LogBook

Auteur	Date de début	Projet	Description	Communication
Joey Martig	19.04.2021	Travail de diplome Covid propagation	Applications permettant de voir l'évolution d'un virus dans un environnement peuplé d'individus.	Joey.mrtg@eduge.ch Joey.mrtg@gmail.com

19.04.2021 - 08h05/17h00

- Rendu appréciation du travail de stage.
- Démarrage du travail de diplôme et présentation moodle.
- Documentation
  - Intégration CDC
  - Introduction
  - Résumé
  - Planning prévisionnel
- Comparaison WPF et winforms
  - Référence 1
    - Peu de différence autre que la structure
    - WPF semble plus récent et puissant
  - Référence 2
    - Interface GUI plus performante (très intéressant)
    - Plus récent (Plus de librairies mises à jour)
  - Référence 3
    - WPF est plus rapide
    - Aussi plus complexe
  - Référence 4
    - Les deux se valent
    - WPF est meilleur pour les UI
    - WPF n'est pas disponible pour linux et mac
    - Le data binding est meilleur en WPF ainsi que le design
  - Conclusion
    - L'apprentissage de la structure WPF va prendre du temps, mais semble en valoir la peine.
    - L'UI sera plus belle et simple à réaliser en WPF
    - Le plus gros avantage semble être BEAUCOUP plus efficace pour l'affichage de l'interface graphique me permettant d'utiliser le GPU à pleine puissance contrairement au winform qui ne l'utilise pas. Sachant que les pc sont équipés de gtx 1060, le changement en WPF semble être le meilleur choix.
    - WPF semble donc être le meilleur choix.
  - Une autre alternative à l'utilisation de l'interface graphique de WPF est Unity qui peut être intégré et communiquer avec le projet. Intégration d'Unity en WPF. Communication. Génération dynamique

    - Unity semble être une bonne idée cependant, dans mon cas, en prenant en compte le nombre d'appelle, utiliser Unity et faire communiquer les deux projets me semblent trop compliqué. Peut-être que rester sur WPF est plus sûr.
- Documentation
    - Analyse interface graphique
        - Comparaison WPF - WinForms - Unity
        - Choix de technologie
- Tentative de communication réussie à l'aide de ce tutoriel.
    - Inutilisable pour ce projet, car la communication est trop restreinte.
    - La communication se fait uniquement avec des string ou des images, mais pas d'objets c#.
- Tentative d'intégration d'Unity dans un projet WPF
    - Réussi
    - Problème de resize
        - Au lancement prend toute la fenêtre
        - Le resize et positionnement fonctionnent
        - À la fermeture du programme, Unity ne s'arrêtait pas et utilisait 15% du processeur à chaque ouverture.

20.04.2021 - 08h05/17h00

- Rétrospective 19.04.2021
- Recherche des méthodes dans la dll user32.dll
- Recherche de différents moyens de communication entre WPF et Unity
    - Essai d'envois de données de WPF à Unity avec un pipeline anonyme
    - Essai d'envois de données de WPF à Unity avec un pipeline nommé
        - Réussi entre Unity et WPF, mais inutilisable actuellement
        - Il faut stopper le programme pour lire le message
        - Peut-être un problème lié au fait que les deux programmes essaient de lire au même moment.
- Problèmes d'intelliSense sur Unity
    - la connexion entre le script vs 2019 et Unity n'est pas établie.
    - Mise à jour de vs 2019 et installation des paquets Unity
    - Redémarrage du pc
    - Modifications de paramètre Unity et vs sans succès
    - Recherche de solutions concernant vs 2019 et non 2015
    - Pas suffisant. Ne fonctionne toujours pas.
    - Le point e. à reglé le problème
- Problème avec Unity qui supprime mes objets à chaque fermeture du projet
    - La scène avait été supprimée et ne s'était pas rajoutée automatiquement
- Communication fonctionnelle, mais le projet Unity freeze après chaque message.

