# Training Center – JS OOP Exam

A training center holds a database of trainers and trainings. Trainings are several types: courses, seminars and remote courses. **Trainers** have username, first name, last name and email. **Trainings** have name, description, trainer, start date and duration. **Courses** have name, description, trainer, start date and duration. **Seminars** have name, description, trainer and date. **Remote courses** have name, description, trainer, start date, duration and location.

### Input Source Code

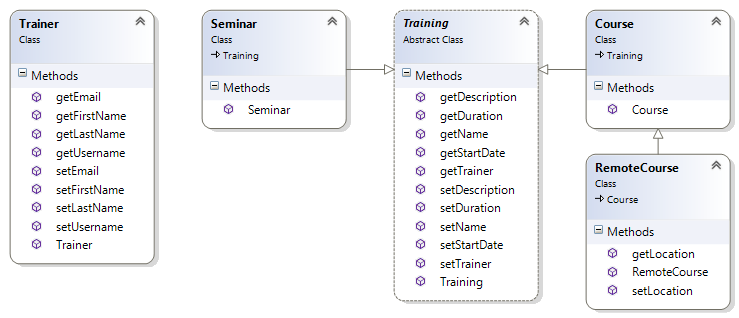
You are given a **JavaScript project (source code)** holding an **engine** (**TrainingCenterEngine** module) for working with trainers and trainings that executes the following **commands** (see the sample input and output below):

* **create Trainer {json data}** – adds a trainer by given unique username and other parameters.
* **create Course {json data}** – adds a course by given parameters.
* **create Seminar {json data}** – adds a seminar by given parameters.
* **create RemoteCourse {json data}** – adds a remote course by given parameters.
* **list** – prints all **trainers** and **trainings** from the database (in the order of their creation) in the format like in the sample output below. The engine knows how to find and print all trainers and trainings. You need to implement just the printing of each individual trainer and training.

The object properties come in standard JSON format. Unneeded properties are ignored. The engine assumes that if a command or its parameters are invalid, it should fail to execute and its output is "**Invalid command.**"

### Implement the Missing Classes

Your **first** **task** is to **implement all missing classes** to model the training center using the **best practices** for JavaScript object-oriented programming (OOP). **Avoid duplicated code** through inheritance. Encapsulate correctly all fields with getters and setters with data validation. Create the following classes:



* **Trainer**(username: string, firstName:string, lastName: string, email: string) – holds a trainer
* **Training**(name: string, description: string, trainer: Trainer, startDate: Date, duration: number) – holds a training; cannot be instantiated (abstract class)
* **Course**(name: string, description: string, trainer: Trainer, startDate: Date, duration: number) – course
* **RemoteCourse**(name: string, description: string, trainer: Trainer, startDate: Date, duration: number, location: string) – holds a remote course
* **Seminar**(name: string, description: string, trainer: Trainer, startDate: Date) – seminar; duration is always 1

### Validation Rules

* Trainer's **username** and **last name** should be non-empty strings.
* Trainer's **first name** is non-mandatory string: can be missing, but should be non-empty string when exists.
* Trainer's **email** is non-mandatory string: can be missing, but should hold '**@**' when exists.
* Training's **name** should be non-empty string.
* Training's **description** is non-mandatory string: can be missing, but should be non-empty string when exists.
* Training's **trainer** is non-mandatory: can be missing, but should be non-empty Trainer when exists.
* Training's **start date** should be non-empty valid date in range [1-Jan-2000…31-Dec-2020]. The date format is always "**day-month-year**", where **day** is in range [1..31], **month** is one of [Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec] and **year** is in range [2000…2020]. Note that some dates are invalid according to the traditional calendar system, e.g. "29-Feb-2014", "31-Nov-2015" and "38-Apr-1980".
* Training's **duration** is non-mandatory: can be missing, but should be integer in range [1…99] when exists.
* Remote course's **location** should be non-empty string.

### Implement the Existing Commands

Implement your classes so that the engine **executes correctly all above described commands**.

### Implement Additional Commands

Add additions to the engine to implement support for the following commands (look for "**TODO**" in the code):

* **delete Trainer username** – deletes an existing trainer from the database. Deleted trainers should be removed from all trainings they teach. Attempts to delete non-existing trainer throws an exception.

