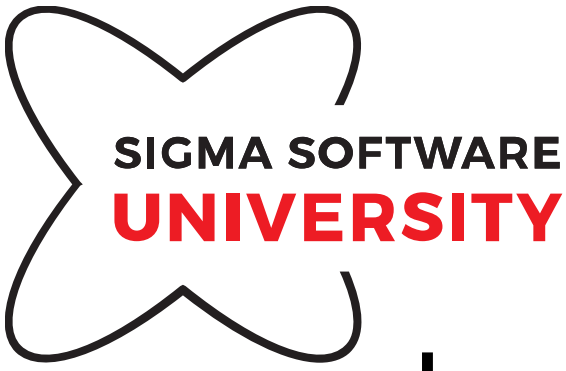


homework

SIGMA SOFTWARE UNIVERSITY

ЛЕКЦІЯ №4

Підготовчі завдання до лекції на тему

“Управління памяттю в Swift”

Дата проведення: __.__.201_
Лектор: Горбушко Кирил

СПИСОКЛІТЕРАТУРИ

Ознайомтеся зі списком літератури наведеним нижче. Описані джерела надать необхідну базову інформацію для засвоїння матеріалу лекції та виконання практичного завдання.

1. Objective-C Memory Management Essentials By Gibson Tang, Maxim Vasilkov (978-1-84969-712-5)..
2. The Swift Programming Language, Apple Inc,
   - chapter Automatic Reference Counting.
3. Weak vs Unowned (krakendev.io/blog/weak-and-unowned-references-in-swift).
4. Weak vs unowned (stackoverflow.com/questions/24320347/shall-we-always-use-unowned-self-inside-closure-in-swift).
5. Мэтт Гэлловей - Сила Objective-C 2.0 - 2014. (Chapter 5)..
6. А. Махер - "Программирование для iPhone. Высшый уровень" (розділ 1.3).
7. Memory management - (https://developer.apple.com/library/ios/documentation/Cocoa/Conceptual/MemoryMgmt/Articles/MemoryMgmt.html#//apple_ref/doc/uid/10000011-SW1).
8. Object Life-time https://developer.apple.com/library/archive/documentation/General/Conceptual/DevPedia-CocoaCore/ObjectLifeCycle.html
9. Stack vs Heap - https://www.gribblelab.org/CBootCamp/7_Memory_Stack_vs_Heap.html
10. About memory - https://swiftrocks.com/memory-management-and-performance-of-value-types.html

САМОКОНТРОЛЬ

Ознайомтеся зі списком ключових слів, що характеризують матеріал лекції. Володіння усіма описаними термінами є розумінням матеріалу лекції.

1. MMR.
2. ARC.
3. retain.
4. release.
5. retain cycle.
6. unowned.
7. capture list.
8. allocation/deallocation.
9. weak/strong.
10. zombie object.
11. autorelease pool
12. retain counter
13. memory leak
14. NSAutoreleasePool
15. dealloc/deinit
16. copy object and related processes
17. garbage collector

ПРАКТИЧНЕЗАВДАННЯ

Виконайте завдання наведені нижче. Кожне завдання складено у межах матеріалу лекції та не потребує додаткових знань. Дивіться технічні вимоги до виконання практичного завдання у відповідному розділі.

ЗАВДАННЯ №1

ОПИС:

Робота з типами.

ДЕТАЛІ:

Реалізуйте пункити описані нижче.

ПУНКТИ ВИКОНАННЯ:

1. Опишіть тип який може зберігати інформацію про стан дверей (нова чи ні, колір, тип покриття і тп)
2. Уявіть що пройшов певний час і вам на основі попереднього стану треба отримати поточний стан дверей, опишіть цей стан
3. 2 стани мають існувати одночасно
4. Який тип (клас чи структура) краще використати задля вирішення цього завдання? Чому?

5. Створити сутність *Person* і описати в ньому властивість *name*.
6. Створити ще одну сутність *Person* і описати в ньому властивість *name*
7. Створити змінні яким присвоїти створені вище сутності
8. Модифікувати ім’я змінних
9. Порівняти ім’я першої сутності з іменем другої
10. Який тип (клас чи структура) краще використати задля вирішення цього завдання? Чому?

ЗАВДАННЯ №2

ОПИС:

Робота з пам’яттю.

ДЕТАЛІ:

Реалізуйте пункити описані нижче.

ПУНКТИ ВИКОНАННЯ:

1. Створити клас *Person* і описати в ньому властивість *name*.
2. В ініціалізаторі присвоїти імя і вивести в лог
3. Описати deinit і вивести в лог інформацію
4. Створити об’єкт типу *Son*
5. Запусти playground - і звернути увагу на те, що лог з *deinit* ніколи не викликається
6. Створити *skope (наприклад do {})* для створеного обєкту і перезапустити playground. Звернути увагу на логи
7. Створити об’єкт *Son* (з логами в *init/dealloc* як і в *Person*) з властивістю *parent* і додати властивість *child* в *Person*
8. *З*берегти посилання *Person* в *Son*, *Son* в *Person*
9. Запустити код - звурнути увагу на логи (deinit не викликається)
10. Модифікувати код так, щоб deinit з кожного об’єкту викликався

ЗАВДАННЯ №3

ОПИС:

Практика з reference cycles in closures.