- Tentative d'utilisation de l'asynchrone pour pallier au problème
  - Ce problème est réglé cependant un problème de décalage fait que WPF parle chinois.
  - 

  - Essai de lecture de byte pour obtenir le résultat. sans succès
  - Refactoring du code pour fonctionner en async.
  - Le code est toujours imparfait et décalé
    - "Test communication" => "Test communication"
    - "Test communication" => "est communication"
    - "Test communication" => Signes chinois
- Communication en string fonctionnelle à 100% en utilisant une méthode async récursive appelée au démarrage du projet Unity.
  - Méthode appelée chaque seconde ayant des résultats très aléatoires.
  - ```
    void Start()
    {
        // Code
        InvokeRepeating("ReadPipeData", 0, 1f);
    }

    0 références
    private async void ReadPipeData()
    {
        string result = await ss.ReadStringAsync();
        ChangingText.GetComponent<Text>().text = result;
    }
    ```
  - Méthode appelée une fois au démarrage puis à chaque fois qu'elle termine l'affichage de données reçues.
  - ```
    void Start()
    {
        //Code
        ReadPipeData();
    }
    private async void ReadPipeData()
    {
        string result = await ss.ReadStringAsync();
        ChangingText.GetComponent<Text>().text = result;
        ReadPipeData();
    }
    ```
- Idée pour l'implémentation en Unity : Envoyer des objets en JSON via le pipeline.
- Test d'envoi d'un objet de WPF à Unity en JSON

```
public class WeatherForecastWithPOCOs
{
    0 références
    public DateTimeOffset Date { get; set; }
    0 références
    public int TemperatureCelsius { get; set; }
    0 références
    public string Summary { get; set; }
    public string SummaryField;
    0 références
    public IList<DateTimeOffset> DatesAvailable { get; set; }
    0 références
    public Dictionary<string, HighLowTemps> TemperatureRanges { get; set; }
    0 références
    public string[] SummaryWords { get; set; }

    1 référence
    public WeatherForecastWithPOCOs()
    {
        SummaryWords = new string[] { "1", "3", "3" };
        TemperatureRanges = new Dictionary<string, HighLowTemps>();
        TemperatureRanges.Add("range1", new HighLowTemps());
    }
}
2 références
public class HighLowTemps
{
    0 références
    public int High { get; set; }
    0 références
    public int Low { get; set; }
}
```

- [16:48:48] {"Date":"0001-01-01T00:00:00+00:00","TemperatureCelsius":0,"Summary":null,'
'DatesAvailable":null,"TemperatureRanges":{"range1":{"High":0,"Low":0}},"SummaryWords":["1","3","3"]}

21.04.2021 - 08h05/17h00

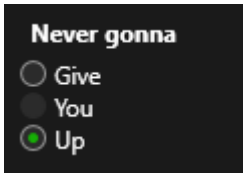
- Documentation
  - Complétion de la comparaison de technologie d'interface graphique des pipelines.
  - Problèmes rencontrés - Pipelines
  - Intégration d'Unity à WPF
- Modification de l'envoi de données pour utiliser un BinaryWriter à la place d'une Stream
  - Augmente la limite de données transmissible d'un Uin16 à int32.
- Modification de la réception des données pour utiliser un BinaryReader à la place d'un stream.
- Test d'intégration + communication réussie
- Début de la création de l'UI
  - Recherche d'un thème pour l'application
    - Thème prometteur
    - Test du thème
    - Le thème fonctionne et possède énormément de possibilités
  - Test de différents composants du thème
  - Boutons
  - TabControl
    - Changement d'icônes

        - Changements de la couleur
        - Tentative de changer le background pour un background non prédéfini.
        - Intégration d'une autre page xaml depuis le TabControl
    - Création d'une page, pour la simulation, les paramètres graphiques et les paramètres de la simulation
    - Abandon du thème trouvé dû à son manque de responsivité malgré sa grande variété de contenu.

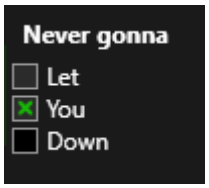
22.04.2021 - 08h05/17h00

- UI
    - Ajouts d'icône de boutons
    - Création de l'interface principale
        - Menu
        - Sections
        - Responsivité
    - Utilisation des columns et rows
    - Impossible de changer la couleur de la barre de titre du projet
        - Désaffichage de la barre
        - Création de ma propre barre
        - Différence entre les deux versions
        - 

- 

  - Modification des méthodes de la barre de titre pour être utilisable sur toutes les pages
  - Modification de la structure des boutons pour intégrer directement l'image dans leur balise et non appeler une image. La version précédente empêchait plusieurs boutons d'avoir la même image. Résultat en l'affichage de premier bouton sans image.
  - Création de différentes pages de contenus
  - Navigation entre les pages
  - Modification de l'affichage d'un slider
    - Difficulté à trouver les bons paramètres
    - Binding de la couleur à la valeur du slider
    - Création d'un template pour le slider en récupérant la version de base de WPF
    - Suppression de la track-bar
    - Modification du RepeatButton
      - Création de deux versions se situant à gauche et droit du thumb et ayant des couleurs différentes
    - Comparaison du slider de Windows et du slider personnalisé à l'aide d'un template
      - 
  - Modification de l'affichage des radioBoutons en récupérant la version de base de WPF
    - Nouveau design
      - 

