Licence MIASHS

INF F5 — Flux et fichiers

UGA UFR SHS Université Grenoble Alpes

Daniel.Bardou@univ-grenoble-alpes.fr

Sommaire

- Introduction
- Flux d'octets
  - en lecture
  - en écriture
- Flux de caractères
  - en lecture
  - en écriture
- D'autres classes pour les fichiers

Introduction

Notion de flux

- Un flux (de données) est l'abstraction d'une séquence de taille variable (voire inconnue) de données éventuellement hétérogènes qui peut constituer une entrée ou une sortie pour un programme.
- Un flux peut être associé à divers éléments du système d'entrées/sorties d'un ordinateur : clavier, écran, imprimante, fichier, connexion réseau, …

Les flux en Java

- Il existe plusieurs classes prédéfinies en Java permettant de créer et manipuler des flux. On distingue les flux selon 2 critères :
  - leur direction (flux en lecture ou en écriture),
  - l'élément atomique pouvant être lu ou écrit (octet ou caractère).
- Les classes relatives aux flux sont pour la plupart regroupées dans le paquetage java.io, mais aussi dans d'autres paquetages (java.util, java.util.zip, java.security, javax.crypto).

Flux d'octets

Flux d'octets

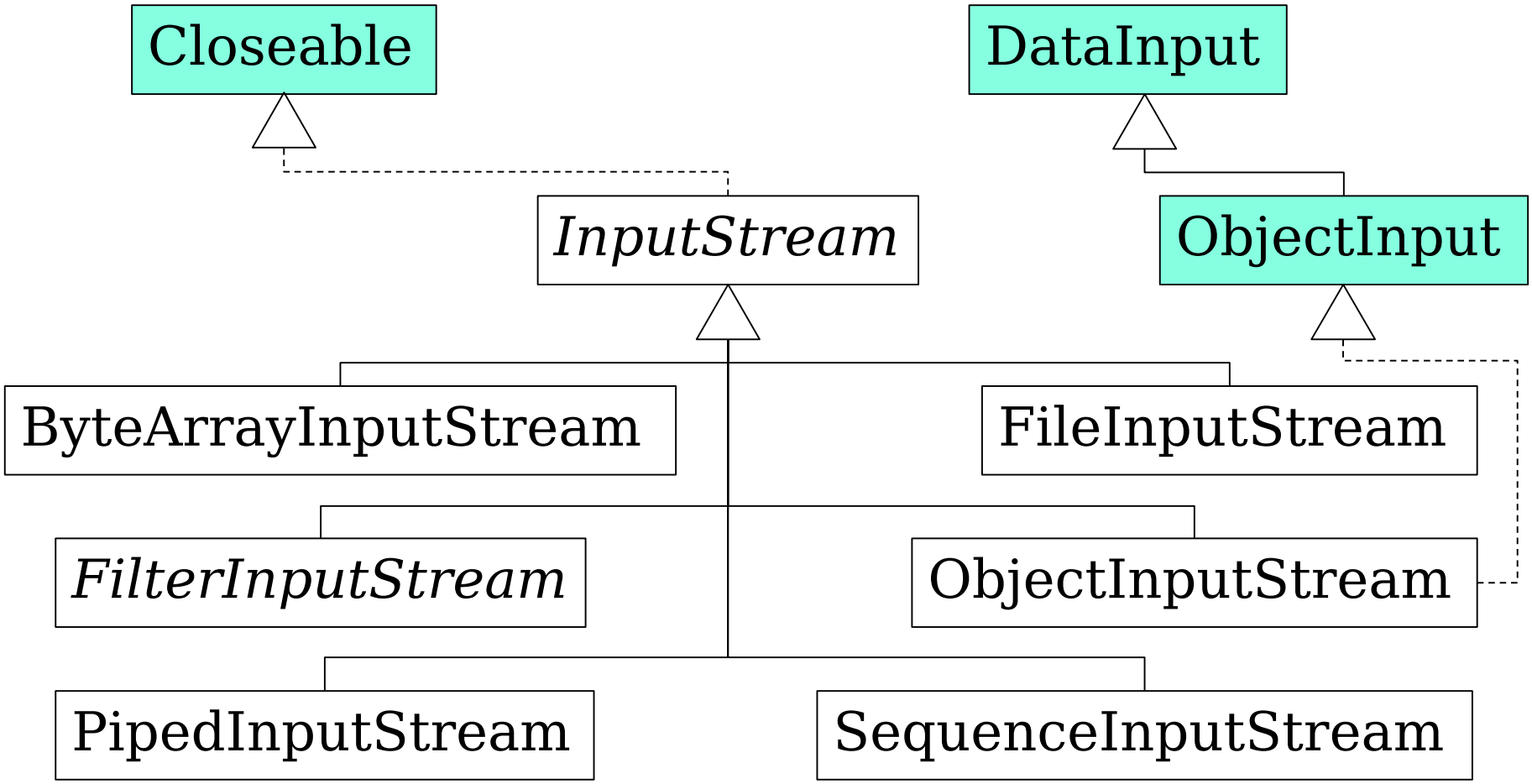
- Deux hiérarchies de classes prédéfinies existent en Java pour les flux d'octets.
- La classe `java.io.InputStream` est la racine de la hiérarchie d'héritage des classes concernant les flux d'octets en lecture (en entrée).
- La classe `java.io.OutputStream` est la racine de la hiérarchie d'héritage des classes concernant les flux d'octets en écriture (en sortie).

