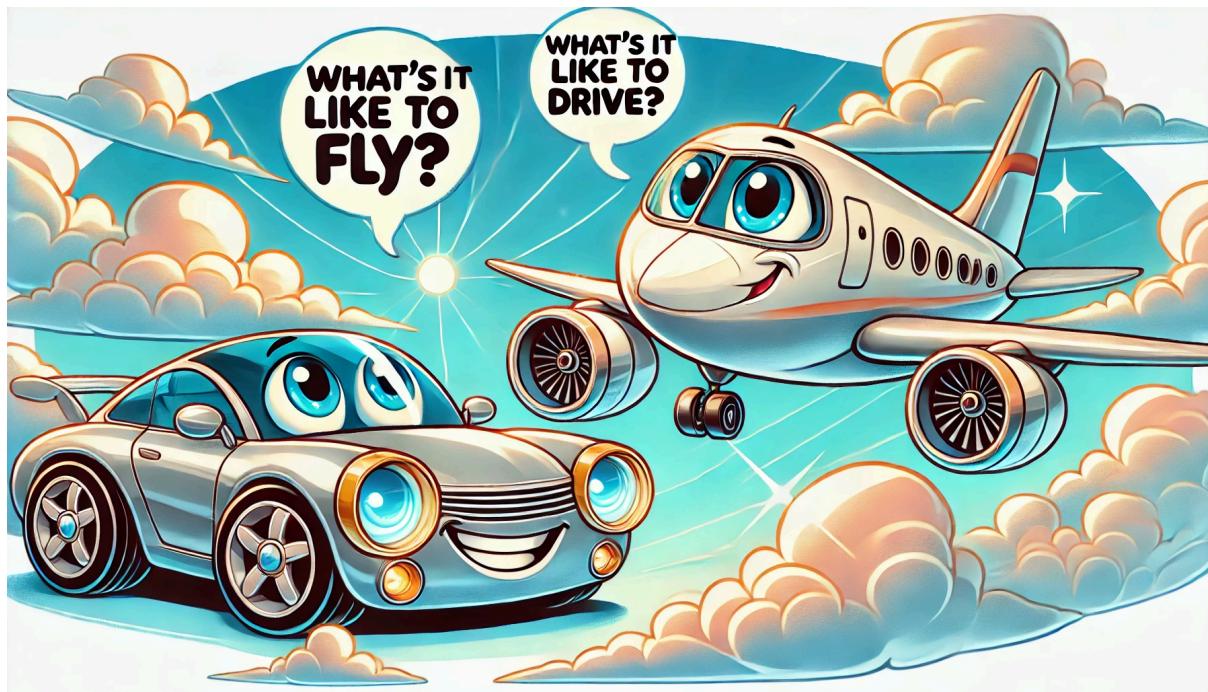


# ✈️ L'histoire de la fusion Avion/Voiture 🚗



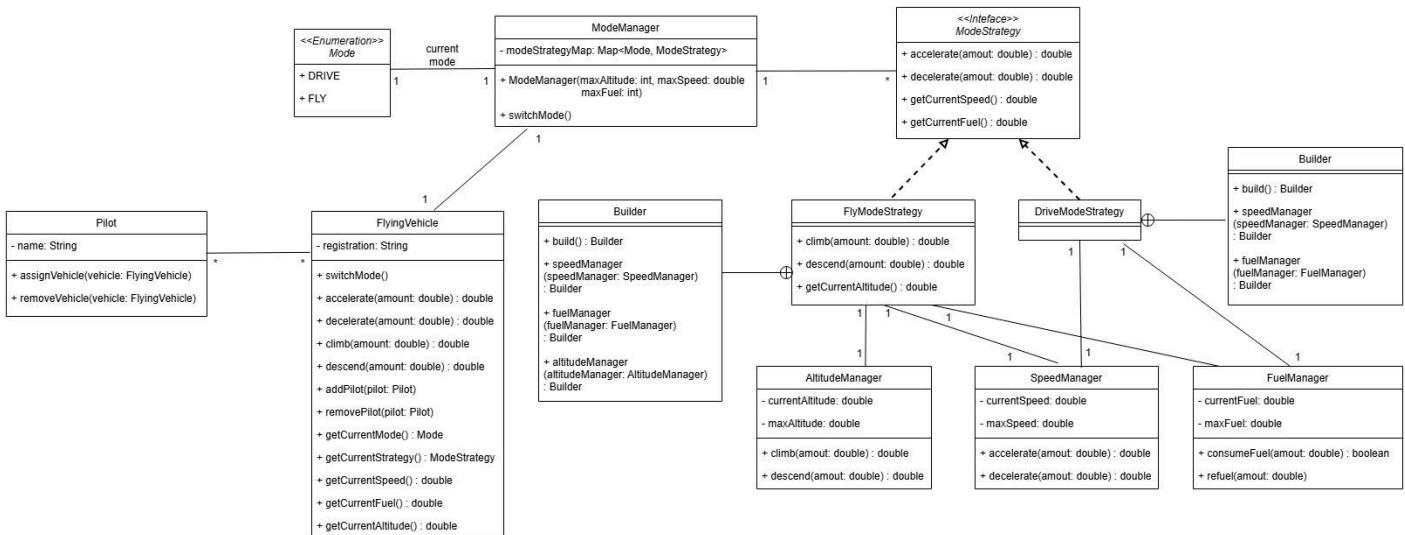
## L'épopée de la Voiture Volante

« Que dirais-tu de survoler les montagnes, les océans et les vallées ? » demanda l'avion. La voiture, intriguée, répondit : « Et toi, que penserais-tu de conduire sur des routes sinueuses, à travers des paysages que tu ne peux deviner d'en haut ? »

Ainsi naquit une idée brillante : combiner leurs forces pour donner naissance à une voiture volante. Mais comment fusionner des concepts si différents ? La réponse était simple : les design pattern !

## L'assemblage des forces : la construction de la voiture volante

Pour rendre cette vision possible, nous avons combiné les fonctionnalités de l'avion et de la voiture dans une classe baptisée `FlyingVehicle`. Voici le diagramme de classe décrivant cette fusion :



Tels des ingénieurs visionnaires, nous avons organisé notre code autour de deux design patterns majeurs : le **Strategy** et le **Builder**, accompagné d'une touche de **Factory Method**

### Strategy : des ailes rétractables

Pour permettre à la voiture volante de changer de mode, nous avons défini une interface **ModeStrategy**. Cette interface permet de gérer les comportements spécifiques à chaque mode, comme l'accélération, la décélération et la consommation de carburant. Deux implémentations principales s'en dégagent :

1. **DriveModeStrategy** : Adapté à la conduite sur route, ce mode est économique en carburant.
2. **FlyModeStrategy** : Ce mode ajoute la capacité de monter (« climb ») et descendre (« descend »), nécessaire pour naviguer dans les airs.

Ces stratégies sont encapsulées dans la classe **ModeManager**, qui agit comme un cerveau central, capable de basculer entre les modes à la demande.

### Builder : L'artisan de la précision

Pour que chaque mode soit configuré avec soin, nous avons utilisé le design pattern **Builder** dans nos stratégies. Voici comment il fonctionne :