  - Modification des couleurs pour qu'elles soient contenues dans des balisent facilement modifiable.
  - Modification de l'affichage des checkbox en récupérant la version de base de WPF
    - Nouveau design



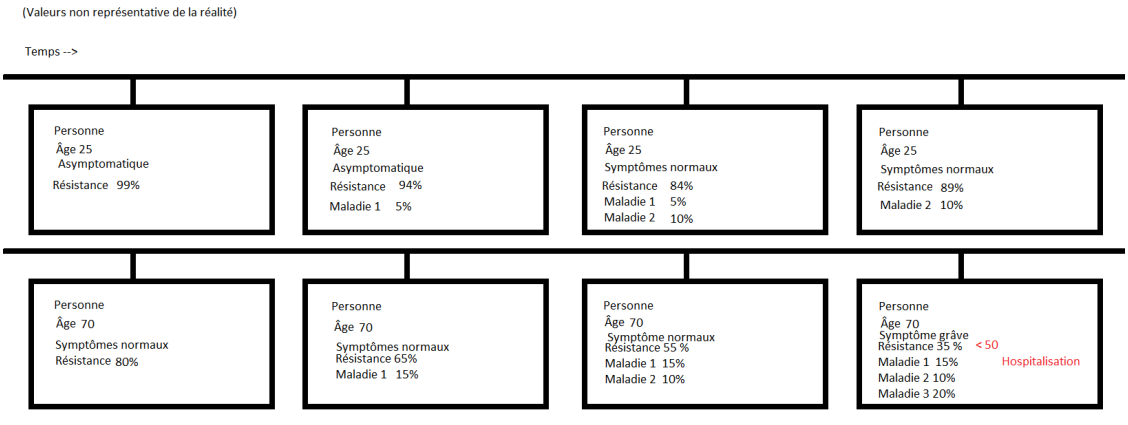
- Modification de la fermeture des fenêtres pour cacher les fenêtres secondaires et fermer le programme en cas de fermeture de la fenêtre principale.

23.04.2021 - 08h05/16h10

- Documentation
  - Organisation
  - Maquettes
  - Introduction
    - Technologies utilisées
  - Schéma de fonctionnement
  - Description des technologies utilisées
- Réflexion sur le fonctionnement des threads pour l'application.
  - Parallel
- Installation LiveCharts
  - Installation packet LiveCharts
  - Installation packet LiveCharts.Wpf
  - Installation packet LiveCharts.Geared
- Graphiques
  - Bug à chaque changement de taille, régénérer la solution est nécessaire.
    - Uniquement dans le concepteur et non au lancement du programme
    - Nettoyer la solution permet de tout de même voir l'affichage de la fenêtre
  - Intégration de graphiques
    - Courbe
    - HeatMap
    - Colonne
    - Ligne
    - Bulle
    - Circulaire (Camembert)
    - Area Stack

