Train

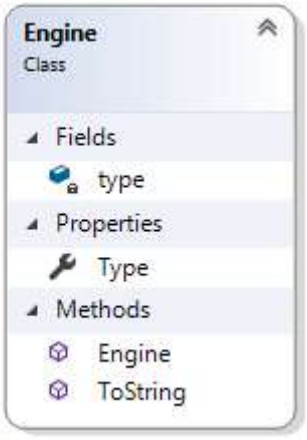
Vytvořte třídu **Locomotive**, která bude popisovat lokomotivu vlaku.

Třída má tyto datové složky a k nim odpovídající vlastnosti:

- driver : Person
- engine : Engine

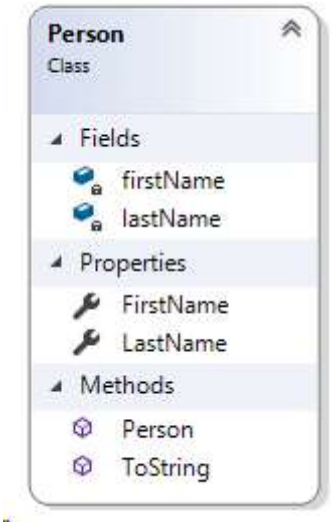
Konstruktory:
+ Locomotive ()
+ Locomotive (Person, Engine)

Třída **Engine** má datovou složku type (String) a k ní vlastnost, v pokročilém kurzu bude ze Stringu enum.

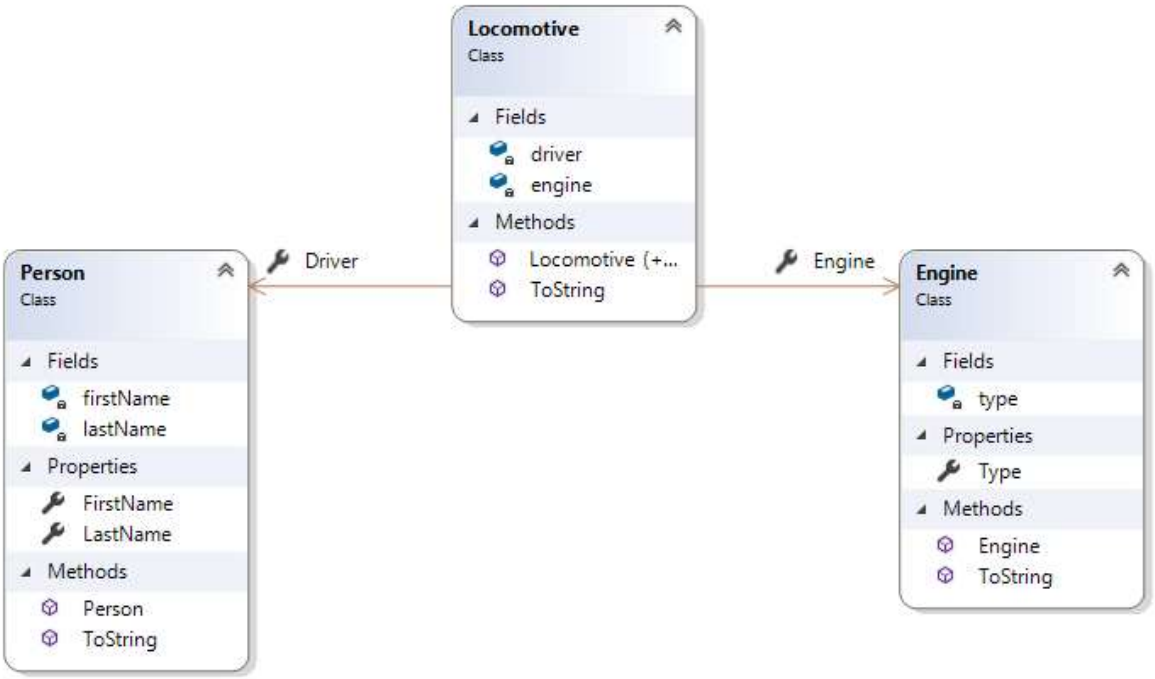


Lokomotiva může být diesel, elektrická, parní (ta je omezeného výkonu, utáhne jen 5 vagonů, pořešíme v jiné třídě).

Vytvořte třídu **Person**, která má datové složky firstName a lastname a k nim odpovídající vlastnosti. Bude mít konstruktor s oběma parametry.