Interface Closeable

- Une seule méthode :
  void close() **throws** IOException
  ferme le flux et libère toutes les ressources systèmes associées au flux. N'a aucun effet si le flux est déjà fermé.

Flux d'octets en lecture

Flux d'octets en lecture



InputStream (abstraite)

- Méthodes de lecture
  - **abstract int** read() **throws** IOException
    lit et retourne l'octet suivant. Retourne -1 si EOF. Bloque jusqu'à la lecture de l'octet, EOF ou levée d'une IOException.
  - **int** read(**byte**[] t) **throws** IOException
    lit jusqu'à t.length octets, les range dans t et retourne le nombre d'octets lus ou -1 si EOF. Méthode bloquante. Lève une NullPointerException si t vaut **null**.

InputStream (abstraite)

- Méthodes de lecture (suite)
  - **int** read(**byte**[] t, **int** d, **int** lg) **throws** IOException
    lit jusqu'à lg octets, les range dans t à partir de l'indice d et retourne le nombre d'octets lus ou -1 si EOF. Méthode bloquante. Lève une NullPointerException si t vaut **null**.

InputStream (abstraite)

- Autres méthodes
  - **int** available()
    retourne le nombre d'octets disponibles (pouvant être lus sans bloquer avec read()).
  - **void** close() **throws** IOException
    ferme le flux.
  - **long** skip(**long** n) **throws** IOException
    saute n octets et retourne le nombre d'octets effectivement sautés.

InputStream (abstraite)

- Autres méthodes (suite)
  - **void** mark(**int** limite) **throws** IOException
    positionne une marque à laquelle il est possible de revenir par un reset. limite indique le nombre d'octets pouvant être lus avant que cette marque soit invalidée.
  - **boolean** markSupported() **throws** IOException
    retourne **true** si et seulement si le flux autorise l'utilisation de mark et reset.
  - **void** reset() **throws** IOException
    repositionne la lecture à la dernière marque placée dans le flux. Une IOException est levée si aucune marque valide existe ou si les marques ne sont pas autorisées par le flux.

FileInputStream

- Constructeurs
  - FileInputStream(String name) **throws** FileNotFoundException
    ouvre une connexion en lecture sur le fichier nommé name dans le système de fichiers. Levée d'exception s'il n'est pas possible de lire dans le fichier.
  - FileInputStream(File file) **throws** FileNotFoundException
    ouvre une connexion en lecture sur le fichier file. Levée d'exception s'il n'est pas possible de lire dans le fichier.
  - FileInputStream(FileDescriptor fd) **throws** FileNotFoundException
    initialise le FileInputStream d'après fd (pouvant être obtenu depuis une connexion existante).

FileInputStream

- Méthodes additionnelles
  - FileDescriptor getFD() **throws** IOException
    retourne le FileDescriptor représentant la connexion courante au fichier.
  - FileChannel getChannel() **throws** IOException
    retourne le FileChannel représentant la connexion courante au fichier (avec la même position de lecture).

FilterInputStream (abstraite)

- La classe FilterInputStream est une classe abstraite dont héritent de nombreuses sous-classes destinées à être des « décorateurs » pour un InputStream existant, dont notamment :
  - BufferedInputStream
  - DataInputStream
  - InflaterInputStream (dans le paquetage java.util.zip)
  - ...

BufferedInputStream

- La classe BufferedInputStream hérite de FilterInputStream.
- Constructeurs
  - BufferedInputStream(InputStream in)
    décore in en le dotant d'un accès tamponné de 8192 octets.
  - BufferedInputStream(InputStream in, **int** taille)
    décore in en le dotant d'un accès tamponné de taille octets.
- Aucune méthode supplémentaire

DataInputStream

- La classe DataInputStream hérite de FilterInputStream et réalise l'interface DataInput.
- Constructeur
  - DataInputStream(InputStream in) décore in en le dotant d'un accès conforme à l'interface DataInput.