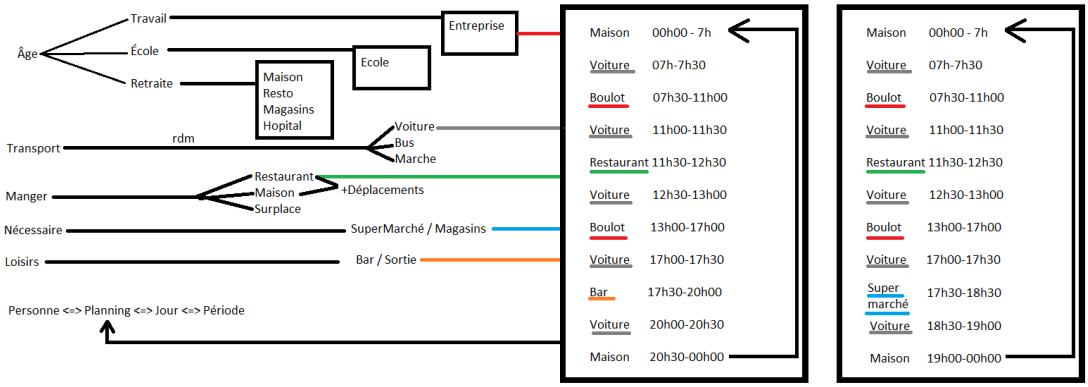
26.04.2021 08h05 / 16h10

- Début de la création de la simulation
- Récupération du code du stage et suppression des éléments graphiques
- Commentaires
- Création d'une classe "Site" regroupant tous les lieux et véhicules
- Relecture du cahier des charges
- Relecture du fichier Excel contenant la transmission par aérosol 2021_COVID-19_Aerosol_Transmissionn_Estimator
- Création du code des lieux en fonction du fichier Excel
    - Taille du bâtiment, longueur - largeur - hauteur - air - volume
    - Paramètres de l'air et de déposition
    - Utilisation du type double pour le calcul du taux d'infection
    - Suppression de certains paramètres non utilisés
- Réflexion sur la structure du code et l'interaction entre les individus et les lieux
    - Résistance au virus
        - Maladies
        - Âge



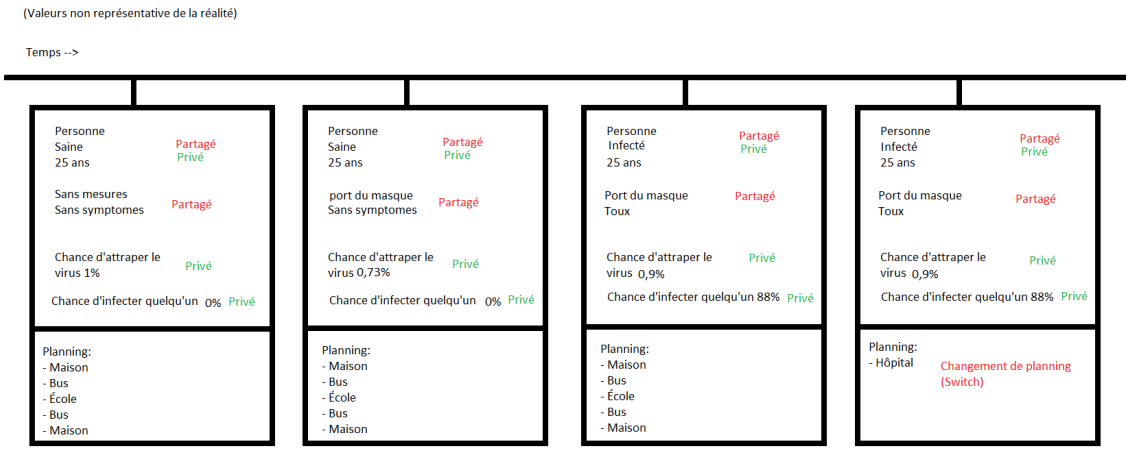
    - Création des plannings
        - Pattern

        - (Gérer les cercles sociaux)
        - Création en fonction de l'âge



    - Mesures & Symptômes
        - resistance
        - comportement
        - Augmentent certains paramètres

(Valeurs non représentative de la réalité)



