Статичні члени класу

Статичні поля. Опис

Статичні поля :

- це поля, які належать класу, а не окремому об’єкту
- існують в одному екземплярі
- описуються з використанням службового слова ***static***
- для звертання до них необхідно використовувати назву класу

Загальна форма	Приклад

```
============== опис статичних полів =========
class Назва класу  {

  //---Опис полів класу (належать класу а не об’єктам)---
  //---(існують тільки в одному екземплярі) ---

  static назва_статичного_поля_1 = поч. значення1
  static назва_статичного_поля_2 = поч. значення2
    . . . . . .

 constructor(форм.параметри){
   //--- Опис полів (індивідуальні дані об’єктів)---
  //---(у кожного об’єкта свої окремі екземпляри)----
   this.властивість1 = значення1
   this.властивість2 = значення2
    . . .
 }
  . . . . . .
}
```

```
class Person {


  //---Опис полів класу (належать класу а не об’єктам)---
  //--- Максимальний вік, що вважається молодим ---
  //--- це величина для будь-якої людини ---

  static maxYoungLimit = 45

   constructor(age) {

      //---Опис полів (вік у кожного свій)---
      this.Age = age

   }

   . . . . . .

}
```

Статичні поля. Звертання до статичного поля поза межами класу

- це поля, які належать класу, а не окремому об’єкту
- існують в одному екземплярі
- описуються з використанням службового слова ***static***
- для звертання до них необхідно використовувати назву класу

Загальна форма

```
============== опис статичних полів =========
class Назва класу  {

  //---Опис полів класу (належать класу а не об’єктам)---
  //---(існують тільки в одному екземплярі) ---

  static назва_статичного_поля_1 = поч._значення1
  static назва_статичного_поля_2 = поч._значення2

   . . . . . .


}
```

```
//---- створення об’єкта ----

об’єкт = new Назва класу (… параметри …)

//---- звертання до статичного поля ----
Назва класу . назва_статичного_поля = значення

змінна = Назва класу . назва_статичного_поля
```

Приклад

```
class Person {


  //---Опис полів класу (належать класу а не об’єктам)---

   static maxYoungLimit = 35

   constructor(name, age) {
      //---Опис полів (вік у кожного свій)---
      this.Name = name
      this.Age = age
   }
   . . . . . .
}
```

```
//---- створення об’єкта ----
let p1 =  new Person(‘Ivan’, 43)


//---- звертання до статичного поля ----
Person . maxYoungLimit = 45    //Зміна значення

let s = Person . maxYoungLimit //Зчитування значення
```

Статичні поля. Звертання до статичного поля всередині нестатичних методів класу

- це поля, які належать класу, а не окремому об’єкту
- існують в одному екземплярі
- описуються з використанням службового слова ***static***
- для звертання до них необхідно використовувати назву класу

Загальна форма	Приклад
`============== опис статичних полів =========` `class` *Назва класу* { //---Опис полів класу (належать класу а не об’єктам)--- `static` *назва_статичного_поля_1* = поч._значення1 `static` *назва_статичного_поля_2* = поч._значення2	class *Person* { //---Опис полів класу (належать класу а не об’єктам)--- static maxYoungLimit = 45 constructor(age) { //---Опис полів (вік у кожного свій)--- this.Age = age }
//-- Опис методів -- функція-метод_1 (форм.парам.) { . . . *Назва класу* . *назва_статичного_поля_1* . . . } . . . . . . }	isUserYoung() { return this.Age <= *Person*.maxYoungLimit } }
//---- створення об’єкта ---- об’єкт = `new` *Назва класу* (… параметри …) //---- звертання до статичного поля ---- *Назва класу* . *назва_статичного_поля* = значення	//---- створення об’єкта ---- let p1 = new Person(‘Ivan’, 43) let p2 = new Person(‘Olga’, 36) document.write(Person . maxYoungLimit) document.write(p1.isUserYoung()) document.write(p2.isUserYoung()) //---- звертання до статичного поля ---- Person . maxYoungLimit = 45 document.write(Person . maxYoungLimit) document.write(p1.isUserYoung()) document.write(p2.isUserYoung())

