Java8 - Alapértelmezett metódustörzs interfészekben

Aki követi a Java fejldését, találhat egy érdekességet a JDK8 (Lambda build) kipróbálható verziójában, amely letölthet a http://jdk8.java.net/lambda/ oldalról: az interfészekben meg tudunk adni alapértelmezett kódrészletet, amely felüldefiniálható a leszármazott osztályban. A megoldás elnye, hogy ezzel lehetségünk van egy interfészhez hozzáadni egy új metódust, s bevezetéséhez nem kell az **összes** leszármazottban ezt az új metódust implementálnunk.

Az interfész

Nézzünk egy példát, hozzunk létre egy interfészt, amelynek legyen egy meglév metódusa, illetve egy új metódust, amelyet nem szeretnénk a leszármazott osztályokban implementálni:

LambdaInterface.java

```
interface LambdaInterface
{
  public void oldMethod();

  public void newMethod() default
  {
    System.out.println("Default behaviour");
  }
}
```

A megfelel JDK8 használatával könnyedén lefordíthatjuk:

```
$ /opt/jdk1.8.0/bin/java -version
openjdk version "1.8.0-ea"
OpenJDK Runtime Environment (build 1.8.0-ea-lambda-nightly-h161-20120511-b39-b00)
OpenJDK 64-Bit Server VM (build 24.0-b07, mixed mode)
$ /opt/jdk1.8.0/bin/javac LambdaInterface.java
```

Az üres implementáció

Készítsünk egy implementációt, amely nem implementál egy metódust sem:

LambdaImplementation.java

```
public class LambdaImplementation implements LambdaInterface
{
}
```

Fordításkor nyilván panaszkodni fog a fordító, hogy nem implementáltuk az *oldMethod* metódust:

```
$ /opt/jdk1.8.0/bin/javac LambdaImplementation.java
LambdaImplementation.java:1: error: LambdaImplementation is not abstract and does not override abstract method
oldMethod() in LambdaInterface
public class LambdaImplementation implements LambdaInterface
       ^
1 error
```

Implementáció egy metódussal

Implementáljuk a hiányolt metódust:

LambdaImplementation.java

```
public class LambdaImplementation implements LambdaInterface
{
  public void oldMethod()
  {
    System.out.println("Implemented oldMethod");
  }
}
```

Majd fordítsuk le, amely során azt kell tapasztalnunk, hogy hiba nélkül fordul:

```
$ /opt/jdk1.8.0/bin/javac LambdaImplementation.java
$
```

Írjunk gyorsan futtató osztályt, amely egyszeren példányosítja az implementációt, majd meghívja az interfész két metódusát:

LambdaRun.java

```
public class LambdaRun
{
  public static void main(String[] args)
  {
    LambdaInterface li = new LambdaImplementation();
    li.oldMethod();
    li.newMethod();
  }
}
```

A fordítás után, illetve futtatás során az alábbit kell lássuk:

```
$ /opt/jdk1.8.0/bin/javac LambdaRun.java
$ /opt/jdk1.8.0/bin/java LambdaRun
Implemented oldMethod
Default behaviour
$
```

A newMethod metódust nem kellett implementálnunk, s a futtatás során lefutott a általunk megadott alapértelmezett metódustörzs.

Implementáció két metódussal

Amint az kitalálható: ha implementáljuk az új metódust is...

LambdaImplementation.java

```
public class LambdaImplementation implements LambdaInterface
{
  public void oldMethod()
  {
    System.out.println("Implemented oldMethod");
  }

  public void newMethod()
  {
    System.out.println("Implemented newMethod");
  }
}
```

...akkor az implementáció metódustörzse fog lefutni:

```
$ /opt/jdk1.8.0/bin/javac LambdaImplementation.java
$ /opt/jdk1.8.0/bin/java LambdaRun
Implemented oldMethod
Implemented newMethod
```

Két interfészben azonos alapértelmezett metódus?

Az emberben felmerül a kérdés, hogy a megoldással viszünk-e a platformba bizonytalanságot, ha két interfészben azonos metódus szignatúrát vértezünk fel alapértelmezett metódussal, ezért hozzuk létre egy új interfészt:

LambdaInterfaceOther.java

```
interface LambdaInterfaceOther
{
  public void oldMethod();

  public void newMethod() default
  {
    System.out.println("Other default behaviour");
  }
}
```

Látható, hogy a két interfész azonos metódusokat tartalmaz, ám a *newMethod* metódus mindkettben tartalmaz különböz metódustörzset. Módosítsuk az implementációt, implementálja mind a két interfészt:

LambdaImplementation.java

```
public class LambdaImplementation implements LambdaInterface, LambdaInterfaceOther
{
  public void oldMethod()
  {
    System.out.println("Implemented oldMethod");
  }
}
```

A fordításnál kiderül, hogy ez a módszer így nem mködik:

```
$ /opt/jdk1.8.0/bin/javac LambdaImplementation.java
LambdaImplementation.java:1: error: class LambdaImplementation inherits unrelated defaults for newMethod() from
types LambdaInterface and LambdaInterfaceOther
public class LambdaImplementation implements LambdaInterface, LambdaInterfaceOther
       ^
1 error
```

Két leszármazott interfész implementálása

A fenti példát továbbgondolva nézzünk meg egy olyan esetet, amikor egy interfészt két újabb interfész bvít ki, majd az implementáció megvalósítja ezen két új interfész összes metódusát. A próbához javítsuk ki a fent elrontott *LambdaInterfaceOther* interfészt:

LambdaInterfaceOther.java

```
interface LambdaInterfaceOther extends LambdaInterface
{
  public void otherMethod();
}
```

Majd hozzunk létre egy új interfészt:

LambdaInterfaceFoo.java

```
interface LambdaInterfaceFoo extends LambdaInterface
{
  public void fooMethod();
}
```

Az interfészekkel készen is lennénk, már csak módosítani kell az implementációt:

LambdaImplementation.java

```
public class LambdaImplementation implements LambdaInterfaceOther, LambdaInterfaceFoo
{
  public void oldMethod()
  {
    System.out.println("Implemented oldMethod");
  }

  public void otherMethod()
  {
    System.out.println("Implemented otherMethod");
  }

  public void fooMethod()
  {
    System.out.println("Implemented fooMethod");
  }
}
```

A LambdaRun osztály kicsit módosítanunk kell:

LambdaRun.java

```
public class LambdaRun
{
  public static void main(String[] args)
  {
    LambdaInterface li = new LambdaImplementation();
    li.oldMethod();
    li.newMethod();
    ((LambdaInterfaceOther)li).otherMethod();
    ((LambdaInterfaceFoo)li).fooMethod();
  }
}
```

Az eredmény:

```
$ /opt/jdk1.8.0/bin/javac LambdaInterface.java LambdaInterfaceOther.java LambdaInterfaceFoo.java
LambdaImplementation.java LambdaRun.java
$ /opt/jdk1.8.0/bin/java LambdaRun
Implemented oldMethod
Default behaviour
Implemented otherMethod
Implemented fooMethod
```

Vélemény?

Ha egy meglév interfész kibvítésére volt szükségünk, akkor két lehetségünk volt:

- megkeressük az összes leszármazottat, ahol implementálunk egy üres metódust
- létrehoztunk egy új interfészt a régi mellé, és a régit békén hagytuk

Ez a megoldás hasznos dolognak tnik... szerinted is? 🙂