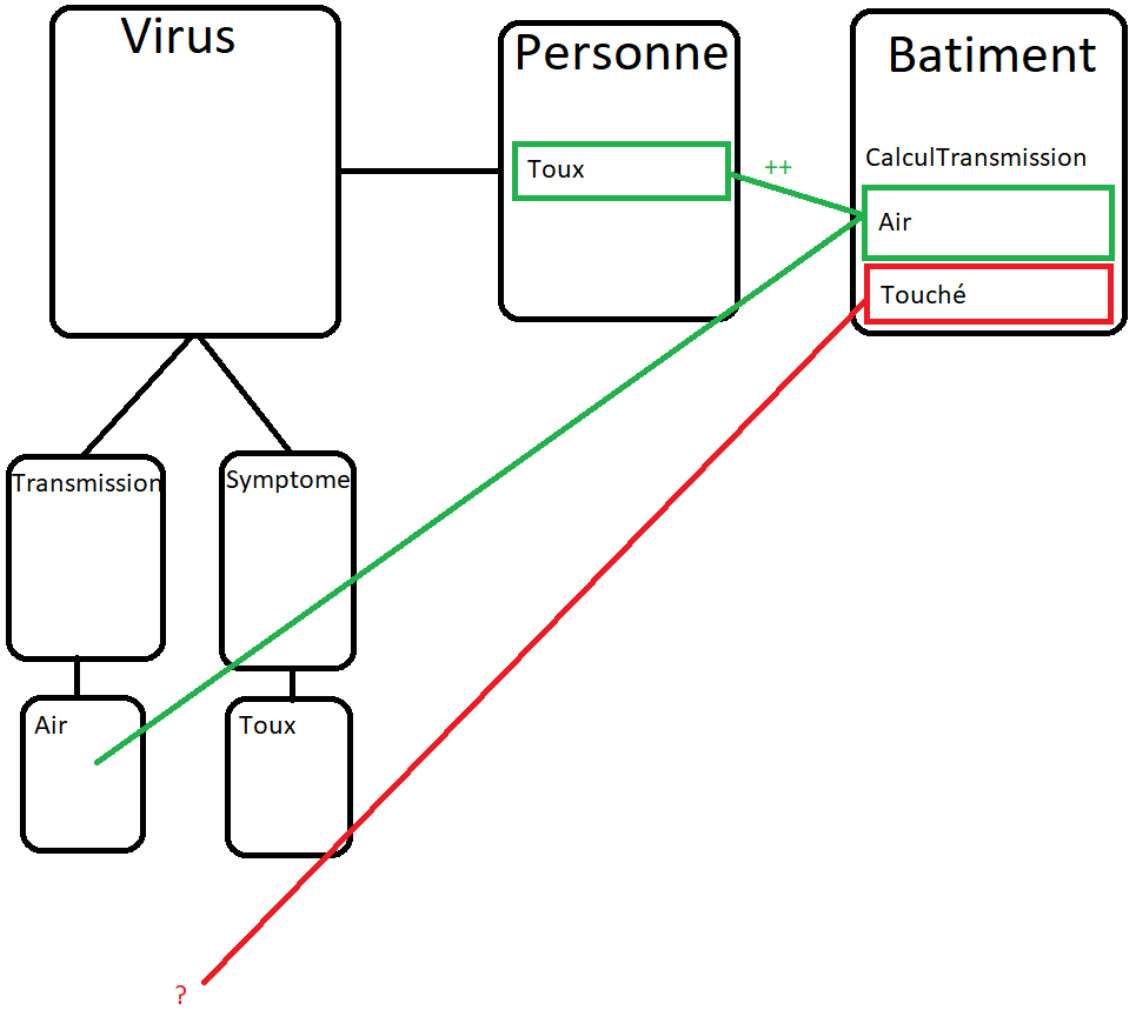
    - Les activités ne contiennent que des types d'objets permettant de définir ce que l'individu fait. Le planning contient les lieux et les indique à l'individu.
- Création des plannings en fonction de la figure concernant leur création.
- Nombre d'école et autres bâtiments en fonction du nombre de personnes. Genève servant d'exemple.
    - Formules
        - 100 + Abs((y2 - y1) / y1)*100 = %
        - (% - 100)/100 * y1 + y1 = y2
    - ~0.033% --> 165 écoles pour 500'000 personnes 100 - ((165 - 500'000) / 500'000)*100
    - ~6,74% --> 33'700 entreprises pour 231'000 emplois
    - ~0.0016% --> 8 hôpitaux, 2 cliniques, 30 lieux de soins
    - ~0,0272% --> ~136 supermarchés
    - ~50% - 100% --> ~250'000 appartements
- pareil, mais pour l'âge de la population
    - ~22% 0-19 ans
    - ~63% 20-64 ans
    - ~10% 65-79 ans
    - ~5% 80+ ans
    - Source
- Pareil pour les transports
    - ~36% --> 218'000 voitures (Chauffeur ou passagers)

    - ~15% --> 452 + 117 tpg
    - ~10% --> ~ 200'000 vélos
    - Source
- Réflexion sur la création des transports et bâtiments et sur la source de certaines données.
    - On a une population de 100 personnes
        - 22% 0-19 ans
        - 63% 20-64 ans
        - 15% 65+ ans
    - Il faut donc au minimum
        - Une école
        - Un supermarché
        - Une maison par personne (pour le moment les < 19 ans vivent seuls)
    - On crée les bâtiments en fonction des besoins cités
    - Pour les véhicules
        - On crée 36 voitures
        - et des bus (à voir plus tard pour la quantité)
    - Ensuite, on crée les 100 individus
        - En fonction du pourcentage d'âge
        - On leur attribue
            - Une maison vide
            - Une école
            - Un moyen de transport (vide si voiture)
        - On créé leurs résistances

