CSE 1322L – Assignment 3 (Spring 2025)

Introduction

In this assignment, you will write a program to keep track of digital items in a database. Specifically, the database belongs to a company called Vapor, who plans to have a massive online store to sell software (games and development software) and digital goods (collectables and tradables). Your program will be the first step in developing this system, allowing the user to add items to the database, list them, and update their prices.

You will be required to submit a UML diagram, along with your code. Check the last pages for details.

Requirements

The features described below must be in your program:

- Item class
  1. Has 4 fields: id (integer), nextId (static integer initialized at 0), price (double), and name (String). Price must never be negative (see below).
  2. **Item(String, double)**: Initializes id with the contents of nextId, incrementing nextId by 1. It then assigns the arguments to their appropriate fields.
  3. All fields except nextId have a getter
  4. name and price both have setters
     - The setter for price must check if its argument is non-negative. If it is, set the price field using the argument. Otherwise, leave the price field with its current value.
  5. An override of toString() which returns the following string (where X is the item's name, Y is its id, and Z is its price):

     Item: X (#Y) | Price: $Z

- Software class
  1. Is a subclass of Item
  2. Has one String field, publisher, which must never be empty (see below)
  3. **Software(String, double, String)**: Calls its superclass constructor using the first two arguments. Uses the last argument to initialize its own field.
  4. The publisher field has a getter and a setter
     - The setter must check if its argument is an empty string. If it is, set the field to "Unknown". Otherwise, set it to the argument.
  5. An override of toString(), which returns the following string (where X is the item's name, Y is its id, Z is its price, and A is its publisher. Note that the

publisher is in its own line and is indented by 1 tab, compared to the first line. You can achieve this by adding a “\t” inside of your string):

```
Item: X (#Y) | Price: $Z
	Publisher: A
```

- DigitalGood class
  1. Is a subclass of Item
  2. Has one String field, description
  3. **DigitalGood(String, double, String)**: Calls its superclass constructor using the first two arguments. Uses the last argument to initialize its own field.
  4. The description field has a getter and a setter
  5. An override of toString(), which returns the following string (where X is the item’s name, Y is its id, Z is its price, and A is its description. Note the indentation):

```
Item: X (#Y) | Price: $Z
	Description: A
```

- VideoGame class
  1. Is a subclass of Software
  2. Has one boolean field, multiplayerSupport
  3. **VideoGame(String, double, String, boolean)**: Calls its superclass constructor using the first three arguments. Uses the last argument to set its own field.
  4. The multiplayerSupport field has a getter and a setter
  5. An override of toString(), which returns the following string (where X is the item’s name, Y is its id, Z is its price, A is its publisher, and B is true or false, depending on whether the game supports multiplayer. Note that the multiplayer support is in its own line and is indented by 2 tabs compared to the first line and 1 tab compared to the second line):

```
Item: X (#Y) | Price: $Z
	Publisher: A
		Multiplayer support: B
```

- DevelopmentKit class
  1. Is a subclass of Software
  2. Has one Arraylist of Strings field, targetPlatforms, which must always have at least 1 item (see below)

3. **DevelopmentKit(String, double, String, ArrayList<String>)**: Calls its superclass constructor using the first three arguments. Uses the last argument to set its own field.
4. The targetPlatforms field has a getter and a setter
   - The setter must check if its argument is an empty arraylist. If it is, add “None” to it. Regardless of whether “None” was added or not, set the object’s field with the argument.
5. An override of toString(), which returns the following string (where X is the item’s name, Y is its id, Z is its price, A is its publisher, and B is a list of the supported platforms. Note the indentation of the target platforms, as well as the fact that the list of target platforms can spawn multiple lines):

   ```
   Item: X (#Y) | Price: $Z

       Publisher: A

       Target platforms:
           B
   ```

   For example:

   ```
   Item: X (#Y) | Price: $Z

       Publisher: A

       Target platforms:
           None
   ```

   Another example:

   ```
   Item: X (#Y) | Price: $Z

       Publisher: A

       Target platforms:
           Platform 1
           Platform 2
           Platform 3
   ```

- Tradable class
  1. Is a subclass of DigitalGood
  2. Has a field saleDelay, an integer, which cannot be negative
  3. **Tradable(String, double, String, int):** Calls its superclass constructor using the first three arguments. Uses the last argument to set its own field.
  4. saleDelay has a getter and a setter

        - The setter for saleDelay must check if its argument is non-negative. If it is, set the saleDelay field using the argument. Otherwise, leave the saleDelay field with its current value.
    5. An override of toString(), which returns the following string (where X is the item's name, Y is its id, Z is its price, A is its description, and B is its sale delay. Note the indentations):

```
Item: X (#Y) | Price: $Z
        Description: A
        Item can only be sold after being owned for B days
```