DataInputStream

- Méthodes imposées par l'interface DataInput (levée d'une exception EOFException si tentative de lire au-delà de la fin du flux) :
  - **void** readFully(**byte**[] tab)
    lit tab.length octets dans le flux et les place dans tab, méthode bloquante.
  - **void** readFully(**byte**[] tab, **int** d, **int** l)
    lit l octets dans le flux et les place dans tab, à partir de l'indice d, méthode bloquante.
  - **int** skipBytes(**int** n)
    tente de sauter n octets dans la lecture du flux et retourne le nombre d'octets effectivement sautés.

DataInputStream

- Méthodes imposées par l'interface DataInput (levée d'une exception EOFException si tentative de lire au-delà de la fin du flux) :
  - **boolean** readBoolean()
    lit l'octet suivant et retourne **true** s'il est différent de 0, **false** sinon.
  - **byte** readByte()
    lit 1 octet.
  - **short** readShort()
    lit 2 octets.
  - **char** readChar()
    lit 2 octets.

DataInputStream

- Méthodes imposées par l'interface DataInput (levée d'une exception EOFException si tentative de lire au-delà de la fin du flux) :
  - **int** readInt()
    lit 4 octets.
  - **long** readLong()
    lit 8 octets.
  - **float** readFloat()
    lit 4 octets.
  - **double** readDouble()
    lit 8 octets.

DataInputStream

- Méthodes imposées par l'interface DataInput (levée d'une exception EOFException si tentative de lire au-delà de la fin du flux) :
  - **int** readUnsignedByte()
    lit 1 octet.
  - **int** readUnsignedShort()
    lit 2 octets.

DataInputStream

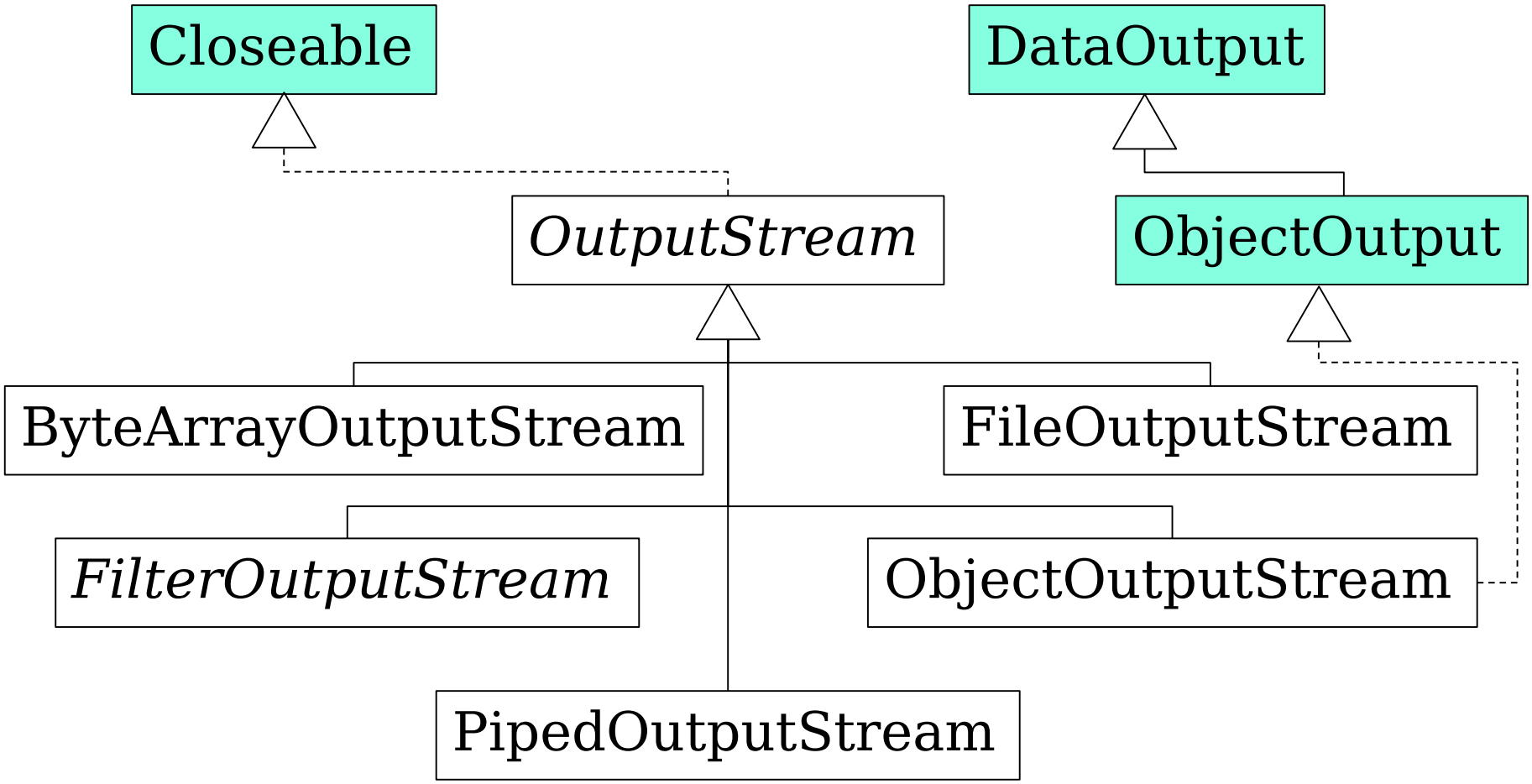
- Méthodes imposées par l'interface DataInput (levée d'une exception EOFException si tentative de lire au-delà de la fin du flux) :
  - String readLine()
    lit plusieurs octets en les interprétant chacun comme des caractères qui sont mis dans la String retournée. La lecture s'arrête à une fin de ligne ou en fin de flux.
  - String readUTF()
    lit plusieurs octets constituant une chaîne de caractères au format UTF8 modifié et retourne la String correspondante.