27.04.2021 08h05 / 16h30

- Les véhicules dans la simulation agissent exactement comme les bâtiments.
    - Pas de déplacements
    - La voiture étant la seule d'"unique" à une personne.
        - La différence réside dans l'assignation et non dans le fonctionnement.
- Suppression du parent "Vehicle" et "Building"
- Création d'un objet parent "Site" représentant tous les lieux, dont les véhicules et l'extérieur.
- Création de variables constantes globales pour la création des sites
- Réflexion sur le fonctionnement des individus
    - Marche à suivre.
    - Utiliser du async pour récupérer le taux de contamination ? (Attendre que tout le monde soit bien entré)
- Création des individus
    - Maladies
    - Résistance au virus
    - Durée de l'immunité
    - Durée de l'infection par le virus
- Création des maladies
    - Puissance
        - En fonction de l'âge
    - Durée

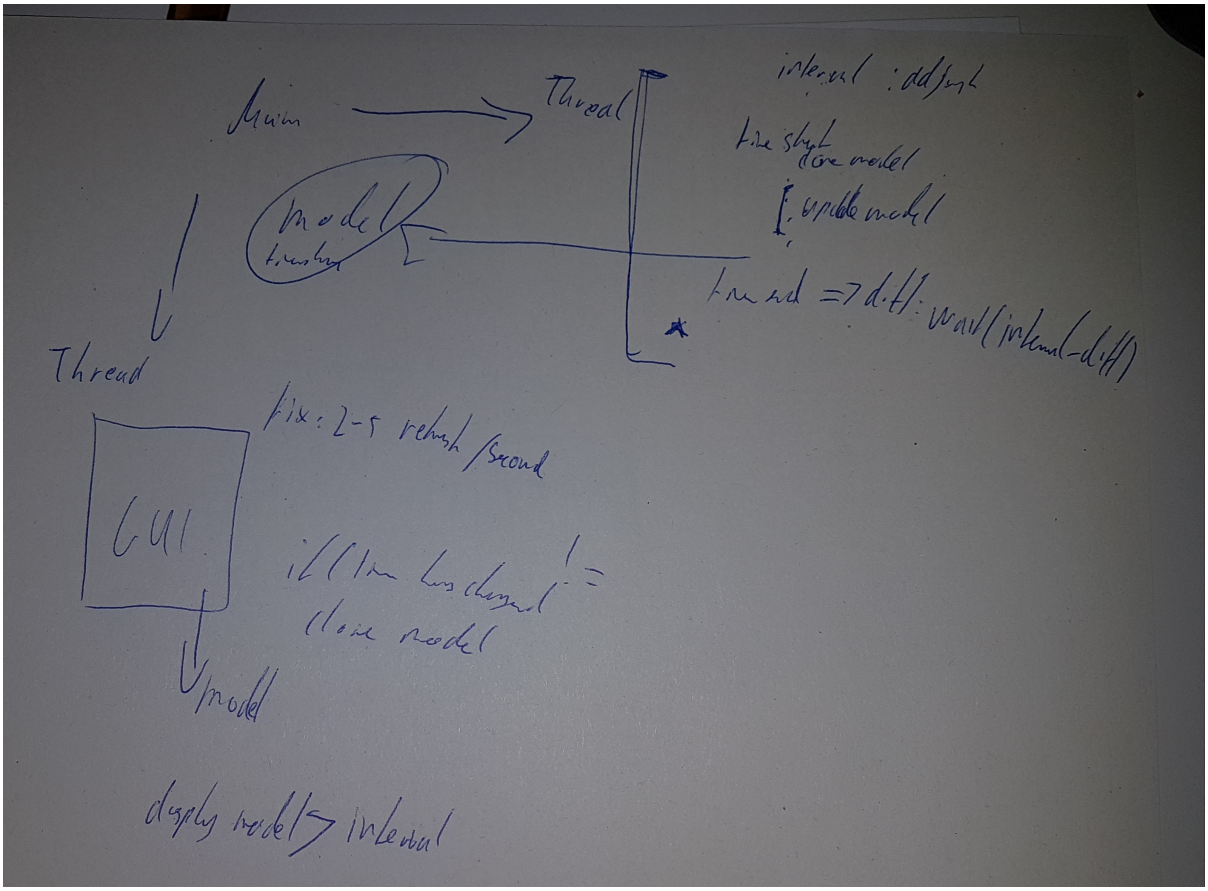
- Recherche de source pour gérer mathématiquement les probabilités qu'une personne attrape des maladies en fonction de son âge.
    - Meilleure compréhension du sujet, mais rien de concluant.
- Modification des namespace qui avaient automatiquement créé sous namespace en créant des dossiers dans le programme WPF
- Modification du site pour prendre en compte les personnes qui se trouvent dedans.
    - Cette modification impacte aussi à quel moment le calcul des probabilités est effectué.
    - Si une personne entre ou sort. Les probabilités sont recalculées.
    - Sinon on récupère la dernière valeur sans recalculer
    - Dois recalculer si une personne a été infectée. (en prenant en compte le temps d'incubation du virus, ce n'est pas probablement pas nécessaire.)
- Données récupérées concernant le temps d'incubation et d'infection du covid
    - Durée du covid
    - Incubation et durée
- Suppression des cercles familiaux et d'amis du planning
    - Raison :
        - Pas par manque de temps, mais par logique due au fonctionnement de la simulation
        - Les plannings gèrent entièrement la simulation, les lieux., les horaires, les contacts.
        - Au final, la famille sont les personnes habitant au même endroit. Les amis, les personnes qui vont aux mêmes bars.
        - La création du planning s'occupera donc de gérer les contacts proches
    - Cette modification pourrait être revue plus tard en fonction de l'évolution du fonctionnement de la simulation et des interactions entre les objets.
- Commentaires et création d'en-têtes dans le code
- Recherche de conversion de probabilité d'années en mois
    - Rien trouvé
    - Uniquement dans l'autre sens
- Est-il justifié d'avoir des symptômes de virus ?
    - La résistance au virus étant simplifié et l'accent étant majoritairement mis sur la propagation, elle-il nécessaire d'intégré des symptômes qui dans 90% des cas ne changent en rien le taux de contamination (excepté la toux donc).
    - Simplifier en utilisant simplement le taux de transmission ?
        - Pas viable, car asymptomatiques
    - Les symptômes pourraient être utilisés pour détecter le virus dans la simulation pour les quarantaines et autres.
    - Garder les symptômes, mais avec pour la majorité, uniquement un indice de détection
- Création de la classe Virus, Transmission et Symptom
- Création du fichier XML comportant les données du Covid
- Lecture du fichier depuis le code et récupération des données.
- Fonctionnement imaginé du virus et de ses composants