ДЕТАЛІ:

Реалізуйте пункити описані нижче.

ПУНКТИ ВИКОНАННЯ:

Частина 1

1. Створити обєкт *Operand* з властивістю *number*
2. Додати в обєкт властивість типу *operand: Operand?*
3. Створити closure для обчислення суми 2-х чисел з кількох *Operand* обєктів
4. Написати код який створить retainCycle
5. Написати код який вирішує проблему з попереднього пункту

Частина 2

1. Додати в клас *Operand* closure як властивість, яка робить якусь операцію (без параметрів). Виконати цей блок і *init.*
2. Створити об`єкт типу Operand і відразу занілити його
3. Перевірити логи (deinit не викликається)
4. Модифікувати код з використанням weak / unowned в capture list для вирішення даної проблеми

ЗАВДАННЯ №4 (Optional)*

ОПИС:

Робота з MMR. Objective-C

ДЕТАЛІ:

Реалізуйте пункити нижче.

ПУНКТИ ВИКОНАННЯ:

1. Створіть проект Objc-C для консольного додатку macOSx
2. При створенні класів використати -fno-objc-arc флаг для файлів (https://stackoverflow.com/questions/6646052/how-can-i-disable-arc-for-a-single-file-in-a-project)
3. Створити клас My*Number* з властивістю *value* використавши принципи MMR
4. В main функції створити обєкти типу My*Number*
5. *Описати виконання найпростіших математичних операцій і вивід цієї інформації в консоль (декілька на ваш вибір)*
6. Присвоїти значення value
7. Вивести значення в консоль
8. Забезпечити звільнення пям'яті від об'єктів
9. Додати альтернативний спосіб з використанням autorelease pool, autorelease

ЗАВДАННЯ №5

ОПИС:

Практика з Debug Memory Graph.

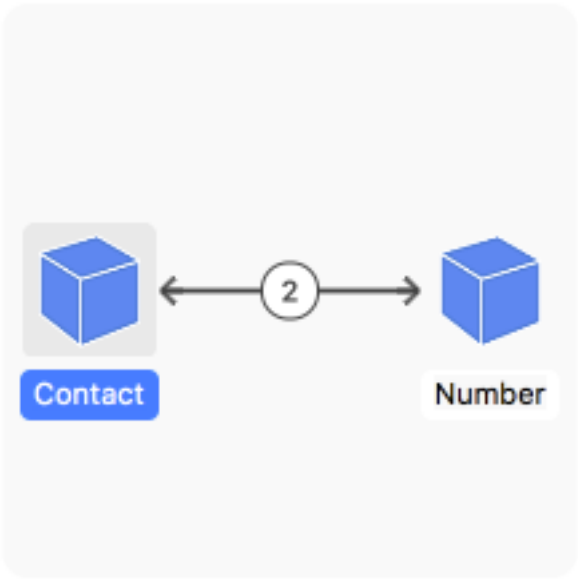
ДЕТАЛІ:

Реалізуйте пункити описані нижче.

ПУНКТИ ВИКОНАННЯ:

1. Завантажте код з посилання (Download the starter project) (Використано https://www.raywenderlich.com/)
2. Запустіть проект і видаліть кілька рядків (використайте swipe)
3. Натисніть Debug Memory Graph

4. Знайдіть в Debug Memory Graph знак оклику (фіолетовий)
5. Виберіть підсвічений пункт



6. Згідно (https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/MemoryMgmt/Articles/mmPractical.html) - parent object should have a strong hold on a child object by convention — not the other way around.
7. Можифікуйте проект так, щоб проблема була вирішена
8. Запустіть проект і дослідіть поведікну лічильників посилань. Проблема має зникнути.

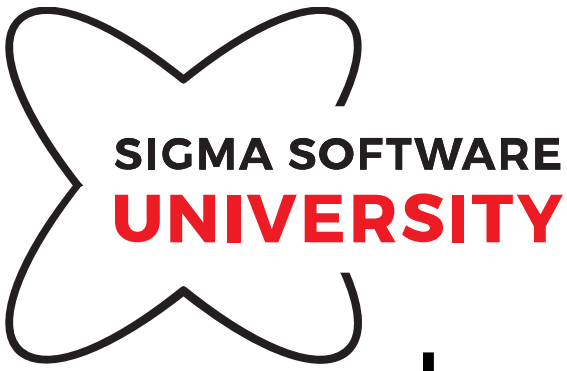
ТЕХНІЧНІВИМОГИ

Додаткові вимоги для успішного виконання тестового завдання
Операційна система: OS X Sierra or higher
Середовище розробки: Xcode 8.X or higher
Платформа: Playground for Swift, Command line template for Objective-C
Мова програмування: Swift / Objective-C
Місце здачі: нова гілка репозиторію, виділеного технічним відділом

ЗВОРОТНІЙ ЗВ’ЯЗОК

У разі виникнення будь-яких питань стосовно матеріалу лекції можна звернутися до

Лектор: Горбушко Кирил
E-mail: kyryl.gorbushko@sigma.software
Skype: kirill.g3



homework