Documentation

Procédure d’installation

Installation du jdk 11

Procédure de désinstallation

Desinstaller le jdk 11

Entrée et sortie de chaque composant

Package controlleur

Classe Archi

controlleur.Archi.main(String[])

Instancie un nouveau parser , une fenêtre et un nouveau controlleur.

Classe Controlleur

Attribut

private Fenetre view;
private Parser parser = new Parser();
private ArrayList<Object> creneau_read_parser;
private ArrayList<Object> ue_read_parser;
private ArrayList<Object> classe_read_parser;
private ArrayList<Object> session_read_parser;

controlleur.Controlleur.Controlleur(Fenetre, Parser)

Constructeur de la classe controlleur qui prend en paramètre 2 objets , de type Fenêtre et parser respectivement. Associe ces 2 paramètres ou attributs parser et view. Finalement, affiche les informations récoltées par le parser. Ces informations représentent les entrées antérieures des utilisateurs.

controlleur.Controlleur.createUEListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle la création d'une ue est suivi d'un enregistrement de cette info dans le fichier.

controlleur.Controlleur.deleteUEListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle la suppression d'une ue est suivi d'un enregistrement de cette info dans le fichier.

controlleur.Controlleur.addCreneauListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle l'ajout d'un créneau est suivi d'un enregistrement de cette info dans le fichier.

controlleur.Controlleur.deleteCreneauListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle la suppression d'un créneau est suivie d'une suppression de cette info dans le fichier.

controlleur.Controlleur.createClasseListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle la création d'une classe est directement suivie d'un enregistrement de cette info dans le fichier.

controlleur.Controlleur.deleteClasseListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle la suppression d’une classe est directement suivie d'une suppression de cette info dans le fichier.

controlleur.Controlleur.addSessionListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle l’ajout d’une classe est directement suivie par un enregistrement de cette info dans le fichier.

controlleur.Controlleur.deleteSessionListener.actionPerformed(ActionEvent)

Méthode hérite de la classe ActionListener , à travers laquelle la suppression d’une session est directement suivie d'une suppression de cette info dans le fichier.

Classe Parser

Attribut
private String path = new String();

controlleur.Parser.Parser()

Constructeur de la classe Parser. Il détecte l'os de l'utilisateur et détermine le chemin des données à analyser par le parser en fonction de cet os.

controlleur.Parser.write(String, Class<?>)

Prend en entrée une chaîne de caractère et une liste. Cette méthode sert à écrire la chaîne de caractère dans le fichier de données correspondant .

controlleur.Parser.remove(String, Class<?>)

Prend une chaîne de caractère et une liste. Cette méthode sert à retirer une chaîne de caractères dans le fichier de données correspondant .

controlleur.Parser.read(Class<?>)

Méthode utilisée pour lire dans les fichiers de données et retourne une liste.

controlleur.Parser.getPath()

Retourne l'attribut path.

Package Modele

Classe AnnneeScholaire

private int anneeDepart, anneeFin;
public AnneeScolaire (int anneeDepart, int anneeFin)

Constructeur qui prend en entrée anneDepart et anneeFin et attribue ces deux valeurs à AnneeScholaire.anneeDepart et AnneScholaire.anneeFin

modele.AnneeScolaire.getAnneeDepart()

Retourne anneeDepart

modele.AnneeScolaire.setAnneeDepart(int anneeDepart)

Prend en entrée l'année de départ et met la valeur d'anneeDepart dans l'attribut privé anneeDepart .

modele.AnneeScolaire.getAnneeFin()

Retourne l'attribut anneeFin

modele.AnneeScolaire.setAnneeFin(int)

Prend en entrée un entier et met la valeur de cette entier à l'attribut anneeFin.

modele.AnneeScolaire.isValid(int, int)

Prend en entrée une 2 entier , et met leur valeur dans les attributs privés anneeDebut et anneeFin.

modele.AnneeScolaire.toString()