-
  - Informations importante concernant la respiration et affectant donc la toux Source
    - Resting – Oral breathing = 2.0 quanta/h
    - Resting – Speaking = 9.4 quanta/h
    - Resting – Loudly speaking = 60.5 quanta/h
    - Standing – Oral breathing = 2.3 quanta/h
    - Standing – Speaking = 11.4 quanta/h
    - Standing – Loudly speaking = 65.1 quanta/h
    - Light exercise – Oral breathing = 5.6 quanta/h
    - Light exercise – Speaking = 26.3 quanta/h
    - Light exercise – Loudly speaking = 170 quanta/h
    - Heavy exercise – Oral breathing = 13.5 quanta/h
    - Heavy exercise – Speaking = 63.1 quanta/h
    - Heavy exercise – Loudly speaking = 408 quanta/h
- Modification de la transmission par aérosol sur les sites pour prendre en compte les paramètres individuels de chaque individu présent sur les lieux.

28.04.2021 07h50 / 17h00

- Refactoring
  - Certain point logiques à revoir

        - Le lieu calculait le risque de transmission aérosol alors que c'est au type de transmission du virus de le faire. Ou un mix des deux.
- Penser à modifier les calculs récupérés sur le fichier Excel pour être plus en accord avec le fonctionnement de la simulation.
- Revoir le pattern visistor pour éventuellement l'implémenter pour le calcul de risque de la transmission
- Transmission effectuée
    - Revoir la structure du code pour certaines améliorations nécessaires
- Ajout de données du covid dans le fichier Excel
- Refactorisation de la lecture du fichier XML
    - Échec
    - La structure de code essayée ne correspond pas aux besoins de la simulation
        - Tentative de conversion d'une boucle itérant dans le XML à un système de query. Cependant, les query ne permettent pas de récupérer des valeurs précisent en groupes et de les lire individuellement.
- Penser à remplacer le XML par du JSON.
    - Permettant de générer le virus de façon dynamique en fonction du JSON.
- Création du diagramme de classe
- Discussion avec M. Mathieu sur les threads de l'application
    - Création de 2 thread, un gérant la simulation, l'autre la GUI.
    - À la place d'un timer, vérifier le temps entre deux "tick" et attendre si le traitement est trop rapide.
    - 

