

TD0 : Révisions

Encapsulation, construction

@startuml

title Diagramme de classes - Gestion de caserne de pompiers (V1)

```
class Pompier {  
- nom: String  
- prenom: String  
- matricule: String  
- grade: String  
- anneesService: int  
- disponible: boolean  
--  
+ Pompier()  
+ Pompier(nom: String, prenom: String, matricule: String)  
+ Pompier(nom: String, prenom: String, matricule: String, grade: String, anneesService: int)  
--  
+ getNom(): String  
+ getPrenom(): String  
+ getMatricule(): String  
+ getGrade(): String  
+ getAnneesService(): int  
+ isDisponible(): boolean  
+ setGrade(grade: String): void  
+ setAnneesService(anneesService: int): void  
+ setMatricule(matricule: String): void  
+ augmenterAnciennete(): void  
+ augmenterAnciennete(nbAnnees: int): void  
+ augmenterAnciennete(nbAnnees: int, promotion: boolean): void  
+ toString(): String  
}
```

```
class Caserne {  
- nomCaserne: String  
- pompiers: List  
--  
+ Caserne(nomCaserne: String)  
--  
+ ajouterPompier(p: Pompier): void  
+ nombrePompiersDisponibles(): int  
+ afficherEffectif(): void  
+ rechercherParMatricule(matricule: String): Pompier  
}
```

Caserne "1" o-- "0..*" Pompier

note on link

Agrégation : le Pompier existe indépendamment de la Caserne

(pas de dépendance dans le constructeur).
end note

@enduml

Associations

@startuml

title Diagramme de classes - Gestion de caserne de pompiers (V2)

```
enum TypeSinistre {  
    INCENDIE  
    ACCIDENT  
    INONDATION  
    FUITE_GAZ  
    SAUVETAGE  
}
```

```
enum NiveauGravite {  
    FAIBLE  
    MODERE  
    ELEVE  
    CRITIQUE  
}
```

```
class Pompier {  
    - nom: String  
    - prenom: String  
    - matricule: String  
    - grade: String  
    - anneesService: int  
    - disponible: boolean  
    --  
    + Pompier()  
    + Pompier(nom: String, prenom: String, matricule: String)  
    + Pompier(nom: String, prenom: String, matricule: String, grade: String, anneesService: int)  
    --  
    + getNom(): String  
    + getPrenom(): String  
    + getMatricule(): String  
    + getGrade(): String  
    + getAnneesService(): int  
    + isDisponible(): boolean  
    + setGrade(grade: String): void  
    + setAnneesService(anneesService: int): void  
    + setDisponible(disponible: boolean): void  
    + augmenterAnciennete(): void  
    + augmenterAnciennete(nbAnnees: int): void  
    + augmenterAnciennete(nbAnnees: int, promotion: boolean): void  
    + afficherFiche(): void  
    + toString(): String
```

```
}
```

```
class Caserne {  
- nomCaserne: String  
- pompiers: List  
--  
+ Caserne(nomCaserne: String)  
--  
+ ajouterPompier(p: Pompier): void  
+ retirerPompier(matricule: String): void  
+ nombrePompiersDisponibles(): int  
+ afficherEffectif(): void  
+ rechercherParMatricule(matricule: String): Pompier  
}
```

```
class Location {  
- adresse: String  
- ville: String  
- codePostal: String  
- latitude: double  
- longitude: double  
--  
+ Location(adresse: String, ville: String)  
+ Location(adresse: String, ville: String, codePostal: String)  
--  
+ getAdresse(): String  
+ getVille(): String  
+ getCodePostal(): String  
+ getLatitude(): double  
+ getLongitude(): double  
+ toString(): String  
}
```

```
class Sinistre {  
- type: TypeSinistre  
- niveauGravite: NiveauGravite  
- dateDeclaration: LocalDateTime  
- location: Location  
--  
+ Sinistre(type: TypeSinistre, niveauGravite: NiveauGravite, location: Location)  
--  
+ getType(): TypeSinistre  
+ getNiveauGravite(): NiveauGravite  
+ getDateDeclaration(): LocalDateTime  
+ getLocation(): Location  
+ toString(): String  
}
```

```
class Intervention {  
- dateDebut: LocalDateTime  
- dateFin: LocalDateTime  
- sinistre: Sinistre  
- pompiersMobilises: List  
--  
+ Intervention(sinistre: Sinistre)  
--
```

```
+ ajouterPompier(p: Pompier): void
+ terminerIntervention(): void
+ dureeIntervention(): long
+ afficherRapport(): void
}
```