Override de la méthode toString. Retourne anneeDepart - anneeFin

classe Classe

Attributs
private String id = UUID.randomUUID().toString(), nomination;
private AnneeScolaire anneeScolaire;

modele.Classe.Classe(String, int, int)

Constructeur qui prend en entrée une chaine de caractere et 2 entiers et met leur valeurs dans les attributs privés nomination et anneeScholaire.

modele.Classe.getId()

Retourne la valeur de l'attribut privé id

modele.Classe.getNomination()

Retourne la valeur de l'attribut privé nomination

modele.Classe.setNomination(String)

Prend en entrée une chaîne de caractère et place la valeur de cette chaîne dans l'attribut privé nomination.

modele.Classe.getAnneeScolaire()

Retourne la valeur de l'attribut anneeScholaire.

modele.Classe.setAnneeScolaire(AnneeScolaire)

Prend une entrée de type AnneeScolaire et met sa valeur dans l'attribut privé anneeScolaire.

modele.Classe.toString()

Override de la méthode toString qui retourne les 2 attributs de la classes de la forme suivante nomination + " " + anneeScholaire

Classe Creneau

Attributs
private String id = UUID.randomUUID().toString();
private DateScolaire date;
private HeureScolaire horaire;

modele.Creneau.Creneau(int, int, int, int, int)

Constructeur de la classe Creneau , prend en entrée 5 entiers , qui représente respectivement le jour , le moi , l'année , heure et la minute. Et crée un objet de type Date avec les entrées jours, moi et années et crée un objet de type HeureScolaire à l'aide des entrées heure et minute.

modele.Creneau.Creneau(String, String, String, String, String)

Constructeur de la classe Creneau , prend en entrée 5 string, qui représente respectivement le jour , le moi , l'année , heure et la minute. Et crée un objet de type Date avec les entrées jours, moi et années et crée un objet de type HeureScolaire à l'aide des entrées heure et minute.

modele.Creneau.getId()

Retourne la valeur de l'attribut id de créneau

modele.Creneau.getDate()

Retourne l'attribut de type date du creneau

modele.Creneau.setDate(DateScolaire)

Prend un objet de type DateScolaire en entrée et met la valeur de cet objet dans l'attribut date de creneau.

modele.Creneau.getHoraire()

Retourne l'attribut horaire de creneau

modele.Creneau.setHoraire(HeureScolaire)

Prend une entrée de type HeureScolaire et met sa valeur dans l'attribut horaire de creneau.

modele.Creneau.toString()

Override de la méthode toString qui retourne les attributs de date et horaire de creneau de la forme suivante : date + " " + horaire.

Classe DateScholaire

Attribut
private int jour, mois, annee;

modele.DateScolaire.DateScolaire(int, int, int)

Constructeur de la classe DateScolaire qui prend en entrée 3 entiers qui représente le jour , moi et année. Et met la valeur de jour , dans l'attribut jours , met la valeur de moi dans l'attribut mois et mais la valeur d'année dans l'attribut années.

modele.DateScolaire.DateScolaire(String, String, String)

Constructeur de la classe DateScolaire qui prend en entrée 3 chaînes de caractères. Qui représente le jour , moi et l'année . Puis utilise parseInt pour prendre lire les valeurs en tant qu'entiers et puis mettre ces valeurs dans les attributs jour, mois année.

modele.DateScolaire.getJour()

Retourne la valeur de l'attribut jours de date scolaire.

modele.DateScolaire.setJour(int)

Prend en entrée un entier qui et met la valeur dans l'attribut jour de DateScolaire.

modele.DateScolaire.getMois()

Retourne la valeur de l'attribut mois

modele.DateScolaire.setMois(int)

Prend en entrée un entier et met la valeur de cette entier dans l'attribut mois .

modele.DateScolaire.getAnnee()

Retourne la valeur de l’attribut annee de DateScolaire.

modele.DateScolaire.setAnnee(int)

Prend en entrée un entier met la valeur de cette entier dans l’attribut année.

modele.DateScolaire.isValid(int, int, int)

Prend en entrée 3 entiers qui représente le jour , mois et l'année et vérifie les conditions de validité dessus.