- Chaque **ModeStrategy** est créée à l'aide d'un constructeur étendu, qui accepte des managers d'altitude, de carburant et de vitesse.
- Ce processus garantit que chaque composant critique soit initialisé avant l'utilisation.

Par exemple, le `FlyModeStrategy.Builder` exige un `AltitudeManager` pour gérer les altitudes, un `SpeedManager` pour les vitesses, et un `FuelManager` pour la gestion du carburant. Sans ces composants, le vol est impossible.

### Factory Method : La ligne de production

Le choix entre les modes « Fly » et « Drive » est régi par une sorte de **Factory Method** dans `ModeManager`. Cette approche assure une instantiation simple et efficace des stratégies sans avoir du code redondant.

## Une histoire de collaboration : la magie des patterns

Un jour, la voiture volante se prépara à effectuer son premier test grandeur nature. Tout semblait parfait, mais à mi-chemin sur une route de montagne, le chemin se termina abruptement devant une falaise. Elle s'arrêta net, perplexe.

« Que faire maintenant ? » pensa-t-elle. Mais c'est alors que ses ailes rétractables se déployèrent gracieusement. En activant le mode « Fly », elle prit son envol et survola la vallée avec une aisance qui étonna même l'avion de ligne qui était de passage.



Dans les airs, elle affronta les vents et gravit les hauteurs. Grâce à sa stratégie adaptative, elle géra la montée et la descente avec précision, tout en consommant son carburant de manière équilibrée. Ce passage fluide entre la route et le vol incarnait la collaboration

parfaite entre les composants du **ModeManager**. La voiture volante avait prouvé qu'elle était prête à relever tous les défis.

## Tests unitaires et fonctionnels : garantir la fiabilité

Pour nous assurer que notre voiture volante fonctionne parfaitement, nous avons conçu une série de tests unitaires et fonctionnels. Ces tests couvrent les scénarios critiques pour les modes « Drive » et « Fly », garantissant la fiabilité de ses performances.