Caserne "1" o-- "0..*" Pompier

note on link

Agrégation : le Pompier existe indépendamment de la Caserne
(pas de dépendance dans le constructeur).

end note

Sinistre "1" *-- "1" Location

note on link

Composition : la Location n'a de sens
qu'à travers le Sinistre.

end note

Sinistre "1" --> "1" TypeSinistre

Sinistre "1" --> "1" NiveauGravite

Intervention "1" o-- "1" Sinistre

note on link

Agrégation : le Sinistre peut exister
avant qu'une intervention soit déclenchée.

end note

Intervention "1" o-- "0..*" Pompier

note on link

Agrégation : les pompiers mobilisés
existent indépendamment de l'intervention.

end note

@enduml

Descriptif des méthodes - Location, Sinistre, Intervention

Location

Constructeurs

- Location(adresse, ville) : constructeur simplifié, codePostal initialisé à "", **latitude/longitude** à 0.0.
- Location(adresse, ville, codePostal) : constructeur complet (sans coordonnées GPS pour rester simple).

Getters

Un par attribut (`getAdresse()`, `getVille()`, `getCodePostal()`, `getLatitude()`, `getLongitude()`). Pas de setters : une fois créée, une localisation ne change pas.

toString()

Retourne une chaîne lisible, ex : "12 rue des Lilas, 75001 Paris".

Sinistre

Constructeur

- `Sinistre(type, niveauGravite, location)` : initialise `dateDeclaration` automatiquement à `LocalDateTime.now()`.

Getters

- `getType()`, `getNiveauGravite()`, `getDateDeclaration()`, `getLocation()`.

Pas de setters : un sinistre déclaré ne change pas de nature.

toString()

Doit afficher une synthèse : type, gravité, date, localisation. Ex :

[INCENDIE] Gravité : ELEVE - déclaré le 21/08/2026 à 10:32 - 12 rue des Lilas, Paris

Intervention

Constructeur

- `Intervention(sinistre)` : initialise `dateDebut` à `LocalDateTime.now()`, `dateFin` à null, `pompierMobilises` à une liste vide.

ajouterPompier(Pompier p): void

Ajoute un pompier à la liste `pompierMobilises`, mais uniquement s'il est disponible (`p.isDisponible()`). Si non disponible, afficher un message d'erreur (ou lever une exception si vous voulez aller plus loin). Une fois ajouté, on peut aussi le passer à `disponible = false`.

terminerIntervention(): void

- Fixe `dateFin` à `LocalDateTime.now()`.

- Repasse chaque pompier de `pompierMobilises` à `disponible = true`.
- Éventuellement afficher un message de confirmation.

dureeIntervention(): long

Calcule la durée en minutes entre `dateDebut` et `dateFin` (utiliser `Duration.between(...).toMinutes()`). Si `dateFin == null`, on peut soit calculer par rapport à `LocalDateTime.now()` (intervention en cours), soit renvoyer -1 ou lever une exception.

afficherRapport(): void

Affiche une synthèse complète de l'intervention : sinistre concerné, dates début/fin, durée, liste des pompiers mobilisés (nom, prénom, matricule).