modele.DateScolaire.verifAnnee(int)

Prend en entrée un entier qui représente l'année et retourne true ou false dépendamment des conditions de validité sur les années.

modele.DateScolaire.compateTo(DateScolaire)

Prend en entrée un objet de type DateScolaire et retourne en sortie un booléen dépendamment du résultat de la comparaison entre les attribut de DateScolaire de l’objet courant et les attribut de l’objet de type DateScolaire passe en paramètre.

modele.DateScolaire.toString()

C’est un override de la méthode toString qui retourne les attributs de l’objet DateScolaire sous la forme suivante jour/moi/annee

modele.DateScolaire.equals(DateScolaire)
Prend en entrée un objet de type DateScolaire et retourne un booléen qui dépend de la comparaison entre l’objet DateScolaire courant et l’objet de type DateScolaire passe en paramètre.

Classe HeureScholaire

Attributs
private int heure, minute;

modele.HeureScolaire.HeureScolaire(String, String)

Constructeur de la classe HeureScolaire prend en entrée 2 chaine de caractère qui représente heure et minute et mettre le valeur de ces paramètres dans les attributs heures et minute

modele.HeureScolaire.getHeure()

Retourne la valeur de l'attribut heure de HeureScolaire.

modele.HeureScolaire.setHeure(int)

Prend en entrée un entier et met la valeur de cette entier dans l'attribut heure.

modele.HeureScolaire.getMinute()

Retourne la valeur de l'attribut minute de HeureScolaire

modele.HeureScolaire.setMinute(int)

Prend en entrée un entier et met la valeur de cet entier dans l'attribut minute de HeureScolaire.

modele.HeureScolaire.isValid(int, int)

Prend en entrée 2 entier qui représente l'heure et la minute et retourne un booléen en fonction de la validité de l'heure et de la minute.

modele.HeureScolaire.compateTo(HeureScolaire)

Prend en entrée un objet de type HeureScolaire et retourne un booléen en fonction de la comparaison entre les valeurs des attributs et celle des attributs de HeureScolaire passée en paramètre.

modele.HeureScolaire.toString()

Override de la fonction toString qui retourne les attributs heure et minute sous la forme suivante : heure : minute

modele.HeureScolaire.equals(HeureScolaire)

Prend en entrée un objet de type HeureScolaire et retourne un booléen en fonction de la comparaison entre les attributs avec les valeurs des attributs de l'objet passé en paramètre.

Class Session

Attributs

private UE ue;
private Classe classe;
private ArrayList<Creneau> creneaux = new ArrayList<>();

modele.Session.Session(UE, Classe)

Constructeur de la classe session , prend en entrée un objet de type UE et un objet de type Classe et met la valeur de ces objet dans les attributs ue et classe.

modele.Session.Session(UE, Classe, ArrayList<Creneau>)

Constructeur de la classe session , prend en entrée un objet de type UE et un objet de type Classe , un objet de type ArrayList et met la valeur de ces objet dans les attributs ue et classe et dans creneaux

modele.Session.ajoutCreneau(Creneau)

Prend en entrée un objet de type créneau et ajoute ce paramètre dans l'attribut créneaux (liste de créneaux.

modele.Session.retirerCreneau(Creneau)

Prend en entrée un objet de type créneau et le retire ce creneau de l'attribut creneaux. (liste de creneaux)

modele.Session.existeCreneau(Creneau)

Prend en entrée un objet de type créneau et retourne un booléen en fonction de la présence du paramètre Créneau dans la liste des créneaux (l'attribut créneaux)

modele.Session.findCreneau(String, String, String, String, String, String, String)

Prend en paramètre 7 chaînes de caractères qui représentent jour , moi , année , heureDepart , minuteDepart , heureFin , minuteFin et Retourne un créneau vérifiant ces paramètres.

modele.Session.findCreneau(int, int, int, int, int, int, int)

Prends en paramètre 7 entiers qui représentent jour , moi , année , heureDepart , minuteDepart , heureFin , minuteFin et Retourne un créneau vérifiant ces paramètres.

