Exercise — War

version **#1.0.0**

Yet Another Kind of Assistants 2026 <assistants@tickets.assistants.epita.fr>

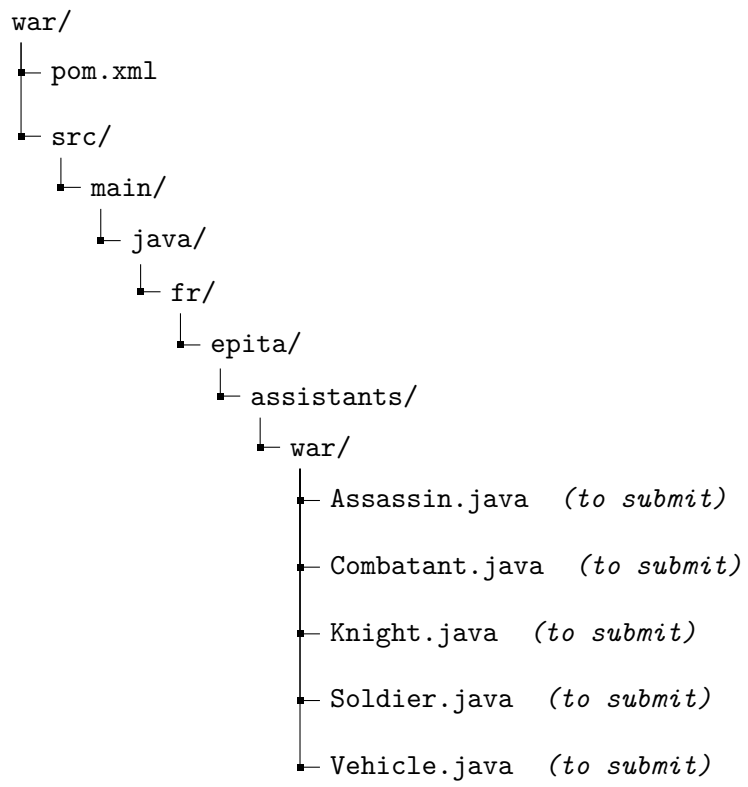
Copyright

This document is for internal use at EPITA (website) only.

Contents

*https://intra.forge.epita.fr

File Tree



Authorized imports

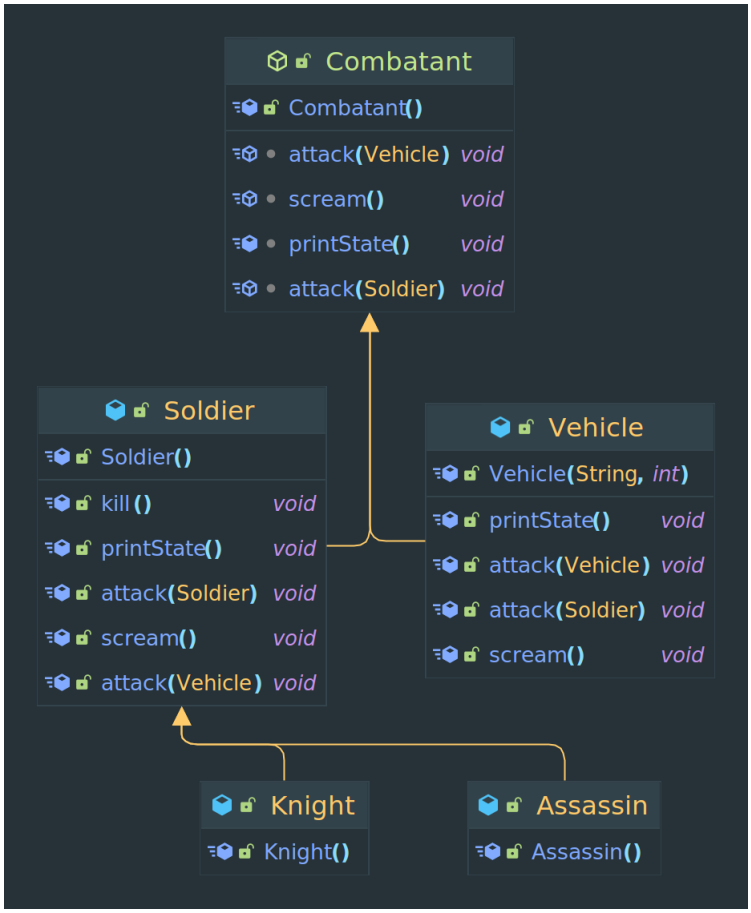
- java.*

1 History

The war has been declared! The enemy is close and you have to defend your kingdom. You have to raise an army in order to save your people's life.

