plus simple et intuitive sur Netbeans. Le choix vous revient donc de prendre celui qui vous convient le mieux (il est également possible d'utiliser les deux, mais sachez que l'édition des interfaces utilisateurs n'est pas compatible d'un à l'autre).
- Un dépôt Git **vous sera attribué** au début de la session, généralement une semaine avant la date de remise du livrable 1. Vous devez **absolument** l'utiliser, car vos travaux y seront récupérés automatiquement via un script pour la correction.
    - Si vous utilisez un autre dépôt que celui attribué, la note de 0 vous sera automatiquement attribuée.
- Il est primordial que chacun utilise un compte associé à son vrai prénom et nom et que l'adresse de courriel ulaval.ca y soit associée pour que l'on sache qui travaille sur quoi.
    - Chacun doit déposer lui-même son propre code dans le dépôt Git au fur et à mesure que le projet avance étant donné que nous évaluons les contributions individuelles (voir les notes à ce sujet dans le plan de cours)
    - Une contribution équitable de chaque membre est demandée **au niveau du code** puisque l'apprentissage autonome du langage Java fait partie des objectifs du cours (voir plan de cours au sujet de l'évaluation individuelle). Comme expliqué dans le plan de cours, la note individuelle sera influencée par l'évaluation par les pairs et par l'évaluation faite par l'enseignant de votre contribution au travail d'équipe. Notamment, pour les livrables 3, 4 et 5 nous ferons usage d'outils statistiques pour évaluer votre contribution individuelle au code de l'application. Vous ne pouvez donc pas faire plus d'analyse (ou plus de travail sur le rapport) pour compenser un manque en programmation. **Vous pourriez perdre jusqu'à 75% des points de chaque livrable.**
- Il est interdit de faire des *marges* de branches tout au long de la session (car cela fausse l'analyse du code). Vous devrez donc fusionner vos branches en utilisant l'opération *rebase*. Il est donc fortement recommandé de faire des petites branches pour simplifier l'opération.
    - L'énoncé de chaque remise spécifiera comment remettre le travail (en créant une nouvelle branche spécifique sur le GIT).
    - Dans GitLab vous pouvez tout de même faire des Merge Request, **mais vous devrez vous assurer que l'option « squash commits » est désactivée.**
- L'analyse du code sera effectuée sur la branche main, assurez-vous donc de ramener fréquemment votre code sur cette branche.

- Procédez de manière incrémentale plutôt que de souhaiter qu'à la dernière minute toutes vos contributions au code puissent se connecter par magie.
- L'utilisation de toute autre librairie que les librairies standards de Java est interdite.
- Vous devez utiliser la librairie Swing pour faire la programmation de votre interface.
- Vous devez utiliser Java 25 pour ce projet.
- Les diagrammes UML **doivent** être produits avec le logiciel Visual Paradigm Community Edition (https://www.visual-paradigm.com/download/community.jsp).

Choses à faire pour augmenter la probabilité d'échouer le projet

- Oublier que pour un cours universitaire de trois crédits la charge de travail "normale" est de 9 heures par semaines pendant 15 semaines, incluant la semaine de lecture et les semaines d'examen, pour un total de 135 heures.
- Chaque semaine où on ne le fait pas suppose qu'on a 9 heures à rattraper plus tard.
- Oublier que pour un.e étudiant.e avec plus de difficultés ou plus méticuleux, il se peut qu'un cours vous demande plus de 9 heures par semaine et donc plus de 135 heures.
- Laisser les autres avancer au début en vous disant que "vous contribuerez plus plus tard ou à la fin". Erreur, c'est un projet qui demande une implication soutenue de tous les membres de l'équipe durant toute la session.
- Confondre "projet de session" et TP. Ceci n'est pas un TP qui peut se faire en quelques jours avant la remise. C'est un projet de session.
- Ne pas apprendre le Java (une contribution à part égale au CODE est exigée de tou.te.s et chacun.e).
- Ne pas apprendre à utiliser GIT.
- Mettre des trucs dans GIT en utilisant le compte d'un ami ou collègue (ce sera comptabilisé comme une contribution de la part de cet ami).
- Bien que la *pair programming* soit une bonne pratique sur le marché du Travail, le faire dans ce cours peut rapidement devenir un problème pour évaluer votre code.
- Prétendre faussement qu'un collègue a fait sa juste part lors de livrables #1 et #2 en croyant qu'il allait se rattraper plus tard. Ce genre de situations ne se corrige jamais et toute l'équipe est plus tard alors en péril.
- Jouer les super-héros (essayer de tout faire vous-même) parce que vous êtes vraiment bon en programmation. On essaie de vous enseigner le travail d'équipe: si vous êtes particulièrement doué on vous demande de jouer un rôle de leader, de former vos collègues et de les rendre meilleurs. Pas de faire le projet à leur place (ce ne serait alors plus un projet d'équipe et toute l'équipe se retrouvera en situation d'échec).
- Ne pas tester suffisamment votre application.
- Vouloir tout apprendre en même temps et appliquer tous les trucs à la mode; appliquer tous les patterns que vous connaissez, utiliser tous les outils technos en même temps. Trop c'est trop. Commencez par la base.
- Faire un mauvais design, partir tout croche et passer 3 semaines à (essayer de) déboguer à la fin de la session.
- Utiliser des coordonnées, distances, etc. en pixel au lieu d'utiliser des nombres réels et espérer qu'il n'y aura pas d'erreurs numériques à la fin lors de la conversion (il va y en avoir c'est certain).