modele.Session.findCreneau(Creneau)

Prend en paramètre un objet de type Creneau et retourne ce même créneau si il est présent dans la liste des créneaux.

modele.Session.getUe()

Retourne l'attribut ue

modele.Session.setUe(UE)

Prend en paramètre un objet de type UE et met la valeur de cet objet dans l'attribut ue

modele.Session.getClasse()

Retourne l'attribut classe.

modele.Session.setClasse(Classe)

Prend en paramètre un objet de type Classe et met la valeur de ce paramètre dans l'attribut classe.

modele.Session.getCreneaux()

Retourne l'attribut creneaux.

modele.Session.setCreneaux(ArrayList<Creneau>)

Prend en paramètre une liste de créneaux et met la valeur de ce paramètre dans l'attribut creneaux.

modele.Session.toString()

Override de la méthode toString qui affiche les attributs de cette objet de la forme suivante : Ue | classe

modele.Session.equals(Session)

Prend en paramètre un objet de type Session et retourne un booléen en fonction de la comparaison des valeurs des attributs à la valeur des attributs de l'objet passé en paramètre.

Classe UE

Attributs
private String sigle, nomination;

modele.UE.UE(String, String)

Constructeur de la classe UE, Prend en entrée 2 chaines de caractère qui représentent le sigle et la nomination et met leur valeur dans les attributs sigle et nomination.

modele.UE.getSigle()

Retourne l'attribut sigle.

modele.UE.setSigle(String)

Prend en entrée un paramètre de type chaîne de caractère et met la valeur de ce paramètre dans l'attribut sigle .

modele.UE.getNomination()

Retourne l'attribut nomination de l'objet.

modele.UE.setNomination(String)

Prend en entrée un paramètre de type chaîne de caractère et met la valeur de ce paramètre dans l'attribut nomination.

modele.UE.toString()

Override de la méthode toString qui affiche les attribut de la façons suivante sigle:nomination

modele.UE.equals(UE)

Prend en paramètre un objet de type UE et retourne un booléen en fonction de la comparaison entre les attribut de ce paramètre et ceux de l'objet.

Package vue

Classe ClasseTab

Attribut

```
private JPanel classePanel;
private JButton ajouter = new JButton("ajouter une classe");
private JButton supprimer = new JButton("supprimer une classe");
private JLabel label = new JLabel();
private JTextField name_classe = new JTextField();
private int current_year = ZonedDateTime.now(ZoneId.of("CET")).getYear();
private SpinnerModel value_year =  new
SpinnerNumberModel(current_year,current_year-10,current_year+10,1);
private JSpinner spinner_years = new JSpinner(value_year);
private JScrollPane scrollListClasse = new JScrollPane();
private ArrayList<Classe> classe = new ArrayList<Classe>();
private DefaultListModel<Classe> listModel = new DefaultListModel<Classe>();
private JList<Classe> listClasse = new JList<Classe>(listModel);
private GridBagConstraints c = new GridBagConstraints();
```

vue.ClasseTab.ClasseTab(JPanel)

Constructeur de la classe ClasseTab , prend en entrée un objet de type JPanel et met sa valeur dans l'attribut classePanel et puis initialise le composant.

vue.ClasseTab.initComponent()

Ajoute les labels et les boutons dans l'onglet relatif aux classe et ajuste leur dimension.

vue.ClasseTab.displayClasse()

Parcours la liste des classes en attribut (classe) et les affiche ultérieurement.

vue.ClasseTab.writeMessage(String)

Prend une chaîne de caractère en entrée qui représente le message d'erreur ou de succès et affiche un message plus descriptif de cette erreur ou de ce succès.

vue.ClasseTab.setNewClasse(Classe)

Prend en entrée un objet de type Classe et l'ajoute à l'attribut classe.

vue.ClasseTab.setDeleteClasse(int)

Prend en entrée un entier qui sera utilisé comme indice de suppression pour une classe dans l'attribut classe. (qui est une liste de classe)

vue.ClasseTab.getDataClasseList()

Retourne l'attribut classe.

vue.ClasseTab.getAddClasse()

Retourne l'attribut ajouter.

vue.ClasseTab.getDeleteClasse()

Retourne l'attribut supprimer.

vue.ClasseTab.getClasseFormation()

Retourne une chaîne de caractère qui représente la formation de l'attribut name_classe.

vue.ClasseTab.getClasseYear()

Retourne une chaîne de caractère qui représente l'année de la classe.

vue.ClasseTab.getIndexListClasse()

Retourne la liste d'indice de l'attribut listClasse.