- Collectable class
    1. Is a subclass of DigitalGood
    2. Has a field type, a String, which must have a specific value (see below)
    3. **Collectable(String, double, String, String):** Calls its superclass constructor using the first three arguments. Uses the last argument to set its own field.
    4. type has a getter and a setter
        - type may only have one of 4 values: "emoticon", "avatar", "background", or "nothing". If the setter is given one of the first three values above, set the field with that value. If anything else is passed as an argument, set the field to "nothing".
    5. An override of toString(), which returns the following string (where X is the item's name, Y is its id, Z is its price, A is its description, and B is what the collectable can be used for. Note the indentations):

```
Item: X (#Y) | Price: $Z
        Description: A
        Can be used as B
```

- Driver class:
- Creates an arraylist of Items called "items", which starts out empty.
- It must then implement the menu options below:
    1. **Add video game**: Prompts the user for the necessary information to create a VideoGame object, adding it to items.
    2. **Add development kit**: Prompts the user for the necessary information to create a DevelopmentKit object, adding it to items.
    3. **Add tradable**: Prompts the user for the necessary information to create a Tradable object, adding it to items.

4. **Add collectable**: Prompts the user for the necessary information to create a Collectable object, adding it to items.
5. **List all items**: Prints out all the elements of items, one per line, using their toString()
6. **Update price**: Prompts the user for an Item id. If items does not contain an item with that id, print an error message. Otherwise, prompt the user for the item's new price, and then update the item with the new price
7. **Quit**: Terminates the program

Deliverables

- Assignment3.java (driver)
- Item.java
- Software.java
- DigitalGood.java
- VideoGame.java
- DevelopmentKit.java
- Tradable.java
- Collectable.java
- UML.pdf (UML diagram)

Considerations

- This assignment may seem intimidating, but that's just because of the number of things you have to do in the driver. Don't forget that you will get partial credit for partial work.
- Feel free to add any helper methods which you believe are necessary, but know you will be not graded for them.
- Despite what the deliverables say above, you can place all of your classes in a single file
    - You will notice that placing all of your code in a single file, while still valid, will make your single file very large and potentially harder to navigate. In general, you will want to place different classes on different files.
- For variables of type double, you will **not** lose points due to rounding errors. If the sample output below says "1.00001" and yours says "1.00002", you will still get full credit **as long as your logic is correct.** Similarly, you do **not** need to round off your numbers to a specific number of decimal places. If the sample output says "2.75" and yours says "2.747155482733", you will still get full credit **as long as your logic is correct.**