- Ne pas avoir compris que la note de passage est de 60 %
- Ne pas avoir compris que le cours est à double seuil.
- Ne pas avoir lu le plan de cours.
- Ne pas avoir consulté cette liste.
- Ne pas suivre, pas prendre de notes ou ne pas écouter pendant les séances de cours.
- Ne pas être présent.e lors des TD d'équipe le vendredi ou de toutes rencontres d'équipe

2. Mise en contexte

Dans les dernières années, les méthodes de constructions de bâtiments s'éloignent de plus en plus des constructions classiques, basées sur des armatures en 2x3 ou autres montées sur le chantier. Différentes méthodes innovantes ont émergé dans les dernières années. Par exemple, Sokïo produit des murs en bois laminé croisé qui peuvent être préfabriqués en usine. Il ne reste ensuite qu'à les assembler sur le chantier, permettant ainsi de construire des maisons bien plus rapidement qu'avec la méthode classique.

Dans le cadre de ce projet, nous explorerons une méthode alternative à la construction traditionnelle basée sur l'installation de blocs inspirés des blocs LEGO. Le principal intérêt de cette méthode étant la présence d'isolant sur le bloc ainsi que sa rapidité d'installation. Voici une image d'un tel bloc :

De plus, cette méthode peut être hybridée avec des méthodes de construction traditionnelle utilisant des armatures en bois, ce qui la rend très flexible. Les blocs s'emboîtent parfaitement pour former des murs étanches et isolés (R30).

Votre mission, si vous l'acceptez, consiste à créer un logiciel qui permet de simuler la construction d'un bâtiment à partir d'un tel système à partir de plans PDF importés ! L'objectif principal étant de calculer le nombre de blocs nécessaires ainsi que de simuler leur placement dans le plan.

3. Importation d'un plan

La première étape consiste à importer un fichier PDF représentant le plan du bâtiment à construire. Ces fichiers ont généralement plusieurs pages représentant différentes vues du bâtiment ainsi que plusieurs détails techniques. Initialement, chaque page du PDF deviendra une vue différente de l'application. Dans chaque vue l'image de la page PDF correspondante sera affichée comme image de fond.

L'utilisateur aura ensuite l'option du supprimer certaines vues (par exemple la page titre ou des vues qui ne représentent pas une façade du bâtiment). Un bâtiment typique ayant 4 façades, l'importation d'un plan donnera généralement 4 vues au final à la suite de la suppression des vues inutiles. L'utilisateur pourra également rogner (*crop)* l'image pour conserver seulement la portion du PDF correspondant au bâtiment. Le résultat sera généralement 4 vues des 4 façades du bâtiment, mais certains bâtiments plus complexes pourraient en avoir plus de 4.

L'utilisateur devra également entrer l'échelle du plan (qui peut varier par vue), car cette échelle sera nécessaire plus tard.