Exemple de lecture basique

```
InputStream entree;
byte b;
try {
      entree = new BufferedInputStream(
                              new FileInputStream("essai"));
      for (int i = entree.read(); i != -1; i = entree.read()) {
                  b = (byte) i;
         // ...
      }
      entree.close();
} catch (IOException ioe) {
      // ...
}
```

Flux d'octets en écriture

Flux d'octets en lecture



OutputStream (abstraite)

- Méthodes d'écriture
  - **abstract void** write(**int** i) **throws** IOException
    écrit l'octet de poids faible de i dans le flux.
  - **void** write(**byte**[] t) **throws** IOException
    écrit tous les octets de dans le flux. Lève une NullPointerException si t vaut **null**.
  - **void** write(**byte**[] t, **int** d, **int** l) **throws** IOException
    écrit l octets qui se trouvent dans le tableau t à partir de l'indice d. Lève une NullPointerException si t vaut **null**. Lève une IndexOutOfBoundsException si les valeurs de d et/ou l sont incorrectes.

OutputStream (abstraite)

- Autres méthodes
  - **void** flush() **throws** IOException
    vide le flux (assure que tout octet éventuellement « bufferisé » est bien envoyé à l'autre extrémité du flux.
  - **void** close() **throws** IOException
    ferme le flux.

FileOutputStream

- Constructeurs
  - FileOutputStream(String name, **boolean** ajout) **throws** FileNotFoundException
    ouvre une connexion en écriture sur le fichier nommé name dans le système de fichiers. Si le fichier n'existe pas, il est créé. Si le fichier existe et que ajout vaut **false**, il est écrasé, sinon il est augmenté. Levée d'exception s'il n'est pas possible d'écrire dans le fichier.
  - FileOutputStream(String name) **throws** FileNotFoundException
    correspond à FileOutputStream(name, **false**).
  - FileOutputStream(File file, **boolean** ajout) **throws** FileNotFoundException
    similaire à FileOutputStream(String, **boolean**).
  - FileOutputStream(File file) **throws** FileNotFoundException
    correspond à FileOutputStream(file, **false**).
  - FileOutputStream(FileDescriptor fd)
    ouvre une connexion en écriture décrite par fd.

FileOutputStream

- Méthodes additionnelles
  - `FileDescriptor getFD() throws IOException`
    retourne le `FileDescriptor` représentant la connexion courante au fichier.
  - `FileChannel getChannel() throws IOException`
    retourne le `FileChannel` représentant la connexion courante au fichier (avec la même position d'écriture).

FilterOutputStream (abstraite)

- La classe FilterOutputStream est une classe abstraite dont héritent de nombreuses sous-classes destinées à être des « décorateurs » pour un OutputStream existant, dont notamment :
  - BufferedOutputStream
  - DataOutputStream
  - DeflaterInputStream (dans le paquetage java.util.zip)
  - ...

