



# Gestion des Fichiers – NIO2

Dans l'API NIO2 les chemins sont encapsulés dans des objets de l'interface **Path**

Les objets de l'interface Path sont associés indifféremment à des **fichiers** ou des **répertoires**.



# Gestion des Fichiers – NIO2

Chemin d'accès à  
un fichier

Création:

```
Path monChemin1 = Paths.get("c:\\temp\\essai.txt");
```

ou

```
Path monChemin1 = Paths.get("c:/temp/essai.txt");
```

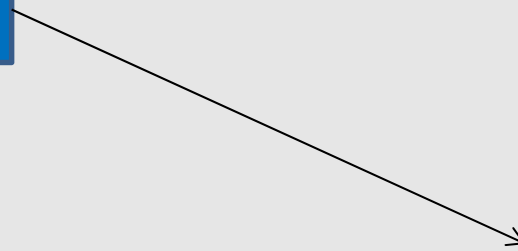
Chemin d'accès  
à un répertoire

```
Path monChemin2 = Paths.get("c:/temp");
```



# Gestion des Fichiers – NIO2

Chemin absolu

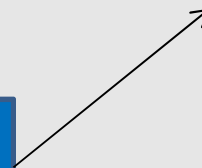


```
Path monChemin1 = Paths.get("c:/temp/essai.txt");
```

ou

```
Path monChemin2 = Paths.get("essai.txt");
```

Pas de chemin absolu , le chemin  
est lié au répertoire courant





# Gestion des Fichiers – NIO2

Récupération du chemin absolu:

**String toAbsolutePath()**

```
Path monChemin = Paths.get("essai.txt");
```

```
System.out.println(monchemin.toAbsolutePath());
```

Affichage Console:

```
D:\travail\Java\ProjetNIO2\essai.txt;
```



# Gestion des Fichiers – NIO2

Récupération du nom du Fichier/Répertoire:

**Path** `getFileName()`

```
Path monChemin = Paths.get("c:\\temp\\essai.txt");
```

```
System.out.println(monchemin.getFileName());
```

Affichage Console: `essai.txt`;



# Gestion des Fichiers – NIO2

Manipulation des fichiers:

Utilisation de la classe **Files**

La classe Files contient des méthodes statiques pour manipuler les fichiers



# Gestion des Fichiers – NIO2

Vérification de l'existence du Fichier/Répertoire dans le système de fichier:

```
boolean exists( Path path )
```

```
Path monChemin = Paths.get("c:\\temp\\essai.txt");  
boolean isExists = Files.exists(monChemin);
```



# Gestion des Fichiers – NIO2

Vérification du type d'élément associé à l'objet Path:

Répertoire ou Fichier

`boolean isRegularFile(Path path)`

`boolean isDirectory(Path path )`

Taille du fichier:

`long size(Path path)`

Si l'élément est un répertoire size retourne 0





# Gestion des Fichiers – NIO2

## Parcours d'un répertoire :

//Creation du Path associé au répertoire

```
Path ch = Paths.get("c:/temp");
```

//Creation d'un flux

```
DirectoryStream<Path> stream = Files.newDirectoryStream(ch);
```

//Parcours du flux avec une boucle for each

```
for(Path p : stream){  
    System.out.println (p.toString());  
}
```

**! newDirectoryStream peut lever une exception**



# Gestion des Fichiers – NIO2

## Création d'un fichier:

//Creation du Path associé au fichier

```
Path ch = Paths.get("c:/temp/essai.txt");
```

//Creation du fichier

```
Files.createFile(ch);
```

**! createFile peut lever une exception qui doit être contrôlée**



# Gestion des Fichiers – NIO2

## Exercice :

- Afficher le contenu d'un répertoire sous la forme d'une liste de fichiers ou de dossiers qu'il contient. (choisir un répertoire qui contient plusieurs dossiers ou fichiers).
- Afficher le contenu de ce répertoire en rajoutant un prefixe –F s'il s'agit d'un fichier, -D s'il s'agit d'un répertoire.
- Afficher pour les fichiers uniquement leur taille.
- Créer deux ArrayList, un pour les fichiers, un pour les répertoires pour contenir les objets du dossier étudié. Afficher ensuite les éléments de ces deux ArrayList.