Прв колоквиум по Напреден Веб Дизајн (група 1)

време за работа: 90 мин.

4 ноември 2017 г.

Упатство за работа: Најавете се на компјутерот со Вашето корисничко име и лозинка. Испитот се состои од практичен и теоретски дел. Задачите од практичниот дел се поставени на порталот `http://courses.finki.ukim.mk/` , додека теоретскиот дел се полага на `http://etest.finki.ukim.mk/`.

Пред завршување, потребно е да креирате фолдер именуван во следниов формат: `index_ime_prezime` (пр. `11111_Mitko_Mitkovski`). Во тој фолдер треба да Ви бидат решенијата на задачите. Фолдерот на крај треба да го зипувате (користете `7Zip` за да го компресирате фолдерот) и да го прикачите на соодветното место на курсот.

1 Задача (15 поени)

`Singleton` претставува објект од кој може да има максимум една инстанца (или да нема ниту една инстанца). За да се постигне тоа, потребно е да се креира посебна метода/функција (пр. `getInstance`), која **ќе го креира објектот (доколку претходно не постоел) или го враќа моментално креираниот објект доколку претходно бил креиран**. Притоа, програмата не смее да има пристап до деловите од Singleton-от кои се одговорни за неговото инстанцирање (односно треба да има пристап само до методот `getInstance` (тука се подразбира дека треба да има пристап до останатите стандардни методи кои доаѓаат предефинирано).

Ваша задача е да напишете JavaScript код кој ќе овозможи да се креира Singleton објект кој во себе ќе го содржи времето кога објектот е креиран (односно вредноста `Date.now()`). Притоа доколку некој проба да креира нов објект, треба да се врати веќе креираниот.

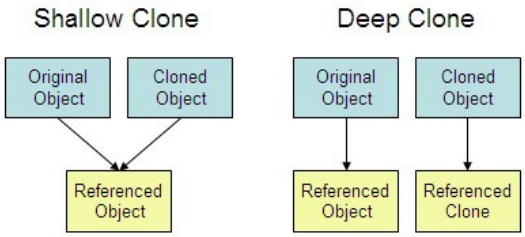
Во `zad1/zad1.js` е даден код кој треба да се проработи според претходно наведеното. Пример излез Ви е даден во `example_output.txt`.

2 Задача (20 поени)

Кога се прави дупликат на објект во JavaScript, се прави т.н. плитка копија (`shallow clone`). Доколку објектот содржи примитивни типови, тогаш тие се дуплицираат и поставуваат во копијата, инаку ако содржат објекти, тие се референцираат.

Да се креира функција `deep_clone(orig_ob)`, која прима објект кој треба да се копира (`orig_obj`) и да врати објект кој треба да биде длабока копија. Почетен код Ви е даден во `zad2/zad2.js`.

ВНИМАНИЕ: *Доколку има повеќе нивоа на објекти, треба да се копираат сите. Исто така доколку има наследување, треба да се обрне внимание на објектите кои се наследуваат. Да не се користат готови функции за копирање кои се присутни во различните JavaScript библиотеки.*



Фигура 1: Shallow vs deep clone

3 Задача (35 поени)

Да се развие објект наречен `NPC`. За секој `NPC` се чува име и неговото здравје (`hitpoints`). Името на секој NPC објект претставува број, кој го претставува редниот број на инстанцираниот `NPC`.

Да се развие објект наречен `Hero`, кој го проширува објектот `NPC`. За секој `Hero` бјект потребно е да се чува име на карактерот (како `String`), неговото здравје (`hitpoints`, предефинирано од 0-100) и колку штета прави неговиот напад (`damage`). При креирање на нов `Hero`, името задолжително се задава како параметар, додека додека не е дополнително наведено, секој карактер има `100 hitpoints` и прави штета од `10` поени

За објектите `Hero` потребно е да се креира метода `attack`, која напаѓа друг објект од типот `NPC` или `Hero`. Дополнително за секој карактер да се креира метода `status` која го печати името на карактерот и неговите моментални hitpoints.

Во поле треба да крерирате (произволно) `N` објекти од типот `Hero` и `NPC` (каде `N` е произволен број помеѓу `10` и `50`. Во еден круг, еден `Hero` објект може да нанесе 1 удар на произволен противник. Еден карактер (`NPC` или `Hero`) умира ако неговиот `hitpoint` е 0. Играта завршува кога останува само 1 жив `Hero`, кој се печати на конзола со порака `Winner: name, hitpoints!`. Првенствено, **еден** од хероите има `critical` својство кое му овозможува да нападне со `150%` од штетата која може да ја направи (пр. ако прави штета од 10, со ова својство ќе прави штета од 15). Доколку тоа својство го искористи врз `NPC` објект, го задржува истото во следниот круг, додека ако нападне друг `Hero` објект, и нападнатиот `Hero` објект преживее, му го предава на нападнатиот херој, за тој да може да го искористи во следниот круг. Доколку со `critical` нападот го убие `Hero` објектот, тогаш својството се губи и не се користи од ниту еден `Hero` објект понатаму.