BufferedOutputStream

- La classe BufferedOutputStream hérite de FilterOutputStream.
- Constructeurs
  - BufferedOutputStream(OutputStream out)
    décore out en le dotant d'un accès tamponné de 8192 octets.
  - BufferedOutputStream(OutputStream out, **int** taille)
    décore out en le dotant d'un accès tamponné de taille octets.
- Aucune méthode supplémentaire

DataOutputStream

- La classe DataOutputStream hérite de FilterOutputStream et réalise l'interface DataOutput.
- Constructeur
  - DataOutputStream(OutputStream out) décore out en le dotant d'un accès conforme à l'interface DataOutput.

DataOutputStream

- Méthodes imposées par l'interface DataOutput (levée possible d'une exception IOException) :
  - **void** writeBoolean(**boolean** b)
    écrit un booléen (un octet valant 1 si **true**, valant 0, si **false**).
  - **void** writeByte(**int** b)
    écrit l'octet de poids faible de b.
  - **void** writeShort(**int** s)
    écrit les 2 octets de poids faible de s.
  - **void** writeChar(**int** c)
    écrit les 2 octets de poids faible de c.

DataOutputStream

- Méthodes imposées par l'interface DataOutput (levée possible d'une exception IOException) :
  - **void** writeInt(**int** i)
    écrit 4 octets.
  - **void** writeLong(**long** l)
    écrit 8 octets.
  - **void** writeFloat(**float** f)
    écrit 4 octets.
  - **void** writeDouble(**double** d)
    écrit 8 octets.

DataOutputStream

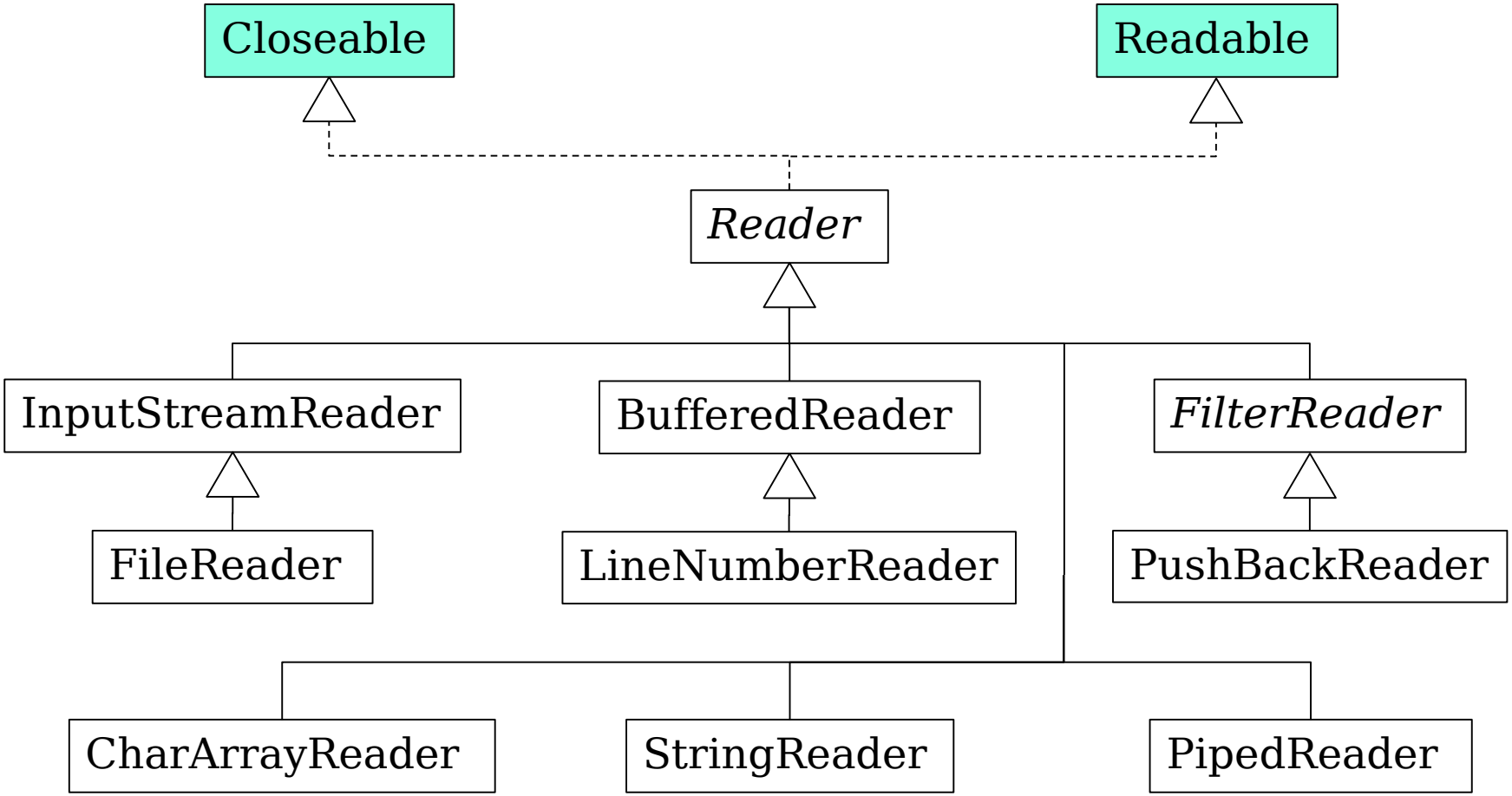
- Méthodes imposées par l'interface DataOutput (levée possible d'une exception IOException) :
  - **void** writeBytes(String s)
    écrit les octets de poids faible de chacun des caractères de s.
  - **void** writeChars(String s)
    écrit les caractères de s.
  - **void** writeUTF(String s)
    écrit 2 octets indiquant un format UTF-8, puis les caractères de s en utilisant le format UTF-8.