### Restrictions

You are **allowed** **to** **add new functions** anywhere in the code and to **modify the "TODO" sections**. You are **not allowed to modify the engine's core logic**. It will work correctly if you implement the missing classes.

### Sample Input #1

|  |
| --- |
| create Trainer {"username":"nakov", "lastName":"Nakov"}  create Trainer {"username":"maria", "firstName":"Maria", "lastName":"Green", "email":"maria@mail.ru"}  create Course {"name":"C# Basics", "startDate":"15-Nov-2014"}  create Course {"name":"JavaScript Apps", "trainer":"nakov", "startDate":"30-Nov-2014", "duration":8}  create Course {"name":"JavaScript Basics", "description":"JS course for beginners", "trainer":"maria", "startDate":"20-Jan-2015", "duration":5}  create Seminar {"name":"Python for Newbies", "date":"10-Jan-2015"}  create Seminar {"name":"MySQL for Dummies", "description":"MySQL overview for beginners", "trainer":"nakov","date":"20-Jan-2015"}  create RemoteCourse {"name":"C# Basics", "startDate":"31-Jan-2015", "location":"Varna"}  create RemoteCourse {"name":"C# Basics", "description":"Programming course for absolute beginners", "trainer":"maria", "startDate":"31-Jan-2015", "duration":9, "location":"Bourgas"}  list |

### Sample Output #1

|  |
| --- |
| Trainer created.  Trainer created.  Course created.  Course created.  Course created.  Seminar created.  Seminar created.  RemoteCourse created.  RemoteCourse created.  Trainers:  \* Trainer[username=nakov;last-name=Nakov]  \* Trainer[username=maria;first-name=Maria;last-name=Green;email=maria@mail.ru]  Trainings:  \* Course[name=C# Basics;start-date=15-Nov-2014]  \* Course[name=JavaScript Apps;trainer=Trainer[username=nakov;last-name=Nakov];start-date=30-Nov-2014;duration=8]  \* Course[name=JavaScript Basics;description=JS course for beginners;trainer=Trainer[username=maria;first-name=Maria;last-name=Green;email=maria@mail.ru];start-date=20-Jan-2015;duration=5]  \* Seminar[name=Python for Newbies;date=10-Jan-2015]  \* Seminar[name=MySQL for Dummies;description=MySQL overview for beginners;trainer=Trainer[username=nakov;last-name=Nakov];date=20-Jan-2015]  \* RemoteCourse[name=C# Basics;start-date=31-Jan-2015;location=Varna]  \* RemoteCourse[name=C# Basics;description=Programming course for absolute beginners;trainer=Trainer[username=maria;first-name=Maria;last-name=Green;email=maria@mail.ru];start-date=31-Jan-2015;duration=9;location=Bourgas] |

### Sample Input #2

|  |
| --- |
| create Trainer {"username":"nakov", "lastName":"Nakov"}  create Course {"name":"C# Basics", "startDate":"15-Nov-2014", "trainer":"nakov"}  list  delete Trainer nakov  list |

### Sample Output #2

|  |
| --- |
| Trainer created.  Course created.  Trainers:  \* Trainer[username=nakov;last-name=Nakov]  Trainings:  \* Course[name=C# Basics;trainer=Trainer[username=nakov;last-name=Nakov];start-date=15-Nov-2014]  Trainer deleted.  No trainers  Trainings:  \* Course[name=C# Basics;start-date=15-Nov-2014] |

### Sample Input #3

|  |
| --- |
| create Trainer {"lastName":"Missing username"}  create Trainer {"username":"nakov", "firstName":"", "lastName":"Empty firstName"}  create Course {"name":"Invalid startDate", "startDate":"31-Nov-2014"}  create Seminar {"name":"Missing date"}  create Seminar {"name":"", "date":"12-Nov-2014", "description":"Empty name"}  create Course {"name":"C# Basics", "startDate":"15-Nov-2014", "trainer":"invalid\_username"}  create RemoteCourse {"name":"C# Basics", "startDate":"12-Mar-2015", "description":"Missing location"}  invalid command |

### Sample Output #3

|  |
| --- |
| Invalid command.  Invalid command.  Invalid command.  Invalid command.  Invalid command.  Invalid command.  Invalid command.  Invalid command. |