Software Distribuït

Pràctica 0: Bytes i Streams

Pràctica 0

Els principals objectius d'aquesta sessió:

- Entendre i saber utilitzar les funcions que formen part de **ComUtils.java**
- Saber llegir i escriure correctament diferents tipus de dades
- Entendre la diferència entre les funcions i en quin cas utilitzar cadascuna d'elles.
- Ser capaç de modificar aquestes funcions de base per aconseguir escriure i llegir un missatge amb unes característiques concretes.
- Comprendre el test d'exemple i realitzar-ne de nous.
- Saber executar el codi i els tests desde l'editor i/o consola.

IMPRESCINDIBLE PER PODER FER LA PRÀCTICA 1

Pràctica 0

Se us demana fer una sèrie de modificacions per estendre les classes de la llibreria **utils** (**ComUtils.java** i **ComUtilsService.java**) , que s'utilitzarà i ampliarà al llarg del Projecte 1.

En aquesta sessió emularem el **DataInputStream** i **DataOutputStream** que utilitzarem més endavant en la comunicació entre sockets amb els que s'utilitzen per escriure i llegir per fitxer. D'aquesta manera podrem visualitzar i quantificar els bytes que s'escriuen/llegeixen.

Pràctica 0

Les funcions principals que componen **ComUtils.java** són:

- **String: read_string(int size)**
- **int: read_int32()**

- **write_string(String s)**
- **write_int32(int)**

Les funcions estan explicades amb més detall en l'enunciat corresponent.

EXERCICIS EXTRES PROPOSATS:

Realitzar l'escriptura/lectura d'un fitxer amb el següent format:

String[20] <El vostre nom i cognom>

int32 <Edat>

String[] <Comentari>

- Que observeu? Quants bytes s'han escrit en el fitxer? Tots són llegibles?
- En quin ordre s'han de llegir? Quines funcions heu fet servir?

Pràctica 0

Al final d'aquesta sessió hauríeu de ser capaços de respondre a les següents preguntes:

- Si volem que un int32 sigui llegible en un fitxer, que haurem de fer?
- Quines creus que poden ser altres funcions útils?

Pràctica 0

```
public class ComUtilsTest {
     @Test
     public void example_test() {
     File file = new File("test");
     try {
          file.createNewFile();
          ComUtils comUtils = new ComUtils(new FileInputStream(file),
          new FileOutputStream(file));

          comUtils.write_int32(2);
          int readedInt = comUtils.read_int32();
          assertEquals(2, readedInt);

     } catch (IOException e) {
          e.printStackTrace();
          }
     }
}
```

Pràctica 0

Serieu capaços de realitzar les funcions **write_char** i **read_char** i realitzar-ne els tests corresponents?

Aquí un link que us pot ser ultil: https://www.guru99.com/junit-assert.html