Sample Output (user input in red)

```
[VaporStore Item System]

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 1

Enter name of item: Super Shooter 3000
Enter price of item: $40.00
Enter name of publisher: Ego Software
Does this game have multiplayer support? YES
Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 1

Enter name of item: Mr. Guy's Empire Builder 12
Enter price of item: $60.00
Enter name of publisher:
Does this game have multiplayer support? no
Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
```

```
Enter option: 2

Enter name of item: ARGmaker Studio
Enter price of item: $-20.00
Enter name of publisher: I Don't Get It Games
Enter the name of each supported platform, one per line. Enter an
empty line when done.
LoseOS
uOS

Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 2

Enter name of item: ComeCode IDE Suite
Enter price of item: $30.00
Enter name of publisher: ThreeValueBit inc.
Enter the name of each supported platform, one per line. Enter an
empty line when done.

Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 3

Enter name of item: Unusual Rivets
Enter price of item: $2.50
Enter item's description: Makes the RivetGun ammo in Super Shooter
3000 leave purple sparkles as it travels.
```

```
How many days before this item can be resold? -5
Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 3

Enter name of item: Flowerpot Joanne
Enter price of item: $0.05
Enter item's description: A trading card of Joanne holding a flower
pot, dropped from playing "Joanne's Greenhouse"
How many days before this item can be resold? 10
Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 4

Enter name of item: You fell for propaganda!
Enter price of item: $0.50
Enter item's description: An avatar of an orange smug-looking cat,
which can barely fit in frame.
Where can this collectable be used? avatar
Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
```

```
Enter option: 4

Enter name of item: Mathsterclass
Enter price of item: $1.75
Enter item's description: A sticker of a very smart ice fairy, giving
an excellent math lecture.
Where can this collectable be used? sticker
Item added.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 5


Here's a list of all items currently available:
Item: Super Shooter 3000 (#0) | Price: $40.00
     Publisher: Ego Software
          Multiplayer support: true

Item: Mr. Guy's Empire Builder 12 (#1) | Price: $60.00
     Publisher: Unknown
          Multiplayer support: false

Item: ARGmaker Studio (#2) | Price: $0.00
     Publisher: I Don't Get It Games
     Target platforms:
          LoseOS
          uOS

Item: ComeCode IDE Suite (#3) | Price: $30.00
     Publisher: ThreeValueBit inc.
     Target platforms:
          None

Item: Unusual Rivets (#4) | Price: $2.50
     Description: Makes the RivetGun ammo in Super Shooter 3000 leave
purple sparkles as it travels.
```

```
	Item can only be sold after being owned for 0 days.

Item: Flowerpot Joanne (#5) | Price: $0.05
	Description: A trading card of Joanne holding a flower pot,
dropped from playing "Joanne's Greenhouse"
	Item can only be sold after being owned for 10 days.

Item: You fell for propaganda! (#6) | Price: $0.50
	Description: An avatar of an orange smug-looking cat, which can
barely fit in frame.
	Can be used as avatar

Item: Mathsterclass (#7) | Price: $1.75
	Description: A sticker of a very smart ice fairy, giving an
excellent math lecture.
	Can be used as nothing


1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 6

Enter id of item: 8
No item with that id.

1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 6

Enter id of item: 6
Enter item's new price: $2.75
Price updated.
```

```
1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 5


Here's a list of all items currently available:
Item: Super Shooter 3000 (#0) | Price: $40.00
     Publisher: Ego Software
          Multiplayer support: true

Item: Mr. Guy's Empire Builder 12 (#1) | Price: $60.00
     Publisher: Unknown
          Multiplayer support: false

Item: ARGmaker Studio (#2) | Price: $0.00
     Publisher: I Don't Get It Games
     Target platforms:
          LoseOS
          uOS

Item: ComeCode IDE Suite (#3) | Price: $30.00
     Publisher: ThreeValueBit inc.
     Target platforms:
          None

Item: Unusual Rivets (#4) | Price: $2.50
     Description: Makes the RivetGun ammo in Super Shooter 3000 leave
purple sparkles as it travels.
     Item can only be sold after being owned for 0 days.

Item: Flowerpot Joanne (#5) | Price: $0.05
     Description: A trading card of Joanne holding a flower pot,
dropped from playing "Joanne's Greenhouse"
     Item can only be sold after being owned for 10 days.

Item: You fell for propaganda! (#6) | Price: $2.75
```

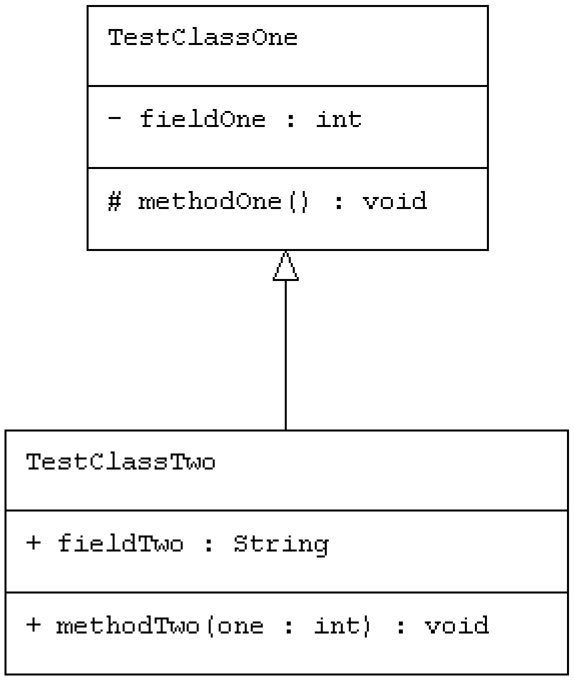
```
     Description: An avatar of a orange smug-looking cat, which can
barely fit in frame.
     Can be used as avatar

Item: Mathsterclass (#7) | Price: $1.75
     Description: A sticker of a very smart ice fairy, giving an
excellent math lecture.
     Can be used as nothing


1. Add Video Game
2. Add Development Kit
3. Add Tradable
4. Add Collectable
5. List all items
6. Update price
7. Quit
Enter option: 7

Shutting off...
```

UML Diagram

Your UML diagram must be submitted in its own file, preferably as a PDF; image files are also acceptable. Note that if your grader is unable to open the file or if they are unable to verify that your diagram is correct due to poor formatting, it will not be graded.