Classe CreneauTab

vue.CreneauTab.CreneauTab(JPanel)

Constructeur de la classe CreneauTab qui prend en entrée un objet de type JPanel et met sa valeur dans le l'attribut creneauPanel. Et puis initialise le composant en appelant la méthode initComponent().

vue.CreneauTab.initComponent()

Ajoute des options pour jours , ces options variant de 1 à 31 . De même, on ajoute des options pour le mois qui varie de 1 à 12. Et on ajoute les labels correspondant à mois et jours. De plus on ajoute les options pour les heures qui varient de 0 à 24 et les options pour les minutes qui varient de 0 à 60 ainsi que les labels correspondant. Finalement on ajoute les boutons ajouter un créneau et supprimer un créneau.

vue.CreneauTab.displayCreneau()

Parcour l'attribut créneau qui est une liste de creneau et les affiches dans l'onglet.

vue.CreneauTab.writeMessage(String)

Prend en entrée une chaîne de caractère qui représente un message d'erreur ou de succès et affiche une message plus descriptif que ce dernier.

vue.CreneauTab.setNewCreneau(Creneau)

Prend en entrée un objet de type Créneau et l'ajoute dans l'attribut creneau.

vue.CreneauTab.setDeleteCreneau(int)

Prends en entrée un entier qui servira d'indice pour la suppression d'un créneau dans la liste.

vue.CreneauTab.getDayCreneau()

Retourne une chaîne de caractères qui représente l'ensemble des options donnée à l'utilisateur pour les jours.

vue.CreneauTab.getMonthCreneau()

Retourne une chaîne de caractères qui représente l'ensemble des options donnée à l'utilisateur pour les Mois.

vue.CreneauTab.getYearCreneau()

Retourne une chaîne de caractères qui représente l'ensemble des options données à l'utilisateur pour les années .

vue.CreneauTab.getHourBeginCreneau()

Retourne une chaîne de caractères qui représente l'ensemble des options données à l'utilisateur pour les heures de début .

vue.CreneauTab.getHourEndCreneau()

Retourne une chaîne de caractères qui représente l'ensemble des options données à l'utilisateur pour les heures de fin.

vue.CreneauTab.getMinuteBeginCreneau()

Retourne une chaîne de caractères qui représente l'ensemble des options données à l'utilisateur pour les minutes de début.

vue.CreneauTab.getMinuteEndCreneau()

Retourne une chaîne de caractères qui représente l'ensemble des options données à l'utilisateur pour les minutes de fin.

vue.CreneauTab.getDataCreneauList()

Retourne l'attribut créneau.

vue.CreneauTab.addCreneau()

Retourne l'attribut ajouter.

vue.CreneauTab.deleteCreneau()

Retourne l'attribut supprimer.

vue.CreneauTab.getIndexListCreneau()

Retourne un entier qui représente l'indice de la l'attribut listCreneau.

Classe Fenetre

Attributs

```
private static final int WIDTH = 500;
private static final int HEIGHT =500;
private JTabbedPane tab = new JTabbedPane();
private JPanel classePanel = new JPanel();
private JPanel uePanel= new JPanel();
private JPanel creneauPanel = new JPanel();
private JPanel sessionPanel = new JPanel();
private UETab ueTab = new UETab(uePanel);
private CreneauTab creneauTab = new CreneauTab(creneauPanel);
```

vue.Fenetre.Fenetre()

Constructeur de la classe Fenetre, dans lequel on précise les caractéristiques de la fenêtre. (dimension , couleur , …) et initialise le composant.

vue.Fenetre.initComponent()

On ajoute les panels en attribut par exemple uePanel et creneauPanel pour avoir l'accès à ces panel depuis la fenêtre principale.

vue.Fenetre.getUETab()

Retourne l’attribut ueTab

vue.Fenetre.getCreneauTab()

Retourne l’attribut creneauTab.

vue.Fenetre.getClasseTab()

Retourne l’attribut classeTab.

vue.Fenetre.getSessionTab()

Retourne l’attribut sessionTab.

