Aplicaciones telemáticas
Examen final, prueba de teoría. 4 de junio de 2021
Grado en ingeniería telemática.
Universidad Rey Juan Carlos

Instrucciones:

- Ejecuta en un terminal `~mortuno/prepara`
- Esto dejará en el ordenador el fichero `~/at.junio.21/examen.TULOGIN.txt`, donde debes escribir tus respuestas. TULOGIN será tu nombre de usuario en el laboratorio.

Ejercicio 1 (4 puntos)

¿Qué significa *NaN* en JavaScript? ¿Para qué sirve? ¿Cómo se usa? ¿Qué problemas suele dar a los programadores principantes en JavaScript? ¿Cuál es su equivalente en la mayoría de lenguajes de programación?

Respuesta

NaN (*Not a Number*) es un valor de error devuelto por una operación. El programador esperaba un número, pero no es el caso. Bien por tratarse de un operación matemática que no devuelve un número real (como $\sqrt{-1}$)), bien porque no todos los operandos son numéricos cuando deberían serlo.

Suele resultar problemático para programadores no experimentados porque, cuando aparece, probablemente el flujo normal del programa debería interrumpirse para ser tratado como un error, no como un número más (paradójicamente, *NaN* es de tipo numérico). Además, no es posible detectarlo como una expresión como `x===NaN`, es necesario emplear la función *isNaN()*.

Es una peculiariad de JavaScript, en la mayoria de lenguajes lo equivalente sería elevar una excepción. Observaciones:

- *NaN* es muy distinto de *undefined* (variable o parámetro no inicializado) y muy distinto de *null* (objeto sin valor).
- *NaN* no es una función para comprobar si un valor es numérico o no.

Ejercicio 2 (3 puntos)

Como sabes, hay un tipo programación orientada a objetos basada en clases y otro basado en prototipos. Las clases son fuertemente acopladas, los prototipos presentan un acoplamiento más débil. Explica esto brevemente.

Respuesta

En POO basada en clases, un objeto hereda atributos y métodos de su clase y de todas las clases de nivel superior en la jerarquía, normalmente todos los atributos y todos los métodos, sean necesarios o no. Esto crea una dependencia muy fuerte de mucho código, potencialmente muy variado. Un cambio en cualquier clase influirá en cualquier objeto que dependa de ella o de una de sus clases descendientes.

En la POO basada en prototipos, no hay clases. Los objetos se crean a partir de otros objetos mediante factorías. Los atributos y métodos que se necesiten se copian, los que no, no. Ningún cambio en el objeto original afecta al objeto copiado.

Ejercicio 2 (3 puntos)

¿Para qué sirven los *web workers*?

Respuesta

Se trata de un API de JavaScript que permite ejecutar un hilo de código en segundo plano. Son especialmente útiles para tareas que consuman mucho tiempo, normalmente por ser intensivas en uso de CPU o de comunicaciones, porque así no interfieren a la tarea en primer plano donde está el interfaz de usuario.