The picture below shows what a class should look like in your diagram. Specifically:

- The box representing the class must be divided into 3 sections: name, fields, and behaviors
- The name must match the class name exactly
- The class members (fields and behaviors) must be preceded by the appropriate accessibility symbol (+ for public, - for private, and # for protected)
- A field must be listed as "name : type" (note the colon)
- A behavior must be listed as "name(name : type) : return_type"
- If two classes have a parent-child relationship, there must be an arrow pointing from the child to the parent



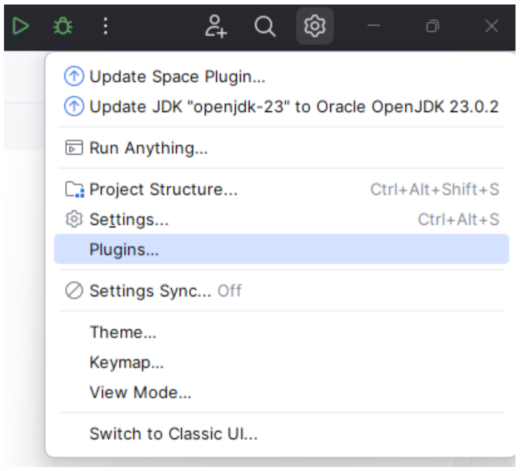
There are no restrictions on which program you use to generate the diagram. In fact, you could even hand-draw your diagram and upload a picture of it along with your code. **<u>You are strongly discouraged from doing this.</u>** Drawing your diagram manually, either by hand or by using any program besides the one listed below increases the chances of you missing something, which will lose you points. It's best to just write your code, and then have a program generate the diagram for you based on your code.

The next pages explains how to install and use the recommended UML diagram generator.

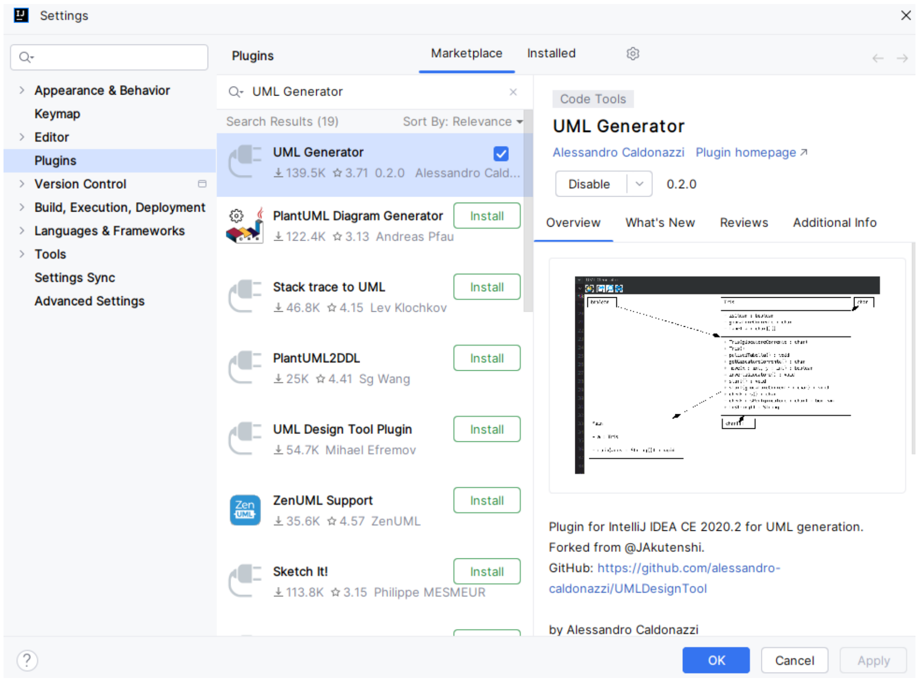
UML Generator

The recommended UML diagram generator is called "UML Generator", developed by Alessandro Caldonazzi. Instructions on installing and using it can be found below:

- With IntelliJ open, open up your list of plugins by clicking the gear icon at the top right, followed by "Plugins"
  - If you haven't updated IntelliJ in a while, the gear icon might be an arrow pointing upwards



- While in the Plugins menu on the left, select "Marketplace" at the top
- Search for "UML Generator", developed by Alessandro Caldonazzi