Classe SessionTab

vue.SessionTab.SessionTab(JPanel)

Constructeur de la Classe Session tab prend en entrée un objet de type JPanel et met sa valeur dans l'attribut sessionPanel. De plus, il initialise le composant.

vue.SessionTab.initComponent()

Initialise le composant SessionTab et met les attributs dans sessionPanel.

vue.SessionTab.displayUe()

Parcours la liste des ue (dans l'attribut ue) et les affiches à l'utilisateur.
vue.SessionTab.displayClasse()

Parcours la liste des classes (dans l'attribut classe) et les affiches à l'utilisateur.

vue.SessionTab.displayCreneau()

Parcours la liste des créneaux (dans l'attribut creneau) et les affiches à l'utilisateur.

vue.SessionTab.displaySession()

Parcours la liste des sessions (dans l'attribut session) et les affiches à l'utilisateur.

vue.SessionTab.writeErrorMessage(String)

Prend en entrée une chaîne de caractère qui représente un message d'erreur ou de succès et affiche une message plus descriptif que ce dernier.

vue.SessionTab.setNewSession(Session)

Prend en entrée un objet de type session et l'ajoute à l'attribut session. (qui est une liste de sessions)

vue.SessionTab.setDeleteSession(int)

Supprime la session présente à l'indice obtenu en paramètre.

vue.SessionTab.getDataUeList()

Retourne l'attribut ue.

vue.SessionTab.getDataClasseList()

Retourne l’attribut classe.

vue.SessionTab.getDataCreneauList()

Retourne l’attribut Creneau.

vue.SessionTab.getDataSessionList()

Retourne l’attribut session.

vue.SessionTab.getAddSession()

Retourne l’attribut ajouter

vue.SessionTab.getDeleteSession()

Retourne l’attribut supprimer.

vue.SessionTab.getIndexListUE()

Retourne un entier qui représente l’indice de l’ue sélectionnée.

vue.SessionTab.getIndexListClasse()

Retourne un entier qui représente l’indice de la classe sélectionnée.

vue.SessionTab.getIndexListCreneau()

Retourne un entier qui représente l’indice du créneau sélectionnée.

vue.SessionTab.getIndexListSession()

Retourne un entier qui représente l’indice de la session sélectionnée.

Classe UETab

Attributs

```
private ArrayList<UE> ue = new ArrayList<UE>();
private JPanel panel;
private JLabel sigle = new JLabel("Sigle");
private JLabel nomination = new JLabel("Nomination");
private JTextArea sigleText = new JTextArea();
private JTextArea nominationText = new JTextArea();
JSplitPane splitPane;
JPanel panel1 = new JPanel();
JPanel panel2 = new JPanel();
JButton createButton = new JButton("ajouter ue");
GridBagConstraints gc = new GridBagConstraints();
```

vue.UETab.UETab(JPanel)

Prend en entrée un objet de type JPanel et met la valeur de cet objet dans l'attribut panel.
Et initialise le composant.

vue.UETab.initComponent()

Ajoute les UE dans l'attribut ue de l'objet. De plus, cette méthode spécifie les dimensions de la fenêtre UE et les caractéristiques de celle-ci.

vue.UETab.displayUE()

Cette méthode parcourt la liste des UE qui est en attribut ue et l'affiche sur cette fenêtre.

vue.UETab.writeErrorMessage()

Retourne en sortie un message d'erreur dans un message dialog qui dit " impossible de créer une UE. "

vue.UETab.setNewUE(Ue)

