Programming 2

Christian Grévisse (christian.grevisse@uni.lu)

Lab 10 – (Im)mutability

2018–05–03

Exercise 31 – To-do List

Write a Swift console application that manages to-do lists. A to-do list has a name and contains an array of items. Each time an item is added to the list, the whole list gets printed.

A to-do item has a title and description, that cannot be modified anymore after creation. Also, an item has a flag whether it has been done or not.

It shall be possible to share the items of one list to another. However, as with two different paper-based to-do lists, changes on the done flag of an item on one list shall not be reflected on the other list.

Use (im)mutability and access control wisely. Test your implementation. A possible output could look like this:

```
Chores:
* Mow the lawn: Check whether the lawn mower has any gasoline left!

Chores:
* Mow the lawn: Check whether the lawn mower has any gasoline left!
* Wash the car: Check windshield washer fluid!

Saturday Afternoon:
* Watch El Clasico: Buy chips & beer!

Share items from 'Chores' on 'Saturday Afternoon' ...
Saturday Afternoon:
* Watch El Clasico: Buy chips & beer!
* Mow the lawn: Check whether the lawn mower has any gasoline left!
* Wash the car: Check windshield washer fluid!

Check 'Wash the car' on 'Saturday Afternoon' ...
Saturday Afternoon:
* Watch El Clasico: Buy chips & beer!
* Mow the lawn: Check whether the lawn mower has any gasoline left!
- Wash the car: Check windshield washer fluid!

Chores:
* Mow the lawn: Check whether the lawn mower has any gasoline left!
* Wash the car: Check windshield washer fluid!
```

Exercise 32 – Emails - IMAP

Write a Swift console application that simulates a mail server and mail clients.

An email has a subject, a message, a date (use the `Date` type from the `Foundation` library). These properties cannot be changed after initialization. In addition, an email can have an optional flag, either *unread*, *important* or *spam*. For this, declare and use an enum:

```swift
enum EmailFlag { case Unread, Important, Spam }

var flag:EmailFlag? = .Unread
```

A mail server holds an array of emails. A mail client has a reference to a server and also holds an array of emails. In this exercise, we model the IMAP protocol, meaning that a change on an email (e.g. its flag) in a mail client will be reflected on the mail server, too. When synchronizing a client, its collection of emails will be the same as the one on the server. The emails on the server will not be deleted.

Test your implementation. Especially, put attention that the change of a flag on an email in one mail client will be reflected in another client and on the server.

A possible output could look like this:

```
Content mail client 1:
[Unread] Culture Flash (Sent: 2018-05-02 18:48:04 +0000)
------------------------------------
Excite your senses!

Marking as spam ...
Content mail client 2:
[Spam] Culture Flash (Sent: 2018-05-02 18:48:04 +0000)
------------------------------------
Excite your senses!

[Unread] Call for Papers (Sent: 2018-05-02 18:48:04 +0000)
------------------------------------
Submit your original work now to the 1st Conference on Dynamic Memory Allocation!

Read email ...
Content mail client 2:
[Spam] Culture Flash (Sent: 2018-05-02 18:48:04 +0000)
------------------------------------
Excite your senses!

Call for Papers (Sent: 2018-05-02 18:48:04 +0000)
------------------------------------
Submit your original work now to the 1st Conference on Dynamic Memory Allocation!

Content mail client 1:
[Spam] Culture Flash (Sent: 2018-05-02 18:48:04 +0000)
------------------------------------
Excite your senses!

Call for Papers (Sent: 2018-05-02 18:48:04 +0000)
------------------------------------
Submit your original work now to the 1st Conference on Dynamic Memory Allocation!
```

Exercise 33 – Emails - POP3

Take the exercise from before, but this time, model the POP3 protocol, i.e. changes on emails in a mail client will not be reflected on the server or in other mail clients.

When downloading emails, you will probably need a way to distinguish between emails you already downloaded. Add an ID to the email type.

Again, test your implementation. Especially, put attention that the change of a flag on an email in one mail client will *not* be reflected in another client or on the server.