Neboli řečí Class Diagramu se zobrazením provázaností:



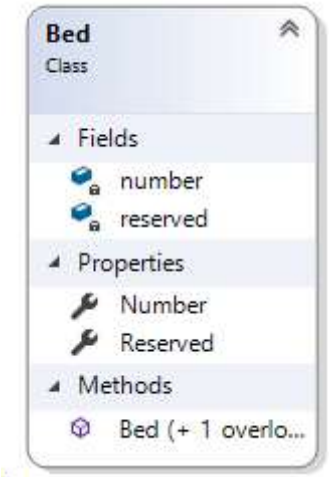
Třída **Chair** :

Datové složky a vlastnosti:
- number : int
- nearWindow : boolean

Metody:
Chair()

Vlastnost NearWindow zatím nevyužijeme.

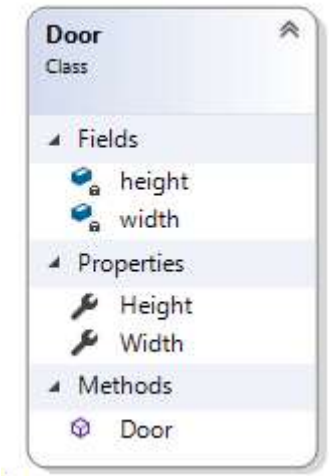
Třída **Bed**



- number: int
- reserved: boolean

Metody:
+ Bed()
+Bed(bool)

Třída **Door**:



Datové složky a vlastnosti:
- height : double
- width : double

Metody:
Door()

Třídy vagonů

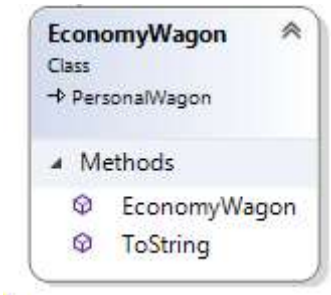
Vagony vlaku mohou být, jak už to tak bývá, vícero různých druhů. Základem pro osobní vagony je třída **PersonalWagon**, z které ale nechceme vytvářet instance, protože pro výpočet ceny jízdenky potřebujeme vědět, v jakém druhu vagonu osoba cestuje, jen označení osobní vagon nám nestačí.

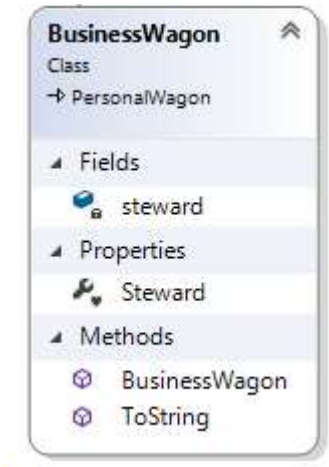
Třída PersonalWagon má tyto datové složky a k nim odpovídající vlastnosti:

- doors : List<Door>
- List<Chair> sits;
- int numberOfChairs;

Metody:
+PersonalWagon(int)
Vagony mohou mít různý počet sedadel. Datová složka numberOfChairs je i parametrem konstruktoru (k čemu ji použít, jistě víte).
(diagram tu není schválně, to už by byla moc velká nápověda)

Vagony třídy Economy a vagony třídy Business oba dědí z osobního vagonu, Business má navíc datovou složku Steward typu Person –
potřebuje v konstruktoru nastavit kromě počtu sedadel i jeho.





Spací vagón NightWagon má pár sedaček, ale hlavně lehátka – pole Beds a dvouparametrický konstruktor volající jednoparametrický konstruktor z bázové třídy.

Třída Hopper (takový ten vagón, co se do něho něco sype), která ale nemá dveře (není tedy osobní vagon).

Datová složka:
- loadingCapacity (double)

Metody:
+ Hopper(double)

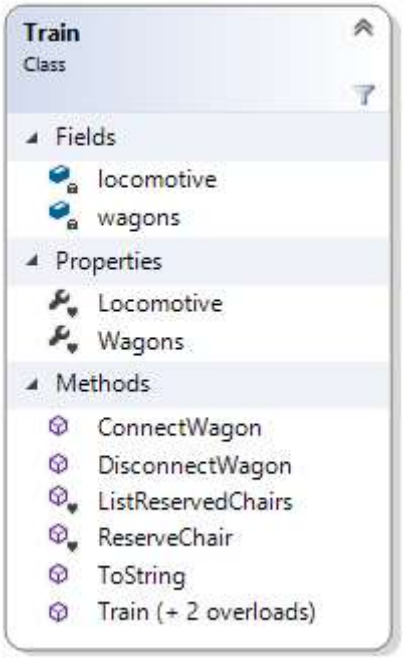
+ToString():String (platí pro všechny druhy vagonů)
Metoda vypíše informaci o tom, co je to za druh vagonu, případně počet sedadel a stewarda, pokud mají.

