# Ukázkový generátor terénů v. 2016/2017

Jednoduchá aplikace pro vytváření binárního souboru s definicí terénu, pozice střelce a cíle. Formát souboru viz zadání samostatné práce.

Vedlejším smyslem je ukázat vzorovou podobu odevzdávané samostatné práce.

## Struktura archivu

* .project, .classpath – Soubory projektu pro Eclipse.
* bin\ – Obsahuje přeloženou aplikaci.
* doc\ – Obsahuje tuto dokumentaci a Javadoc dokumentaci.
* src\ – Obsahuje zdrojový kód aplikace.
* Run.cmd – Spustí aplikaci a předá jí parametry příkazové řádky.  
   Aplikace spuštěná bez parametrů vypíše nápovědu, jaké parametry vyžaduje.
* Demo.cmd – Ukázkově spustí aplikaci.  
   Dávka nečeká žádné parametry, na rozdíl od dávky Run.cmd
* Build.cmd – Přeloží aplikaci; vyžaduje, aby byl dostupný překladač Javy (javac).  
   Dávka nečeká žádné parametry.
* Makedoc.cmd – Vygenerování Javadoc dokumentace do adresáře doc\javadoc.

## Popis implementace

Aplikace je tak jednoduchá, že nevyžaduje komplikovanou objektovou dekompozici. Obsahuje pouze několik statických metod pro generování terénu a jeho uložení do souboru. Metoda main() se postará o dekódování parametrů příkazové řádky a spuštění příslušné funkcionality.

V samostatné práci je třeba implementovat čtení binárního souboru, a proto za pozornost stojí implementace uložení terénu do binárního souboru v metodě saveTerrain(). Pro výstup do souboru se používá třída DataOutputStream a její metoda writeInt(). Intuitivně se dá odhadnout, že Java bude nabízet i třídu DataInputStream, která asi bude mít metodu readInt(). Úprava metody saveTerrain() na metodu, která terén načte, je proto snadná.

Za pozornost stojí i implementace dekódování parametrů příkazové řádky v metodě main(String args[]). Méně zkušený programátor by asi napsal podobný kód:

...

double value = Double.parseDouble(args[1]);

int width = Integer.parseInt(args[2]);

int height = Integer.parseInt(args[3]);

...

To má dvě nevýhody: za prvé se v číslech 1, 2, 3, … snadno udělá chyba, za druhé je krajně nepohodlné měnit pořadí parametrů příkazové řádky, jejich počet apod. Aplikace to implementuje lépe:

int argNo = 0;

...

double value = Double.parseDouble(args[argNo++]);

int width = Integer.parseInt(args[argNo++]);

int height = Integer.parseInt(args[argNo++]);

...

Kromě zjevně menší šance udělat chybu je zde další výhoda: pokud při zpracování parametru dojde k chybě, můžeme uživateli snadno říct, v kterém parametru udělal chybu:

System.err.println("Chyba v zadani parametru " + argNo);