PROO/SEW III - Assignment: *News Agency* With *Observers*

Objective

Create a program for distributing news articles among different *observers*.

Things To Learn

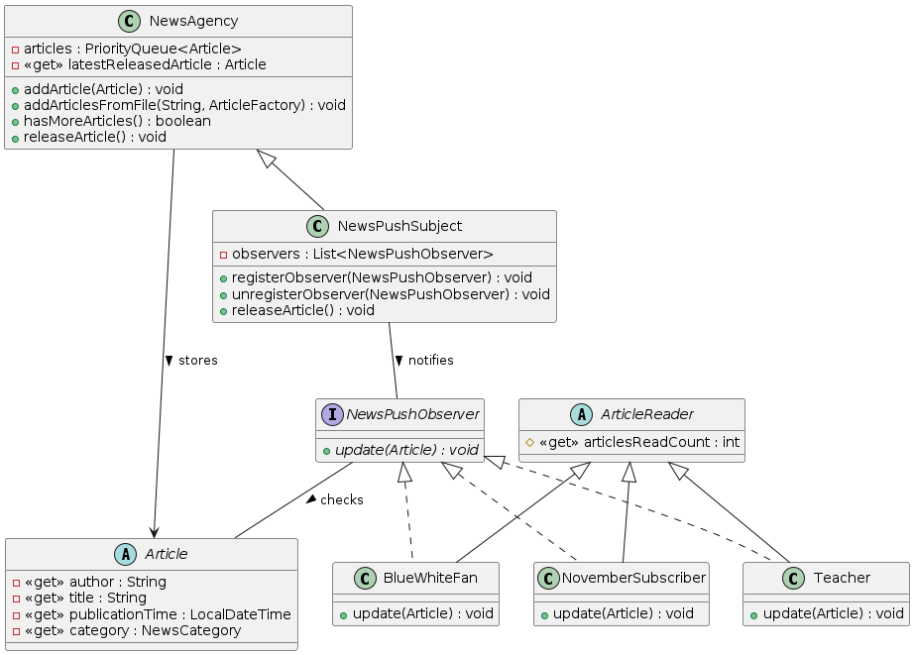
- Using *logging libraries*.
- Implementing the *observer pattern*.
- Understanding the difference between a *push* and *pull*-model.

Submission Guidelines

- Your implemented solution as ***zipped*** *IntelliJ*-project.

Task

News Mail Order (Push)



A news mail order is a good example for demonstrating the workings of the *Observer pattern*: A news mail order (the *subject*) has multiple subscribers (the *observers*), who get the printed newspaper delivered (the *update*-method and its

parameter). The readers then decide, whether the newspaper (or in our case the articles) are worth reading.

The `NewsPushSubject` should inherit from the already implemented `NewsAgency` and extend it by providing methods for adding and removing observers - you can simply store them in a list.

Override the `releaseArticle`-method by calling all the observer's `update`-methods using the latest published article. Don't forget to call the parent's `releaseArticle`-method before that.

Next, implement the observers: All three should extend the `ArticleReader` class, that provides simple functionality (i.e. a *protected* `int`) for storing the number of read articles. Additionally, the `NewsPushObserver`-interface needs to be implemented. When the `update`-method is called, the observer decides, whether the supplied article is worth reading (see below) and increments the count of read articles accordingly.

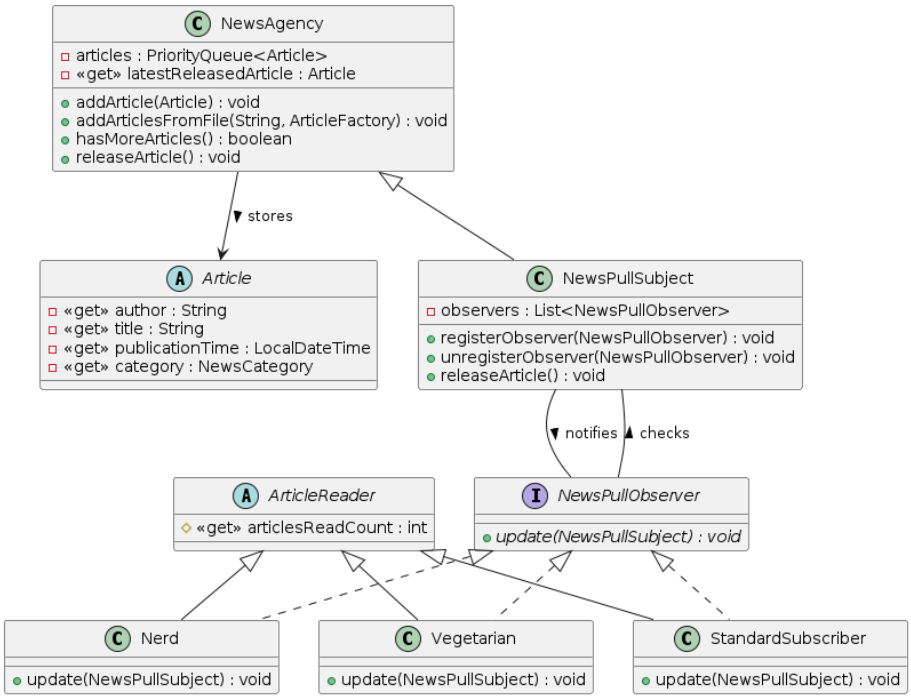
The observers have the following interests:

- The `BlueWhiteFan` only reads articles containing the words `Blau` or `Weiss` in the title (while ignoring the *case*).
- The `NovemberSubscriber` only reads articles that came out in the month of November.
- The `Teacher` only reads articles from the *culture* and *politics* categories.

Online News Portal (Pull)

You can imagine the *Pull*-approach of the *Observer pattern* like an online news website that just informs (*notifies*) their readers, that a new article has been published - the readers then need to check the website themselves.

The implementation is nearly identical to the one above - the only difference is the parameter of the `update`-method. Instead of passing the article, we pass the portal publishing it (i.e. `this`). The observers can then access the latest published article easily, since `NewsPullSubject` extends `NewsAgency`.



The observers of this task have the following interests:

- The `Nerd` only reads articles from the *science* category.
- The `StandardSubscriber` only reads articles written by *Der Standard*.
- The `Vegetarian` only notices articles containing `Fleisch` (case-insensitive) in the title, so they can leave angry comments.

Don't forget to use *logging* when it makes sense.