Všechny druhy vagonů potřebují metodu na připojení a odpojení k/od vlaku. Metody ConnectWagon a DisconnectWagon mají parametr typu Train – aby bylo jasné, ke kterému vlaku se připojují. Nejde, že by např. spací vagon po připojení nešlo od vlaku opět odpojit, takže se to musí udělat tak, aby metoda byla implementovaná povinně. Odpojovat a připojovat se samozřejmě mohou i vagony, které nemají dveře a sedadla (tedy nemohou dědit z PersonalWagon), např. vagon vozící klády, vagon na sypký materiál, do budoucna počítáme s rozšířením. Co teď?

Metoda pro připojení vagonu si také hlídá, aby parní vlak neměl víc než 5 vagonů.
Vzhledem k tomu, že všechny osobní vagonové třídy dědí z jedné, bude stačit, když bude metody mít bázová třída. Psát metodu do všech tříd by bylo neefektivní, navíc co kdybychom museli někdy do metody sahat a upravovat ji.

Ošetření chyb v odpojovací metodě: odpojení vagonu od špatného vlaku. To se v praxi stát nemůže, ale v programu ano. Před odpojením vagonu zjistíme, jestli je vůz opravdu připojený k onomu vlaku. Zkuste se vyhnout výjimkám.

Třída Train



Datové složky a vlastnosti:
- locomotive: Locomotive
- wagons: List vagonů, ve vlaku mohou být pomíchané osobní i nákladní. Běžné to není, ale musí se s tím počítat.

Metody:
+ Train()
+ Train(Locomotive)
+ Train(Locomotive, List<datový typ vagonů>)
Využijte, že máte 3 konstruktory a každý nastaví něco. Pozor, aby se vlaku vytvořeném pomocí konstruktoru s jediným parametrem daly později připojit vagony.

+ ConnectWagon (bere za parametr jakýkoliv vagon, tedy klidně i nákladní)
+ DisconnectWagon (to samé, taky bere jakýkoliv vagon)

Využijte metody pro připojení z třídy dotyčného vagonu - proč bychom měli psát jeden kód dvakrát?

+ ReserveChair(int, int):void
Metoda bere dva parametry – číslo vagonu a číslo sedadla.
Protože pole vagonů mají společný referenční typ a ne všechny vagony mají sedadla, budeme se muset podívat, jestli jsou to osobní vagóny a následně přetypovat. S kolekcí Wagons můžeme zacházet jako s polem (přistupovat k prvku pomocí indexu), metodu GetType() už známe, vypisovali jsme s její pomocí název třídy – prozkoumejte vlastnosti a metody, jestli by se pro zjištění bázové třídy nedaly použít.
Nezapomeňte zkontrolovat, jestli vagon vůbec existuje, případně jestli existuje sedadlo a taky jestli už není jednou rezervované.

+ ListReservedChairs():void
Metoda vypíše sedadla, která má někdo rezervované.

+ toString() : String
Využijte ToString() vagonů a přidejte informace o lokomotivě.

Main:
Vytvořte 3 osobní vagony (jeden z toho bude Business, stewardka Lenka Kozáková), jeden spací a jeden Hopper. Vytvořte lokomotivu (diesel se strojvedoucím Karlem Novákem). Udělejte z toho vlak, vytvořte mu další vagon (ještě jeden Hopper) a připojte.

Vytvořte parní lokomotivu s 5 vagony. Zkuste vytvořit a připojit další. Ověřte, že program protestuje.

Rezervujte sedadlo v třetím vagonu. Zkuste jinému cestovateli rezervovat sedadlo v Hopperu. Ověřte, že to nejde. Zkuste taky rezervovat už jednou rezervované sedadlo, opět ověřte, že to nejde a že program nespadne na výjimku.

Vypište rezervovaná sedadla.

Nakonec vypište informace o obou vlacích pomocí ToString().