Exemple d'écriture basique

```
OutputStream sortie;
byte b;
try {
      sortie = new BufferedOutputStream(
                              new FileOutputStream("essai"));
      for (int i = 0; i < 1000000; i ++) {
            b = (byte)(Math.random() * 256 - 128);
            sortie.write(b);
         // ...
      }
      sortie.flush();
      sortie.close();
} catch (IOException ioe) {
      // ...
}
```

Flux de caractères en lecture

Flux de caractères en lecture



Reader (abstraite)

- Constructeurs
  - Reader()
    permet d'obtenir un Reader synchronisé avec lui-même.
  - Reader(Object lock)
    permet d'obtenir un Reader synchronisé avec lock.

Reader (abstraite)

- Méthodes de lecture
  - **int** read() **throws** IOException
    lit et retourne le caractère suivant. Retourne -1 si EOF. Bloque jusqu'à la lecture du caractère, EOF ou levée d'une IOException.
  - **int** read(**char**[] t) **throws** IOException
    lit jusqu'à t.length caractères, les range dans t et retourne le nombre d'octets lus ou -1 si EOF. Méthode bloquante. Lève une NullPointerException si t vaut **null**.

Reader (abstraite)

- Méthodes de lecture (suite)
  - **int** read(**char**[] t, **int** d, **int** l) **throws** IOException
    jusqu'à l caractères, les range dans t et retourne le nombre d'octets lus ou -1 si EOF. Méthode bloquante. Lève une NullPointerException si t vaut **null**.
  - **int** read(CharBuffer cb) **throws** IOException
    tente de lire des caractères dans cb et retourne le nombre d'octets lus ou -1 si EOF. (Méthode imposée par l'interface Readable).

Reader (abstraite)

- Autres méthodes
  - **void** close() **throws** IOException
    ferme le flux.
  - **boolean** ready() **throws** IOException
    retourne **true** si et seulement si une tentative de lecture ne serait pas bloquante.
  - **long** skip(**long** n) **throws** IOException
    saute n octets et retourne le nombre d'octets effectivement sautés.

Reader (abstraite)

- Autres méthodes (suite)
  - **void** mark(**int** limite) **throws** IOException
    positionne une marque à laquelle il est possible de revenir par un reset. limite indique le nombre d'octets pouvant être lus avant que cette marque soit invalidée.
  - **boolean** markSupported() **throws** IOException
    retourne **true** si et seulement si le flux autorise l'utilisation de mark et reset.
  - **void** reset() **throws** IOException
    repositionne la lecture à la dernière marque placée dans le flux. Une IOException est levée si aucune marque valide existe ou si les marques ne sont pas autorisées par le flux.

InputStreamReader

- Constructeurs
  - InputStreamReader(InputStream in)
    permet d'obtenir un InputStreamReader sur in en utilisant le Charset par défaut.
  - InputStreamReader(InputStream in, Charset cs)
    permet d'obtenir un InputStreamReader sur in en utilisant cs comme Charset.
  - InputStreamReader(InputStream in, CharsetDecoder dec)
    permet d'obtenir un InputStreamReader sur in en utilisant dec comme CharsetDecoder.
  - InputStreamReader(InputStream in, String name) **throws** UnsupportedEncodingException
    permet d'obtenir un InputStreamReader sur in en utilisant le Charset nommé name.

