Ansvarsfördelning:

Vehicle: Lagra data om fordon

Volvo240/Saab95/Scania/Transport : bygga en av bilarna samt skilja den från de andra fordonen

Trailer: Beskriv en trailer som kan användas av vissa fordon

Repairshop: Skapar en repairshop som kan lagra bilar. kan parametriseras för att endast ta vissa sorter

Movable: interface som ser till att allt som ska kunna röra på sig har rätt metoder för att göra det

CarController: pratar med Carview för att styra bilarna när knappar trycks samr ser till att dom rör på sig

CarView: Skapa knappar och bestämma vad dom gör

DrawPanel: Ritar grafik

Nödvändiga Beroenden:

Volvo240, Saab95, Scania och Transport är subklasser av Vehicle

Repairshop använder <T extends Car>

För att knapparna ska funka så måste CarController och CarView interagera

Annat

CarController skapar en repairshop vilket innebär att det alltid kommer finnas en i programmet. Detta bryter mot SIP då CarController egentligen bara ska hantera interaktion med UI samt logik

Mainfunktionen i CarController initierar alla bilar och carView leder till att CarController klassen får många Has-A beroenden och bryter därmed mot OCP

DrawPanel i CarController ritar ut alla bilar samt ändrar positionen på dom vilket är låg cohesion då den sköter både UI och logik

Man kan undvika att DrawPanel behöver lagra positionerna av bilarna genom att göra det i en separat klass. En "Model"-klass (MVC)