Статичні методи. Звертання до статичного поля всередині нестатичних методів класу

- описуються з використанням службового слова ***static***
- для звертання ззовні або з нестатичних методів потрібно використовувати назву класу
- у цих методах this – це сам клас
- для звертання з статичних методів до інших статичних методів чи полів можна також використовувати ***this***
- ***не можуть звертатись до нестатичних полів об'єкту через this !!!***

Загальна форма	Приклад
`class` ***Назва класу*** {	`class Person {`
`//---Опис полів класу (належать класу а не об'єктам)---`	`//---Опис полів класу (належать класу а не об'єктам)---`
`static` ***назва_статичного_поля_1*** = поч._значення1 `static` ***назва_статичного_поля_2*** = поч._значення2	`static maxYoungLimit = 45` `static compareCount = 0`
. . . . .	`constructor(age) {` `//---Опис полів (вік у кожного свій)---` `this.Age = age` `}`
`//-- Опис нестатичних методів --` функція-метод_1 (форм.парам.) { . . . ***Назва класу*** . ***назва_статичного_поля_1*** . . . }	`isUserYoung() {` `return this.Age <=` ***Person***`.maxYoungLimit` `}`
`//-- Опис статичних методів --` `static` ***статична-функція-метод_1*** (форм.парам.) { . . . ***Назва класу*** . ***назва_статичного_поля_1*** Або ***this*** . ***назва_статичного_поля_1*** . . . }	`static` ***arePersonsSameYears***`(person1, person2) {` ***Person***`.compareCount ++` `//або this . compareCount ++` `return person1.Age === person2.Age` `}`
}	}
`//---- створення об'єкта ----` об'єкт = ***new*** ***Назва класу*** (… параметри …) `//---- звертання до статичного метода ----` ***Назва класу*** . ***назва_статичного_метода*** (параметри)	`//---- створення об'єкта ----` `const p1 = new Person('Ivan', 22)` `const p2 = new Person('Olga', 28)` `console.log(Person.`***arePersonsSameYears***`(p1, p2)`

Статичні методи. Звертання до статичного поля всередині нестатичних методів класу

- описуються з використанням службового слова ***static***
- для звертання ззовні або з нестатичних методів потрібно використовувати назву класу
- у цих методах this – це сам клас
- для звертання з статичних методів до інших статичних методів чи полів можна також використовувати ***this***
- ***не можуть звертатись до нестатичних полів об'єкту через this !!!***

Загальна форма	Приклад
`class` ***Назва класу*** {	`class Person {`
`//---Опис полів класу (належать класу а не об'єктам)---`	`//---Опис полів класу (належать класу а не об'єктам)---`
`static` ***назва_статичного_поля_1*** = поч._значення1 `static` ***назва_статичного_поля_2*** = поч._значення2	`static maxYoungLimit = 45` `static compareCount = 0`
. . . . .	`constructor(age) {` `//---Опис полів (вік у кожного свій)---` `this.Age = age` `}`
`//-- Опис нестатичних методів --` функція-метод_1 (форм.парам.) { . . . ***Назва класу*** . ***назва_статичного_поля_1*** . . . }	`isUserYoung() {` `return this.Age <= Person.maxYoungLimit` `}`
`//-- Опис статичних методів --` `static` ***статична-функція-метод_1*** (форм.парам.) { . . . ***Назва класу*** . ***назва_статичного_поля_1*** Або ***this*** . ***назва_статичного_поля_1*** . . . }	`static arePersonsSameYears(person1, person2) {` `Person.compareCount ++` `//або this . compareCount ++` `return person1.Age === person2.Age` `}`
}	}
`//---- створення об'єкта ----` об'єкт = ***new*** ***Назва класу*** (… параметри …) `//---- звертання до статичного метода ----` ***Назва класу*** . ***назва_статичного_метода*** (параметри)	`//---- створення об'єкта ----` `const p1 = new Person('Ivan', 22)` `const p2 = new Person('Olga', 28)` `console.log(Person.arePersonsSameYears(p1, p2)`

Задача. Реалізувати конвертер валют. Курси валют та методи перетвоення повинні бути статичними

Задача. Розробити "Рекламний агент". Випадковим чином у зададному діапазоні задається інтервал,
і через згенеровану кількість секунд виводиться деяка реклама. Після цього знову генерується випадковий інтервал.
У процесі роботи для усіх раніше створених об"єктів - рекламних агентів мінімальне/максимальне значення інтевалу можуть змінюватись.

