# DAO Design Pattern

Team 9

## Overview

Data access object:

An object that provides an abstract interface to some type of database or other persistence mechanism.

## Advantages

- Encapsulation: Hide all details of how the data and the persistent model works
- Abstraction: Allow for changing of the persistent model without impacting the user of the objects
- Objects: Allow user of the object to only deal with the object itself, regardless of the persistent model

## Disadvantages

- Code duplication
- Abstraction problem when abstraction is not carefully designed
- User of the object has no control over how the data is stored and retrieved

#### TrainCompany

- name: String
- setName(String name): String
- + getName(): String

### <<interface>> TrainCompanyDAO

- + createInstance(String name):
- TrainComapny
- + getInstance (String n/1me): TrainCompany

#### TrainCompanyInMemoryDAO

- nameToTrainCompany: Map<String, TrainCompany>
- normalizeName(String name): String
- $\hbox{-} instantiate Train Company (String name):$
- TrainCompany + createInstance(String name): TrainComapny
- + getInstance (String name TrainCompany

#### TrainCompanyLocalFileDAO

- dataFilename: String
- loadData(String filename): void
- saveDate(String filename, TrainCompany trainCompany): void
- createInstance(String name, boolean shouldPersist): TrainCompany
- + createInstance(String name): TrainComapny
- + getInstance (String name): TrainCompany

## Examples

Java Object:

TrainCompany.java

DAO Interface:

TrainCompanyDAO.java

DAO with hashmap:

TrainCompanyInMemoryDAO.java

DAO with file:

TrainCompanyLocalFileDAO.java

## References

WIKI:

http://en.wikipedia.org/wiki/Data\_access\_object

### Example codes:

https://github.com/csc301-fall2014/DAOExample/tree/master/src/csc301