4. Modélisation des vues

Une fois le plan importé, l'utilisateur pourra identifier différentes zones sur l'image qui représenteront les portions de murs que l'on souhaite construire à l'aide de la technique des blocs. Il y a trois types de zones : a) les zones correspondant à une armature classique, b) les zones pour les armatures en blocs et c) les zones correspondant à des ouvertures (portes et fenêtres). Les zones sont créées directement avec la souris.

Ces zones auront trois formes possibles : des rectangles, des triangles ou des triangles tronqués (triangle dont on a coupé le sommet). Cette dernière forme étant particulièrement utile pour représenter le toit du bâtiment (car les blocs ne se rendent pas au sommet). La taille réelle des zones sera déterminée automatiquement à partir de l'échelle fournie, tandis que l'utilisateur pourra modifier la taille à l'écran comme il le désire à l'aide de la souris.

5. Calcul du nombre de blocs nécessaires

Une fois satisfait des zones définies, l'utilisateur pourra ensuite calculer le nombre de blocs nécessaire pour remplir les zones correspondantes. Par défaut, un bloc a les dimensions suivantes : 8' de long par 12'' de haut, mais l'utilisateur pourra configurer ces deux valeurs.

Le placement suit les contraintes suivantes :

1. On démarre de la gauche vers la droite avec un bloc complet, puis on met des blocs complets jusqu'à remplir la rangée complète

2. Si deux zones se touchent et qu'il est possible de placer un bloc complet en hauteur, la rangée continue dans la zone suivante (il y a donc possibilité d'avoir un bloc dans deux zones en même temps)
3. Lorsque l'on termine une rangée, le bloc est (possiblement) coupé. Le restant du bloc (ce qui dépassait) est utilisé pour démarrer la prochaine rangée au-dessus.
4. La hauteur des zones est automatiquement ajustée à la hausse pour contenir un nombre entier de blocs
    - Un avertissement est mis dans une zone dédiée de l'interface pour notifier l'utilisateur de ce changement
5. Les ouvertures causent des comportements spéciaux
    - Les blocs à gauche et à droite d'une ouverture doivent être d'au moins 6 pouces (ce qui peut influencer la taille du bloc initial mentionné en 1.)
    - La hauteur des ouvertures est ajustée automatiquement pour contenir un nombre entier de blocs (comme en 4, avec les mêmes règles)
    - Les ouvertures ont toujours un bloc complet au-dessus, il agira comme un linteau (qu'on nommera bloc linteau)
        - Ce bloc linteau doit dépasser d'au moins 6 pouces de chaque côté de l'ouverture. Il sera toujours centré sur l'ouverture (dépasse de la même distance de chaque côté).
            - Note : avec la taille par défaut des blocs cela implique une taille d'ouverture maximale de 7'
    - Si l'ouverture dépasse la taille maximale, une armature traditionnelle est nécessaire. Considérez que sa fabrication réduit l'espace jusqu'auquel se rendent les blocs. On arrête donc tous les blocs à 4 pouces de l'ouverture de chaque côté, et on laisse un espace d'un bloc au-dessus de l'ouverture (pour placer un linteau).
        - La rangée d'après continuera avec les blocs normalement
        - Des zones d'armatures classiques seront créées automatiquement autour de l'ouverture et supprimées automatiquement pour représenter cela. Elles seront supprimées automatiquement si la taille redevient plus petite que la taille maximale.
6. Contraintes spéciales pour les zones triangulaires et tronquées
    - On coupe automatiquement les côtés de chaque bloc aux extrémités pour suivre la forme du triangle.
        - Contrairement aux zones rectangulaires, il ne sera pas toujours possible de réutiliser les blocs coupés pour démarrer une nouvelle rangée
    - On n'ajustera pas la taille des zones pour mettre un nombre entier de blocs, mais on arrêtera plutôt lorsque ce ne sera plus possible.
7. Un bloc a une longueur minimale de 6 pouces
    - La longueur du bloc initial peut donc être modifiée automatiquement pour respecter cette contrainte

Le logiciel déterminera automatiquement le placement des blocs pour remplir toutes les zones. L'objectif est de déterminer le nombre exact de blocs qui seront nécessaires pour remplir toutes les zones avec blocs tout en respectant toutes les contraintes décrites plus tôt. Finalement, l'utilisateur pourra entrer un prix par bloc et le logiciel calculera automatiquement une estimation du coût total du projet. Cette estimation pourra être visualisée par vue ainsi que globalement.