2 Goal

You have to implement the classes shown in the image below.



3 Specifications

For this exercise, all text outputs have to be followed by a line break.

3.1 Soldiers

You have to implement the `Soldier` class. A soldier has a number of health points (`int`), damage points (`int`) and a scream (`String`). This class is inherited by the `Knight` and `Assassin` classes which are specialized soldiers.

Each kind of soldier should have the following characteristics when instanciated:

Stat	Soldier	Knight	Assassin
Health points	15	20	10
Damage points	3	5	9
Scream	“No pity for losers!”	“Be quick or be dead!”	“Out of the shadows!”

Note: Health points can become negative.

You have to implement the constructors of the classes in each corresponding file:

```
public Soldier() { /* ... */ }
public Assassin() { /* ... */ }
public Knight() { /* ... */ }
```

In the `Soldier` class, implement the following method:

```
public void kill() { /* ... */ }
```

This method will put the health points of the soldier to 0.

3.2 Vehicle

You have a developed kingdom, so you can have some vehicles. Implement the `Vehicle` class. A vehicle has a model name (`String`) and some defense points (`int`).

You have to implement the constructor of the class:

```
public Vehicle(String name, int defense) { /* ... */ }
```

3.3 Combatant

You now have an army but it cannot attack. Implement the `Combatant` abstract class to change that. The `Soldier` and `Vehicle` classes extend it. The class should have the following methods:

```
void printState() { /* ... */ }
abstract void attack(Soldier s);
abstract void attack(Vehicle v);
abstract void scream();
```

- The `printState` method will display on the standard output the current state of the fighter. By default it shows the following value on the error output:

  ```
  Error 404. Class not found.
  ```

- The `attack` method will make the fighter attack a soldier or a vehicle.
- The `scream` method will display on the standard output the scream of the soldier or vehicle (yes, vehicles can scream).

In the `Soldier` class you have to override the `Combatant` methods:

- The `printState` method will display on the standard output the current health points of your soldier in the format shown in the examples below.
- The `attack` method, against another soldier, will make your soldier attack the soldier passed as argument thus decreasing their health points by the amount of your damage points.
- The `attack` method, against a vehicle, will simply display on the standard output:

```
I can't fight this.
```

- The `scream` method, will display on the standard output the soldier scream.

In the `Vehicle` class, you have to override the `Combatant` methods:

- The `printState` method will display on the standard output the current defense points of your vehicle in the format shown in the examples below.
- The `attack` method, against a soldier, will put the health points of the soldier to 0.
- The `attack` method, against another vehicle, will halve the defense points of the other vehicle.
- The `scream` method, will display on the standard output the vehicle name of your vehicle in the format shown in the examples below.

4 Example

The following code:

```java
public static void main(String[] args) {
  Soldier s1 = new Soldier();
  Knight k1 = new Knight();
  Assassin a1 = new Assassin();
  Vehicle v1 = new Vehicle("M47 Patton", 500);

  s1.scream();
  k1.scream();
  a1.scream();
  v1.scream();
}
```

will result to:

```
No pity for losers!
Be quick or be dead!
Out of the shadows!
I'm M47 Patton!
```

The following example:

```java
public static void main(String[] args) {
  Soldier s1 = new Soldier();
  Knight k1 = new Knight();
  Assassin a1 = new Assassin();
  Vehicle v1 = new Vehicle("M47 Patton", 500);

  s1.printState();
  k1.printState();
  a1.printState();
  v1.printState();
}
```

will result to:

```
I have 15 health points.
I have 20 health points.
I have 10 health points.
I have 500 defense points.
```

The following example:

```java
public static void main(String[] args) {
  Soldier s1 = new Soldier();
  Knight k1 = new Knight();
  Assassin a1 = new Assassin();
  Vehicle v1 = new Vehicle("M47 Patton", 500);
  Vehicle v2 = new Vehicle("T-54", 500);

  s1.attack(k1);
  k1.attack(a1);
  a1.attack(s1);
  v1.attack(v2);

  s1.printState();
  k1.printState();
  a1.printState();
  v1.printState();
  v2.printState();
}
```

```
I have 6 health points.
I have 17 health points.
I have 5 health points.
I have 500 defense points.
I have 250 defense points.
```

Being a hero means fighting back even when it seems impossible.