Héritage

@startuml

title Diagramme de classes - Gestion de caserne de pompiers (V3)

```

abstract class Pompier {
- matricule: String
- nom: String
- prenom: String
- grade: String
- disponible: boolean
--
+ Pompier(matricule: String, nom: String, prenom: String, grade: String)
--
+ isDisponible(): boolean
+ setDisponible(disponible: boolean): void
+ getMatricule(): String
+ getNom(): String
+ getPrenom(): String
+ getGrade(): String
+ toString(): String
}

class PompierProfessionnel {
- salaireBase: double
--
+ PompierProfessionnel(matricule: String, nom: String, prenom: String, grade: String, salaireBase: double)
--
+ getSalaireBase(): double
}

class PompierVolontaire {
- employeur: String
- heureDebutTravail: LocalTime

```

```

- heureFinTravail: LocalTime
- quotaHeuresMensuel: int
- heuresEffectueesMoisCourant: int
--
+ PompierVolontaire(matricule: String, nom: String, prenom: String, grade: String, employeur: String,
heureDebutTravail: LocalTime, heureFinTravail: LocalTime, quotaHeuresMensuel: int)
--
+ isDisponible(): boolean
+ ajouterHeuresIntervention(heures: int): void
- estEnHorairesTravail(): boolean
+ getEmployeur(): String
+ getHeuresEffectueesMoisCourant(): int
}

```

```
Pompier <|-- PompierProfessionnel
```

```
Pompier <|-- PompierVolontaire
```

```

class Caserne {
- nomCaserne: String
- pompiers: List
--
+ Caserne(nomCaserne: String)
--
+ ajouterPompier(p: Pompier): void
+ nombrePompiersDisponibles(): int
+ afficherEffectif(): void
+ rechercherParMatricule(matricule: String): Pompier
}

```

```
Caserne "1" o-- "0..*" Pompier
```

```
note on link
```

```
Agrégation : le Pompier existe indépendamment de la Caserne
(pas de dépendance dans le constructeur).
```

```
end note
```

```
@enduml
```

Evolutions

Encapsulation

- Ajoutez à Pompier un attribut privé dateEmbauche: LocalDate, avec un getter getAnciennete(): int qui calcule le nombre d'années depuis dateEmbauche (pas de setter pour anciennete — elle ne doit jamais être stockée directement).
- Ajoutez une méthode distanceVers(autre: Location): double dans Location, utilisant la formule de distance euclidienne simplifiée sur latitude/longitude. Utilisez-la dans Caserne pour écrire une méthode casernePlusProche(sinistre: Sinistre): boolean (nécessite que Caserne ait elle-même une Location — à ajouter).

Héritage

Ajoutez à PompierProfessionnel une redéfinition de méthode manquante : une méthode

calculerSalaireMensuel(): double dans Pompier (retournant 0 par défaut ou étant abstraite), redéfinie dans :

- PompierProfessionnel → salaireBase + prime d'ancienneté (2% par année de service)
- PompierVolontaire → heuresEffectueesMoisCourant * tauxHoraire (il faudra ajouter tauxHoraire)

Créer une classe Officier héritant de PompierProfessionnel avec :

- un attribut primeCommandement: double
- une redéfinition de calculerSalaireMensuel() qui appelle super.calculerSalaireMensuel() et ajoute la prime
- une redéfinition de toString() qui appelle super.toString() et ajoute la mention du grade d'officier

Ajoutez à Pompier une méthode abstraite peutIntervenirSur(type: TypeSinistre): boolean. Chaque sous-classe la redéfinit :

- PompierProfessionnel → toujours true
- PompierVolontaire → true sauf pour FUITE_GAZ (compétence non acquise, à simuler par une contrainte simple)
- Officier → toujours true, et en plus logue dans la console qu'un officier prend le commandement

Utilisez cette méthode dans Intervention.ajouterPompier() pour refuser l'ajout d'un pompier incompetent sur ce type de sinistre.

From:

<http://slamwiki2.kobject.net/> - **SlamWiki 2.1**

Permanent link:

<http://slamwiki2.kobject.net/sio/bloc2/2a/td0>

Last update: **2026/08/26 10:31**