FileReader

- Constructeurs
  - FileReader(String name) **throws** FileNotFoundException
    ouvre une connexion en lecture sur le fichier nommé name dans le système de fichiers. Levée d'exception s'il n'est pas possible de lire dans le fichier.
  - FileReader(File file) **throws** FileNotFoundException
    ouvre une connexion en lecture sur le fichier file. Levée d'exception s'il n'est pas possible de lire dans le fichier.
  - FileReader(FileDescriptor fd) **throws** FileNotFoundException
    initialise le FileInputStream d'après fd (pouvant être obtenu depuis une connexion existante).

BufferedReader

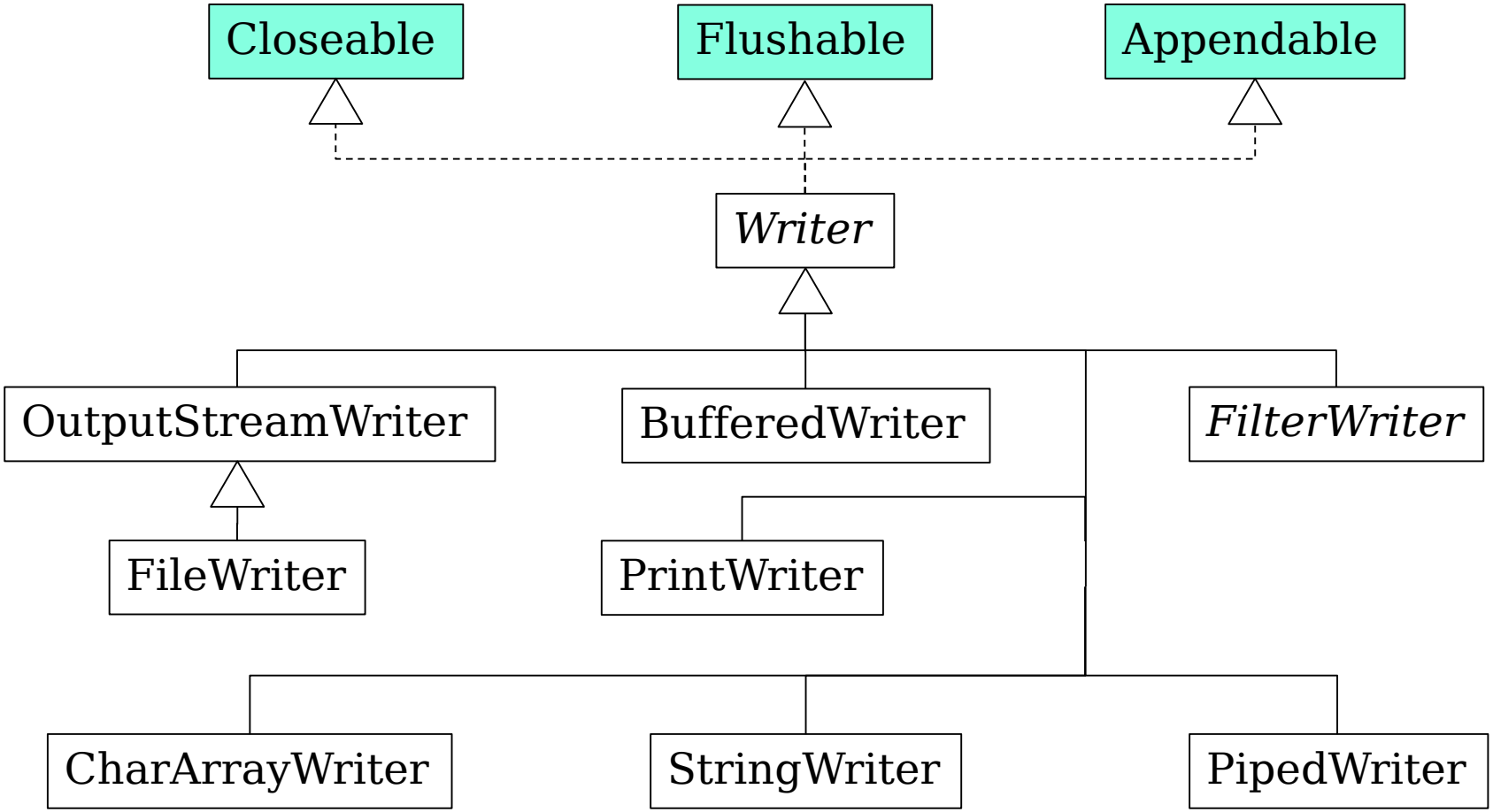
- Constructeurs
  - `BufferedReader(Reader r)`
    permet d'obtenir un `BufferedReader` sur `r` avec un tampon de taille `8192`.
  - `BufferedReader(Reader r, int t)`
    permet d'obtenir un `BufferedReader` sur `r` avec un tampon de taille `t`.

