

# Itération 3 : Vue Kanban + Dépendances + Gestion des jours + gestion de l'état de la tâche

## Travail dans la branche `Kanban`

Pour tester, on ouvre `MainKanban` qui lance une vue Kanban.

### CRUD Tache

- **Lecture/Affichage** : Mise à jour de `VueKanban` pour afficher les nouvelles informations (Jour, État sous forme de pastille).
- **Mise à jour (Édition)** : Refonte de `VueEditeurTache` (ajout sélecteurs Colonne, Jour, État) et mise à jour de `ControleurSauvegarderModif` pour persister les changements dans l'objet.
- **Archivage** : Utilisation de `ControleurArchiverTache` et filtrage dans `Modele` (`getTaches()` ignore les archivées).

### CRUD Colonne

- **Modèle** : Ajout des méthodes `renommerColonne` et `supprimerColonne` dans `Modele.java` (avec gestion de sécurité : les tâches d'une colonne supprimée vont dans "À faire").
- **Contrôleurs** : Création de `ControleurAjouterColonne`, `ControleurRenommerColonne` et `ControleurSupprimerColonne`.
- **Vue** : Adaptation de l'en-tête dans `VueKanban` pour inclure les boutons d'actions (Renommer/Supprimer) à côté du titre.

### Gestion des dépendances :

- **Architecture** : Suppression du Pattern Composite strict (suppression des fichiers `TacheSimple` et `TacheComposite`). La classe `Tache` devient concrète et intègre une liste `enfants` et les méthodes de gestion (`ajouterEnfant`, `aDesEnfants`).
- **Création** : Ajout d'une `ComboBox` dans la `Dialog` de `ControleurCreerTache` listant toutes les tâches existantes pour rattacher la nouvelle tâche à un parent.
- **Affichage** : Modification de la méthode `creerCarteTache` dans `VueKanban` pour itérer et afficher visuellement les sous-tâches dans la carte du parent.

### Dates → Jours de semaine

Remplacement de la gestion complexe `LocalDate` par une approche catégorielle (Jours de la semaine).

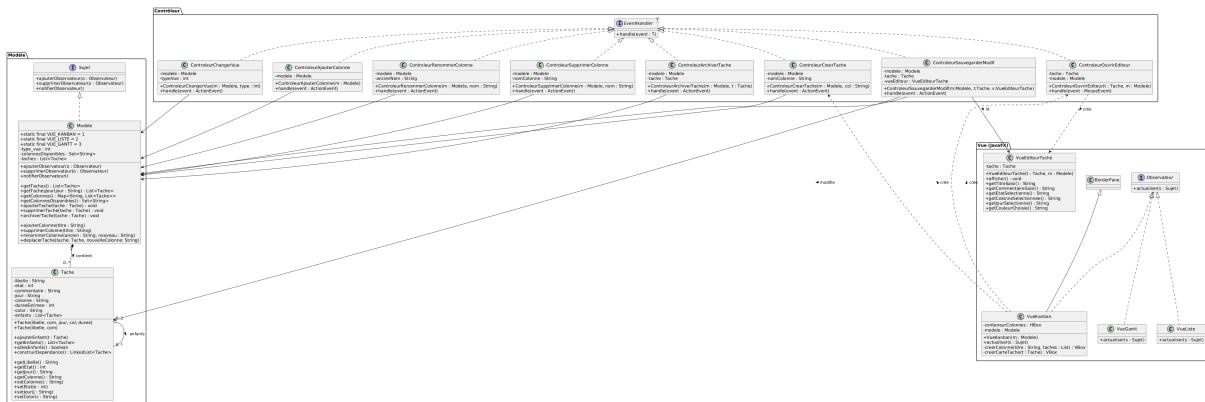
- **Modèle** : Dans `Tache`, remplacement des attributs `dateDebut` / `dateFin` par un String `jour` (validé par un Set static `JOURS_AUTORISES`).
- **Contrôleurs & Vue** : Adaptation de `VueEditeurTache` (remplacement `DatePicker` par `ComboBox`), de `ControleurSauvegarderModif` et de l'affichage dans `VueKanban`.

### Début de drag & drop (fonctionne seulement en interaction avec une autre Tâche)

**Interaction** : `VueKanban` gère les événements `DragDetected`, `DragOver` et `DragDropped` pour transférer la référence de l'objet `Tache`.

Ajout d'une colonne par défaut appellée `Principale`

## Diagramme de classe



@startuml

'--- STYLE & CONFIGURATION ---

```
skinparam classAttributeIconSize 0
skinparam componentStyle uml2
```