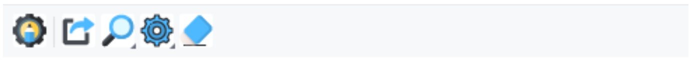


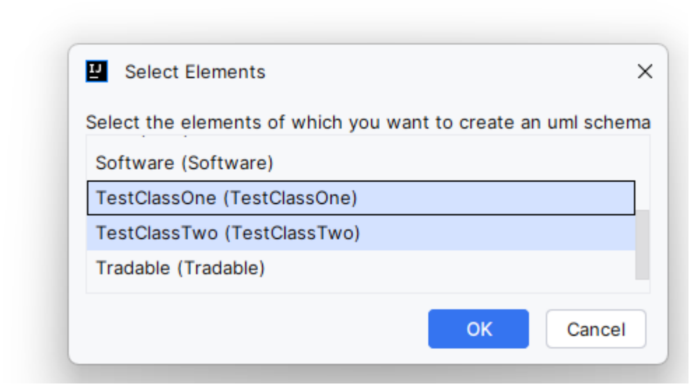
Plugin for IntelliJ IDEA CE 2020.2 for UML generation.
Forked from @JAkutenshi.
GitHub: https://github.com/alessandro-caldonazzi/UMLDesignTool

by Alessandro Caldonazzi

- Click on install. Once it's done, you may be asked to restart IntelliJ. Please do so.

- On the far right, you will now see a UML button. Clicking on it will show you a blank window with some buttons
- To generate your diagram, click the gear icon with the pencil in the middle (the first button). Then, select all the classes that are part of your diagram using the control key. Finally, click ok.



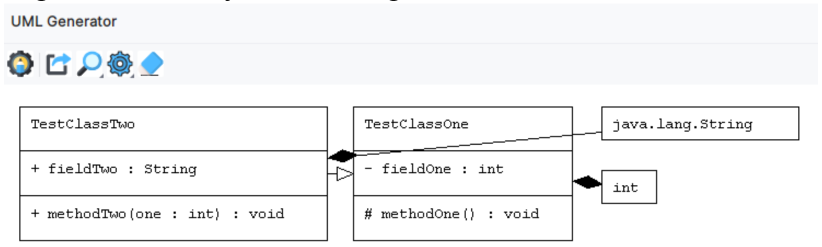


Select Elements

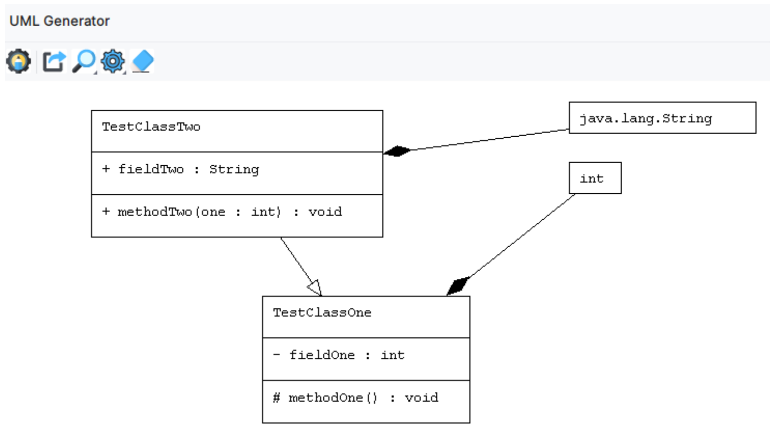
Select the elements of which you want to create an uml schema

Software (Software)
TestClassOne (TestClassOne)
TestClassTwo (TestClassTwo)
Tradable (Tradable)

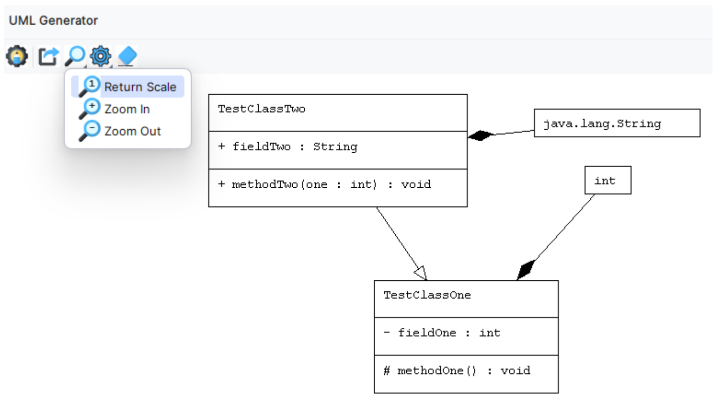
- Your initial diagram will likely look disorganized.



- You can drag its entities with your mouse, to make your diagram more legible



- You can zoom in and out using the magnifier glass button. You can pan around the image using the scroll bars at the right and bottom



- Once your diagram is presentable, click save (the second button). Select a place to save, and give your diagram a name. **<u>Do not use the name provided in the name box, as that will make your diagram fail to save</u>**.
- <u>Don't forget to submit your diagram along with your code to Gradescope.</u>