6. Dernières petites choses

- L'interface doit être conviviale.
- Il faut permettre à l'utilisateur de sauvegarder dans des fichiers en vue d'une réouverture et utilisation future.
- Undo/redo (minimum 9999999 opérations).
- Zoomer/dézoomer **à l'infini** (en utilisant la roulette de la souris). **Important : le zoom se fait par rapport à la position du curseur de la souris.**
- Le logiciel peut exporter un fichier PNG des différentes vues ainsi qu'un fichier PNG combinant toutes les vues.
- Lorsque la souris passe au-dessus d'un point quelconque, on affiche les coordonnées dans une zone dédiée de l'interface.
- L'utilisation de fenêtres flottantes (popup) est interdite, à l'exception de la fenêtre d'ouverture/enregistrement d'un fichier, de messages d'erreur ou encore une fenêtre pour la sélection d'une couleur (advenant que vous en ayez besoin). En cas de doute, référez-vous au client, mais assurez-vous de ne pas le décevoir – votre client **déteste** les popups et vous serez pénalisé en cas d'utilisation injustifiée d'un popup.
- Les valeurs des différents paramètres des éléments sélectionnés sont modifiables à l'aide d'un « panel » (typiquement situé à gauche ou à droite de la fenêtre du logiciel).
- L'édition ne doit pas être séquentielle. En tout temps, je peux éditer n'importe quel élément de mon projet (notamment modifier la taille d'une zone, ajouter une zone d'ouverture, etc.).
- L'usage de toute librairie externe est interdit sans l'autorisation d'un enseignant.
    - Voici la liste des librairies autorisées :
        - JUnit
        - Google Truth
        - Mockito
        - FlatLaf
        - PDFBox
        - Batik
        - IntelliJ UI Designer
- L'utilisation d'un trackpad/touchpad/magicpad/padthaï à la place d'une souris est interdite lors des démonstrations/évaluations. Selon nos statistiques cette consigne nous fait collectivement sauver plusieurs heures. Contrairement à la croyance populaire, un trackpad c'est vraiment lent pour faire des dessins précis.

Prix Yves-Roy

La meilleure application[1] remportera le prix et la bourse Yves-Roy à titre de « meilleur projet départemental en génie logiciel orienté-objet ». Il s'agirait là d'une réalisation digne de mention sur votre curriculum vitae!

Vous êtes bien sûr encouragés à ajouter des fonctionnalités supplémentaires à votre application si vous le souhaitez. Voici quelques suggestions :

- Vue 3D
- Personnalisation du placement des blocs, principalement la taille du premier bloc
- Affichage des armatures traditionnelles et calcul du nombre de pièces de bois nécessaires + calcul du prix
- Gestion des vues de dessus pour la construction de murs mitoyens
- Etc.

Remarques

Certains éléments du descriptif de projet sont naturellement flous à ce stade (si nous vous transmettions des spécifications parfaites accompagnées de diagrammes UML… vous n'auriez pas à faire l'analyse et ce ne serait plus un projet complet). Il vous appartient de faire la lumière là-dessus et de développer une bonne compréhension du projet. Vous serez appelés à poser des questions au **client** (un de vos enseignants) au début de chaque cours tout au long du projet. N'hésitez pas à poser des questions.

Notez bien que certains éléments peuvent changer au courant de la session pour refléter exactement les besoins du client. Des versions mises à jour du présent document pourront être publiées par le client. Surveillez également les informations supplémentaires qui pourront apparaître dans les documents distincts associés à chaque livrable.

Vous disposez également de **coachs** (auxiliaires d'enseignement). Vous pouvez les considérer comme des employés séniors de votre entreprise qui vous donneraient un coup de main.

Consultez également la section « Méthodologie » du plan de cours pour les différentes façons d'obtenir de l'aide.

Travaillez fort et amusez-vous bien!

[1] Les enseignants et auxiliaires du cours voteront pour l'application dans laquelle ils investiraient s'ils devaient créer une entreprise. Nous tiendrons donc compte de l'expérience utilisateur, mais aussi de la qualité de la conception, du code, etc. Dans le cas où un gagnant ne pourrait être sélectionné de manière unanime, nous nous réservons le droit d'organiser un vote en faisant appel à un jury externe.