'--- PACKAGE MODÈLE ---

```
package "Modèle" {
    interface Sujet {
        + ajouterObservateur(o : Observateur)
        + supprimerObservateur(o : Observateur)
        + notifierObservateur()
    }

    class Tache {
        - libelle : String
        - etat : int
        - commentaire : String
        - jour : String
        - colonne : String
        - dureeEstimee : int
        - color : String
        - enfants : List<Tache>

        '--- Constructeurs ---
        + Tache(libelle, com, jour, col, duree)
        + Tache(libelle, com)

        '--- Méthodes ---
        + ajouterEnfant(t : Tache)
        + getEnfants() : List<Tache>
        + aDesEnfants() : boolean
        + construireDependance() : LinkedList<Tache>

        '--- Getters/Setters ---
        + getLibelle() : String
        + getEtat() : int
        + getJour() : String
        + getColonne() : String
        + setColonne(c : String)
        + setEtat(e : int)
        + setJour(j : String)
    }
}
```

```

+ setColor(c : String)
}

class Modele implements Sujet {
+ static final VUE_KANBAN = 1
+ static final VUE_LISTE = 2
+ static final VUE_GANTT = 3
- type_vue : int
- colonnesDisponibles : Set<String>
- taches : List<Tache>

+ ajouterObservateur(o : Observateur)
+ supprimerObservateur(o : Observateur)
+ notifierObservateur()

'--- Gestion Données ---
+ getTaches() : List<Tache>
+ getTachesJour(jour : String) : List<Tache>
+ getColonnes() : Map<String, List<Tache>>
+ getColonnesDisponibles() : Set<String>
+ ajouterTache(tache : Tache) : void
+ supprimerTache(tache : Tache) : void
+ archiverTache(tache : Tache) : void

'--- Gestion Structure ---
+ ajouterColonne(titre : String)
+ supprimerColonne(titre : String)
+ renommerColonne(ancien : String, nouveau : String)
+ deplacerTache(tache: Tache, nouvelleColonne: String)
}

'--- PACKAGE VUE (JavaFX) ---
package "Vue (JavaFX)" {
interface Observateur {
+ actualiser(s : Sujet)
}

' --- VUES MÉTIER ---

class VueKanban extends BorderPane implements Observateur {
' Gestion interne des colonnes via des VBox standards
- conteneurColonnes : HBox
- modele : Modele
+ VueKanban(m : Modele)
+ actualiser(s : Sujet)
- creerColonne(titre : String, taches : List) : VBox
- creerCarteTache(t : Tache) : VBox
}

class VueListe implements Observateur {
+ actualiser(s : Sujet)
}

class VueGantt implements Observateur {
}

```

```

+ actualiser(s : Sujet)
}

' --- COMPOSANTS UNITAIRES ---

' Popup d édition
class VueEditeurTache {
    - tache : Tache
    + VueEditeurTache(t : Tache, m : Modele)
    + afficher() : void
    + getTitreSaisi() : String
    + getCommentaireSaisi() : String
    + getEtatSelectionne() : String
    + getColonneSelectionnee() : String
    + getJourSelectionne() : String
    + getCouleurChoisie() : String
}
}

' --- PACKAGE CONTRÔLEUR ---
package "Contrôleur" {

interface EventHandler<T> {
    + handle(event : T)
}

' -- Actions Globales --
class ControleurCreerTache implements EventHandler {
    - modele : Modele
    - nomColonne : String
    + ControleurCreerTache(m : Modele, col : String)
    + handle(event : ActionEvent)
}

class ControleurAjouterColonne implements EventHandler {
    - modele : Modele
    + ControleurAjouterColonne(m : Modele)
    + handle(event : ActionEvent)
}

class ControleurRenommerColonne implements EventHandler {
    - modele : Modele
    - ancienNom : String
    + ControleurRenommerColonne(m : Modele, nom : String)
    + handle(event : ActionEvent)
}

class ControleurSupprimerColonne implements EventHandler {
    - modele : Modele
    - nomColonne : String
    + ControleurSupprimerColonne(m : Modele, nom : String)
    + handle(event : ActionEvent)
}

class ControleurChangerVue implements EventHandler {
}
}

```

```

- modele : Modele
- typeVue : int
+ ControleurChangerVue(m : Modele, type : int)
+ handle(event : ActionEvent)
}

'-- Actions sur Tâche --
class ControleurOuvrirEditeur implements EventHandler {
- tache : Tache
- modele : Modele
+ ControleurOuvrirEditeur(t : Tache, m : Modele)
+ handle(event : MouseEvent)
}

class ControleurArchiverTache implements EventHandler {
- modele : Modele
- tache : Tache
+ ControleurArchiverTache(m : Modele, t : Tache)
+ handle(event : ActionEvent)
}

class ControleurSauvegarderModif implements EventHandler {
- modele : Modele
- tache : Tache
- vueEditeur : VueEditeurTache
+ ControleurSauvegarderModif(m:Modele, t:Tache, v:VueEditeurTache)
+ handle(event : ActionEvent)
}
}

```

#### '--- RELATIONS ---'

' MVC - Modèle  
' Relation réflexive pour les sous-tâches (remplace Composite)  
Tache "0..1" o-- "0..\*" Tache : enfants >  
Modele "1" \*-- "0..\*" Tache : contient >

' Contrôleurs → Modèle  
ControleurCreerTache → Modele  
ControleurAjouterColonne → Modele  
ControleurRenommerColonne → Modele  
ControleurSupprimerColonne → Modele  
ControleurChangerVue → Modele  
ControleurArchiverTache → Modele  
ControleurSauvegarderModif → Modele

' Contrôleurs → Vues / Taches  
ControleurOuvrirEditeur ..> VueEditeurTache : créée >  
ControleurSauvegarderModif → VueEditeurTache : lit >  
ControleurSauvegarderModif → Tache : modifie >  
VueKanban ..> ControleurCreerTache : créée >  
VueKanban ..> ControleurOuvrirEditeur : créée >

@enduml