### Tests fonctionnels

- **Gestion de l'accélération et de la montée**

- *User Story:* En tant que pilote, je veux pouvoir accélérer et monter en mode vol pour contrôler efficacement la vitesse et l'altitude.
- *Exemple:* En mode « Fly », si l'altitude initiale est de 1000 mètres et que le pilote monte de 500 mètres, l'altitude finale doit être de 1500 mètres.

```
# Scenario: Accelerate in Drive mode
# Description: Verifies that the vehicle can accelerate while in
Drive mode.

Scenario Outline: Accelerate the vehicle in Drive mode
  Given a flying vehicle is in "DRIVE" mode with a current speed of
<initialSpeed>
  When the pilot accelerates by <accelerationAmount>
  Then the current speed should be <expectedSpeed>

# Scenario: Climb in Fly mode
# Description: Verifies that the vehicle can climb when in Fly mode.
Scenario Outline: Climb while in Fly mode
  Given a flying vehicle is in "FLY" mode with a current altitude
of <initialAltitude>
  When the pilot climbs by <climbAmount>
  Then the current altitude should be <expectedAltitude>
```

- **Changement de mode**

- *User Story:* En tant que pilote, je veux pouvoir passer d'un mode à l'autre pour m'adapter aux différents terrains.
- *Exemple:* Si le véhicule est en mode « Drive » et que le pilote change de mode, il doit passer en mode « Fly ».

```
# Scenario: Switching from Drive to Fly mode
# Description: Verifies that the vehicle correctly switches from
Drive to Fly mode.

Scenario: Switch from Drive mode to Fly mode
  Given a flying vehicle is in "DRIVE" mode
  When the pilot switches the mode
  Then the flying vehicle should be in "FLY" mode

# Scenario: Switching from Fly to Drive mode
```

```

# Description: Verifies that the vehicle correctly switches from Fly
to Drive mode.
Scenario: Switch from Fly mode to Drive mode
  Given a flying vehicle is in "FLY" mode
  When the pilot switches the mode
  Then the flying vehicle should be in "DRIVE" mode

# Scenario: Cannot switch from Fly to Drive mode while flying
# Description: Verifies that the vehicle does not switch from Fly to
Drive mode while flying.
Scenario: Switch from Fly mode to Drive mode while flying
  Given a flying vehicle is in "FLY" mode
  And the vehicle is currently flying
  When the pilot switches the mode
  Then the flying vehicle should be in "FLY" mode
  And an exception should be thrown

```

- ✓ ✓ Manage Vehicle Acceleration and Climbing
  - ✓ Accelerate the vehicle in Drive mode #1
  - ✓ Accelerate the vehicle in Drive mode #2
  - ✓ Accelerate the vehicle in Drive mode #3
  - ✓ Climb while in Fly mode #1
  - ✓ Climb while in Fly mode #2
  - ✓ Climb while in Fly mode #3
- ✓ ✓ Switch Between Drive and Fly Modes
  - ✓ Switch from Drive mode to Fly mode
  - ✓ Switch from Fly mode to Drive mode
  - ✓ Switch from Fly mode to Drive mode while flying

## Tests unitaires

- **Accélération**
  - Vérifie que la vitesse augmente correctement en mode « Drive ».
  - *Exemple:* Si la vitesse initiale est de 50 km/h et que le pilote accélère de 20 km/h, la vitesse finale doit être de 70 km/h.
- **Montée et descente**
  - Vérifie que le véhicule peut monter et descendre en mode « Fly » avec des restrictions de carburant.
  - *Exemple:* Si le véhicule monte de 100 m mais n'a pas assez de carburant, une exception doit être levée.
- **Changement de mode**
  - Vérifie que le mode actuel du véhicule change correctement.
  - *Exemple:* Si le mode est « Fly », le changement doit le basculer en mode « Drive ».

## Couverture des tests

Nous pouvons voir que nous avons une très bonne couverture du code grâce à nos tests !

Element ^	Class, %	Method, %	Line, %
org.dauphine.agile	100% (11/11)	96% (57/59)	94% (120/127)
mode	100% (6/6)	100% (28/28)	96% (64/66)
DriveModeStrategy	100% (2/2)	100% (9/9)	94% (16/17)
FlyModeStrategy	100% (2/2)	100% (13/13)	96% (24/25)
Mode	100% (1/1)	100% (2/2)	100% (2/2)
ModeManager	100% (1/1)	100% (4/4)	100% (22/22)
ModeStrategy	100% (0/0)	100% (0/0)	100% (0/0)
AltitudeManager	100% (1/1)	100% (4/4)	100% (8/8)
FlyingVehicle	100% (1/1)	86% (13/15)	82% (23/28)
FuelManager	100% (1/1)	100% (4/4)	100% (9/9)
Pilot	100% (1/1)	100% (4/4)	100% (8/8)
SpeedManager	100% (1/1)	100% (4/4)	100% (8/8)

## Conclusion : un vol sans accroc

Cette fusion entre la voiture et l'avion est bien plus qu'une prouesse technique. Elle représente l'harmonie entre deux mondes, réunie grâce aux principes de l'agilité et des design patterns.

« Voler ou rouler ? Pourquoi choisir quand on peut avoir les deux ? » conclut la voiture volante, fière de ses nouvelles capacités.