- Création du "timer" dans la simulation
- lecture du code TrainSimulatorApp
    - Fonctionnement des Threads
    - Boucle
- Intégration de la simulation au programme WPF

    - Start
    - Stop
    - Changement de l'interval en fonction du slider
- Recherche sur le binding de données en WPF
    - Trop complexe pour le moment. À éventuellement ajouter plus tard.
- Création des bâtiments
    - Reprise de réel
- Perte du câble du disque dur externe --> pas de sauvegarde dessus
- Données concernant le nombre d'étudiants à Genève pour calculer le ratio école / élèves.
    - Total = 103'008 étudiants en 2020
    - Sources
- Rectification du compteur de bâtiment nécessaire à la population
    - Formules
        - (% - 100) / 100 * y1 + y1 = y2
        - 100 - ((y2 - y1) / y1) * 100 = %

29.04.2021 08h05 / 17h00

- Début de la création des plannings
    - Modification de la création des bâtiments
    - Ajouts des restaurants
    - Ajouts de l'extérieur
- Quantité de personne retraitée à Genève = ~14%
    - Source
- Création de la structure du planning d'un individu qui travaille
    - Jour de travail :

```
Home 5h-8h
trajet (5h30 - 8h30)
company (Arrivé 6h - 9h  Départ (11h-12h))
trajet | company (11h30-12h30)
Resto  | Company (Arrivé 12h - 13h  Départ (12h30-13h30))
trajet | company (13h-14h)
company (Arrivé 13h30-14h30  Départ (16h-19h))
trajet (16h30 - 19h30)
Supermarché 30m-1h30 | maison 30min-Fin | restaurant 1h-3h | Bar (17h-19h)
                    \ /                \ /                 | Bar
Supermarché           | maison           | restaurant       | Bar (17h30-19h)
                    \                  \ /                 | Bar
                      | maison           | restaurant       | Bar (19h - 20h)
 Outisde              | maison                              | Bar (20h - minuit)
 maison
```

- Ajouts de méthode d'extensions pour le calcul de random booléen
    - Méthode choisissant un choix aléatoire entre plusieurs choix à disposition selon un poids
- ~~Création d'une méthode créant la soirée d'une personne de manière dynamique~~

    - ~~Manque de connaissance pour la réaliser de manière propre et fonctionnel~~
- Création de différents patterns préfaits choisis de manière aléatoire.
    - Simplifie la version précédente
    - Suffisamment aléatoire
- Utiliser une string en temps que seed pour générer des plannings ?
    - +Meilleur contrôle sur les plannings
    - +Simple
    - +Possibilité d'ajouter une génération aléatoire plus tard
    - -Prégénéré
    - -durée des périodes (ne devraient pas changer)
- Création de la string
- Modification de l'emplacement des méthodes de création de planning.
    - CreateDays simulation --> Day
    - CreatePlanning simulation --> Planning
- Génération du planning en fonction de la seed
- Création de seeds pour étudiants, adultes et retraités
- Première compilation de la simulation (Alpha) réussie après quelques bugs mineurs
- Suppression des paramètres "Pression", "Temperature" et "Co2" dans les lieux, car pas encore utilisé par .

30.04.2021 08h05 / 17h00

- Documentation
    - ~~Revoir la structure pour fonctionner avec le système de sprint.~~
    - Réduction de la quantité de code affiché dans la documentation
    - Modification de l'agencement
    - Correction de l'orthographe
- Discussion avec M. Mathieu
    - Contenu du rapport
        - Pas de code
        - Pseudo-codes OK
        - Utiliser un "ID" sur le pseudo-code qui se retrouve dans les commentaires du code correspondant. Permets de retrouver rapidement.
    - Contenu du LogBook
    - Rendu Sprint 1
        - Uniquement le rapport
- Modification du diagramme de classe
- Avancement de l'interface graphique simulation
    - Ajout d'une barre de scroll
    - Ajouts d'éléments à l'emplacement des futurs éléments graphiques
    - Création d'une grille par le code
    - Création d'une méthode d'extension permettant de récupérer le dernier index d'une liste.
    - Ajouter / retirer colonnes et lignes responsives