BufferedReader

- Méthode additionnelle
  - String readLine()
    retourne une ligne lue dans **this**. Une ligne est toute suite de caractères se terminant par '\n', '\r', ou "\r\n". Les caractères de fin de ligne ne font pas partie du résultat. Retourne **null** si EOF.

Flux de caractères en écriture

Flux de caractères en écriture



Writer (abstraite)

- Constructeurs
  - Writer()
    permet d'obtenir un Writer synchronisé avec lui-même.
  - Writer(Object lock)
    permet d'obtenir un Writer synchronisé avec lock.

Writer (abstraite)

- Méthodes d'écriture
  - **void** write(**int** i) **throws** IOException
    écrit le caractère encodé dans les 2 octets de poids faible de i dans le flux.
  - **void** write(**char**[] t) **throws** IOException
    écrit tous les éléments de t dans le flux.
  - **void** write(**char**[] t, **int** d, **int** l) **throws** IOException
    écrit les l éléments de t situés à partir de l'indice d dans le flux.

Writer (abstraite)

- Méthodes d'écriture (suite)
  - **void** write(String s) **throws** IOException
    écrit tous les caractères de s dans le flux.
  - **void** write(String s, **int** d, **int** l) **throws** IOException
    écrit les l caractères de s situés à partir de l'indice d dans le flux.

Writer (abstraite)

- Méthodes d'écriture (suite)
  - Writer append(**char** c) **throws** IOException
    écrit c dans le flux et retourne le flux (permet la cascade de messages).
  - Writer append(CharSequence cs) **throws** IOException
    écrit tous les caractères de cs dans le flux et retourne le flux. String, StringBuffer, StringBuilder en particulier implémentent l'interface CharSequence.
  - Writer append(CharSequence cs, **int** d, **int** f) **throws** IOException
    écrit les caractères de cs de l'indice d inclus à l'indice f exclus dans le flux et retourne le flux.

Writer (abstraite)

- Autres méthodes
  - **void** close() **throws** IOException
    ferme le flux.
  - **void** flush() **throws** IOException
    vide le flux.

OutputStreamWriter

- Constructeurs
  - OutputStreamWriter(OutputStream out)
    permet d'obtenir un OutputStreamWriter sur out en utilisant le Charset par défaut.
  - OutputStreamWriter(OutputStream out, Charset cs)
    permet d'obtenir un OutputStreamWriter sur out en utilisant cs comme Charset.
  - OutputStreamWriter(OutputStream out, CharsetDecoder dec)
    permet d'obtenir un OutputStreamWriter sur out en utilisant dec comme CharsetDecoder.
  - OutputStreamWriter(OutputStream out, String name) **throws** UnsupportedEncodingException
    permet d'obtenir un OutputStreamWriter sur out en utilisant le Charset nommé name.

FileWriter

- Constructeurs
  - FileWriter(String name, **boolean** ajout) **throws** IOException
    ouvre une connexion en écriture sur le fichier nommé name dans le système de fichiers. Si le fichier n'existe pas, il est créé. Si le fichier existe et que ajout vaut **false**, il est écrasé, sinon il est augmenté. Levée d'exception s'il n'est pas possible d'écrire dans le fichier.
  - FileWriter(String name) **throws** IOException
    correspond à FileWriter(name, **false**).
  - FileOutputStream(File file, **boolean** ajout) **throws** IOException
    similaire à FileWriter(String, **boolean**).
  - FileWriter(File file) **throws** IOException
    correspond à FileWriter(file, **false**).
  - FileWriter(FileDescriptor fd)
    ouvre une connexion en écriture décrite par fd.

BufferedWriter

- Constructeurs
  - `BufferedWriter(Writer w)`
    permet d'obtenir un `BufferedWriter` sur `w` avec un tampon de taille `8192`.
  - `BufferedWriter(Writer w, int t)`
    permet d'obtenir un `BufferedWriter` sur `w` avec un tampon de taille `t`.

BufferedWriter

- Méthode additionnelle
  - **void** newLine()
    écrit un séparateur de ligne dans le flux (dépend du système).