Prend en entrée un objet de type UE et l'ajoute dans l'attribut ue. (liste de UE)

vue.UETab.setDeleteUE(int)

Prend un entier en entrée qui représente un indice et supprime l'ue présente à cet indice dans l'attribut ue.

vue.UETab.getAddUE()

Retourne l'attribut createButton

vue.UETab.getDeleteUE()

Retourne l'attribut deleteButton

vue.UETab.getUEsigle()

Retourne le texte de l'attribut sigleText.

vue.UETab.getUEnomination()

Retourne le texte de l'attribut nomination .

vue.UETab.getIndexListUE()

Retourne l'indice de l'ue choisi dans la listUE.

vue.UETab.getDataUEList()

Retourne l'attribut ue.

Environnement matériel nécessaire à l'installation

Cpu 64 bit

Environnement logiciel nécessaire à l'installation

Version de Java Development kit (JDK) : jdk-11.0.11
Version de Java runtime environment (JRE) : jre1.8.0_291

Cheminement des appels de fonctions pour 2 fonctionnalités

Suppression d'une UE

- Appelle la méthode vue.UETab.getIndexListUE() pour savoir l'indice de l'UE choisit dans la liste des UEs
- Appelle la méthode controlleur.Parser.remove(String, Class<?>) pour retirer l'UE des fichiers de sauvegarde de données.
- Met à jour l'onglet des UE à l'aide de la méthode vue.UETab.setDeleteUE(int) et vue.UETab.displayUE()
- Met à jour l'onglet de session à l'aide de la méthode java.util.ArrayList.remove(Object) et vue.SessionTab.displayUe().

Suppression d'un créneau

- Appelle la méthode vue.CreneauTab.getIndexListCreneau() pour retourner l'indice du créneau choisie dans la liste des créneaux.
- Appelle la méthode controlleur.Parser.remove(String, Class<?>) pour retirer le créneau des fichiers de sauvegarde de données.
- Met à jour l'onglet des créneaux à l'aide de la méthode vue.CreneauTab.setDeleteCreneau(int) et vue.CreneauTab.displayCreneau()
- Met à jour l'onglet de session à l'aide de la méthode java.util.ArrayList.remove(Object) et vue.SessionTab.displayCreneau().

Liste des exigences satisfaites

- Le système fonctionne sous linux
- Le système fonctionne sous windows
- Le système fonctionne sous Macos
- Le système est développé sous java 11
- La livraison contient tous les éléments nécessaires à la génération de la version binaire.
- La livraison contient la version binaire du système.
- La livraison contient toute la documentation
- Le système est constitué d'un seul exécutable.
- Le programme principale instancie deux objets : l'un implémentant l'ihm et l'autre exposant les fonctionnalités.
- Les communications entre ihm et les fonctions passe par une unique interface java.
- Les requêtes vont uniquement de l'ihm vers l'objet exposant les fonctions.
- Les opérations exposées par l'interface sont de 4 types : lecture, création, modification, suppression d'un objet.
- Les classes implémentant l'IHM et les fonctionnalités sont packagées dans des jar distincts
- L'identifiant des objets créés est attribué par l'IHM.
- Après redémarrage du système il est dans le même état qu'avant son arrêt (données)
- Il est possible de mettre à jour l'IHM (automatiquement ou à la demande de l'utilisateur)
- L'utilisateur peut créer une unité d'enseignement
- L'utilisateur peut supprimer une unité d'enseignement.
- L'utilisateur peut créer un créneau
- L'utilisateur peut supprimer un créneau.
- L'utilisateur peut créer une classe
- L'utilisateur peut supprimer une classe
- L'utilisateur peut créer une session.
- L'utilisateur peut supprimer une session.
- L'utilisateur peut changer une session de créneau temporel.
- L'utilisateur peut créer des sessions sur un ensemble de créneaux.

Liste des exigences non satisfaites

Opérations de modifications .

Les anomalies

Procedure de compilation

Le lancement du programme sur eclipse génère les fichiers .class.