


Open · Opened 2 weeks ago by  **Rubén Montero**

Registro horario

Resumen

- Se pedirá una clase que permita almacenar información de registro horario (hora de entrada y hora de salida) con un método, y, generar un JSON con la información acumulada

Descripción

El control del tiempo que alguien invierte en una ubicación o actividad es un problema recurrente en la informática: Control de asistencia a una reunión, control de jornada laboral, tiempo invertido en un curso online,...

Nosotros vamos a crear una clase que permita **grabar** tiempos de asistencia. No sabemos a *qué* actividad (trabajo, festival,...) pero **sí** sabemos que permitiremos grabar múltiples horas de entrada y salida.

La idea es la siguiente:

1. Una clase con un método que permite registrar horas de entrada y salida
2. Se invoca múltiples veces (e.g. 10-12h, 14-17h, 6-7h, 6-21h,...), suponiendo que cada registro es en un día distinto
3. Al final, se invoca otro método que genera un JSON con la información de todos los registros horarios

La tarea

Crearemos una clase `TimeRegister`.

Tendrá un método `public void addHours` que reciba **2** parámetros tipo `short`:

- `entranceHour`.
 - Debe ser `>= 0`, `< 24` y `< exitHour`. De lo contrario, se lanzará una excepción con `throw new RuntimeException();`
- `exitHour`
 - Debe ser `>= 0`, `< 24` y `> entranceHour`. De lo contrario, se lanzará una excepción con `throw new RuntimeException();`

Estos datos, pasados por parámetros, deberían ser almacenados (*añadidos*) a un atributo de tipo lista ó *array*, pues los usaremos a continuación.

Tendrá un método `produceJSON`:

- `public void`, **no** recibe parámetros
- Generará un objeto JSON y lo **guardará** en `assets\hours.json`
- Tendrá una forma *como* la del siguiente **ejemplo**:

```
{
  "totalHours": 23,
  "registers": [
    {
      "entranceHour": 0,
      "exitHour": 10
    },
    {
      "entranceHour": 0,
      "exitHour": 12
    },
    {
      "entranceHour": 19,
      "exitHour": 20
    }
  ]
}
```

(`totalTime` equivale a la suma de las horas registradas)

Pequeño comentario

Nótese que, para representar una fecha/hora es más adecuada la clase `Date`. Nosotros nos conformaremos con dos `short` de momento.

Por último

Verifica que el `test` funciona correctamente.

Haz `commit` y `push` para subir los cambios al repositorio.



Rubén Montero [@ruben.montero](#) changed milestone to [%Sprint 1](#) 2 weeks ago



Ania Blanco [@ania.blanco](#) mentioned in commit [0e837cb2](#) 1 week ago