Задача. Статистика методів.
Дано клас Масив, який зберігає масив і має методи для знаходження суми, добутку, максимального.
Користувач може створити довільну кількість об’єктів даного класу. Підрахувати загальну кількість викликів кожного із методів (незалежно від об’єкта)

Статичні приватні поля, методи (геттери/сеттери). Властивості

- приватні поля, геттери, сеттери описуються з використанням службового слова ***static***
- для звертання ззовні або з нестатичних властивостей потрібно використовувати назву класу
- у цих методах this – це сам клас
- для звертання з статичних методів до інших статичних методів чи полів можна також використовувати ***this***

Загальна форма	Приклад
class *Назва класу* { //---1) Опис приватного статичного поля --- ***static #*** *назва_статичного_поля* = поч._значення1 . . . . . . static get *метод_зчитування_геттер* () { return this.# *назва_статичного_поля* } static set *метод_запису_сеттер* (newVal) { . . . . . . this.# *назва_статичного_поля* = newVal } . . . . . . }	class *Person* { //---Опис полів класу (належать класу а не об'єктам)--- static #maxYoungLimit = 45 . . . . . . static get *maxYoungLimit*() { return this.#maxYoungLimit } static set *maxYoungLimit*(newVal) { if (newVal < 16) throw new Error('The value is incorrect') this.#maxYoungLimit = newVal }
//---- створення об'єкта ---- об'єкт = ***new*** *Назва класу* (… параметри …) //---- звертання до статичного метода ---- *Назва класу* . *назва_статичної_властивості* = значення змінна = *Назва класу* . *назва_статичної_властивості*	//---- створення об'єкта ---- const p1 = new Person('Ivan', 22) const p2 = new Person('Olga', 28) Person. *maxYoungLimit* = 35 let s= Person. *maxYoungLimit* console.log(Person. *maxYoungLimit*)

Статичні поля. Singleton

- гарантує, що клас матиме тільки один екземпляр
- забезпечує глобальну точку доступу до цього екземпляра
- через низку притаманних недоліків деякі розробники вважають його антипатерном

Загальна форма	Приклад
`============== опис статичних полів =========` `class` *Назва класу* { //---Опис полів класу (належать класу а не об'єктам)--- //---(*існують тільки в одному екземплярі*) --- `static` *назва_статичного_поля* . . . . . . `constructor(форм.параметри)`{ //---Якщо значення статичного поля визначене--- //---то повертаємо вже раніше створений об'єкт --- if(*Назва класу* . *назва_статичного_поля*) return *Назва класу* . *назва_статичного_поля* //---інаше створюємо новий об'єкт---- `this`.властивість1 = значення1 `this`.властивість2 = значення2 . . . *Назва класу* . *назва_статичного_поля* = `this` } . . . . . . }	class Director { //поле для зберігання першого створеного об"єкту static *directorRef* constructor(name, department) { if (Director.*directorRef*) return Director.*directorRef* this.name = name this.department = department Director.*directorRef* = this } toString() { return `${this.name} - ${this.department}` } }
	const d1 = new Director('Ivan', 'depratment 1') console.log(d1.name) //Ivan const d2 = new Director('Olga', 'depratment 1') console.log(d2.name) //Ivan console.log(d1 === d2) //true

Задача. Черговий. Дано спосок студентів одного курсу (ПІБ, курс). Розробити менеджер чергових, який дозволяє випадковим чином обирати і запам’ятовувати обраного чергового студента