



Tvorba tried

Informatika 1

Z prednášky

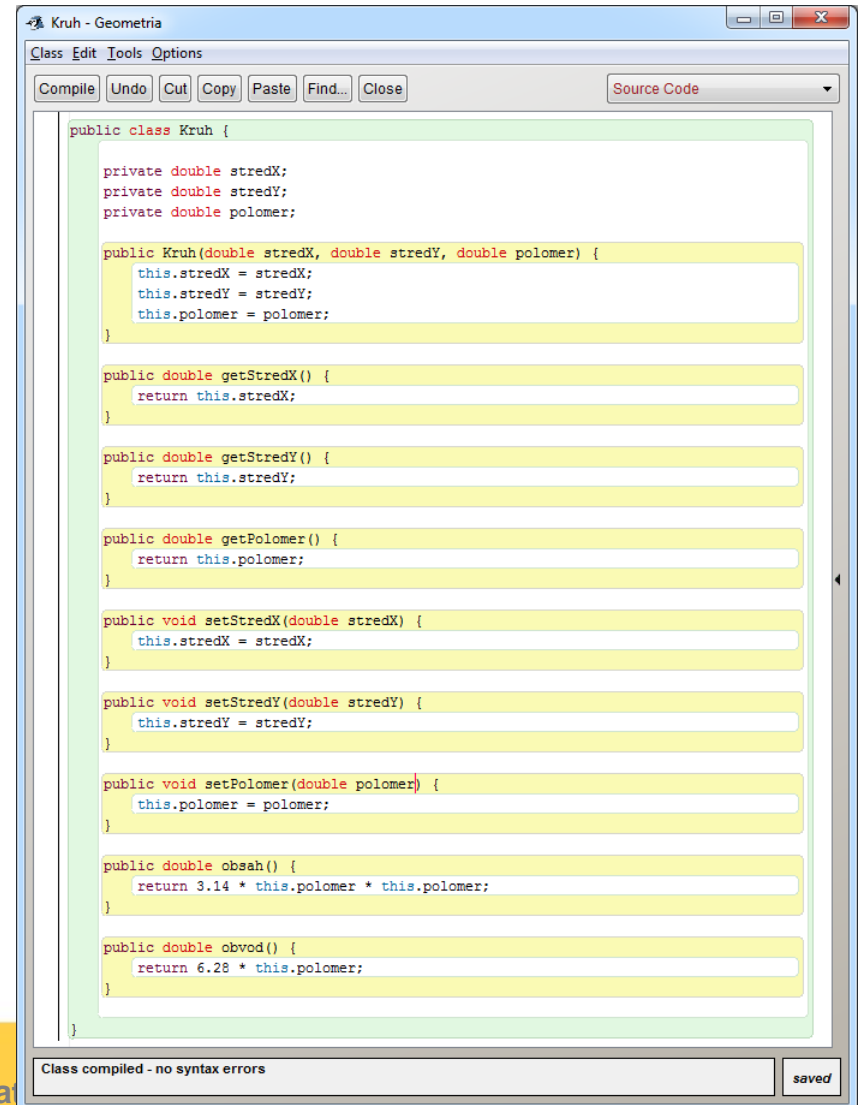
- Čo je trieda?
- Čo je to dátový typ?
- Aké dátové typy poznáte?
- Čo je atribút?
- Čo je metóda?
- Čo je konštruktor?

Cieľ cvičenia

- Rozbor jednoduchej triedy.
- Vytvorenie jednoduchej triedy.
 - Atribúty.
 - Konštruktor.
 - Metódy.
- Štandardný výstup.

Projekt geometria

- Stiahnite si z moodle projekt **geometria**.
- Preskúmajte zdrojový kód ukážkovej triedy Kruh:
 - Aké má atribúty?
 - Aké správy je možné poslať inštancii triedy?
 - Aké sú (formálne) parametre konštruktora triedy?



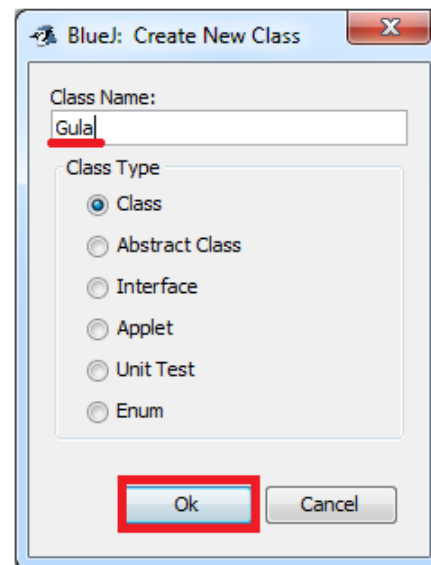
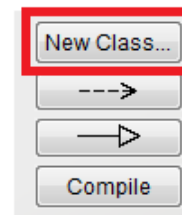
```
public class Kruh {  
    private double stredX;  
    private double stredY;  
    private double polomer;  
  
    public Kruh(double stredX, double stredY, double polomer) {  
        this.stredX = stredX;  
        this.stredY = stredY;  
        this.polomer = polomer;  
    }  
  
    public double getStredX() {  
        return this.stredX;  
    }  
  
    public double getStredY() {  
        return this.stredY;  
    }  
  
    public double getPolomer() {  
        return this.polomer;  
    }  
  
    public void setStredX(double stredX) {  
        this.stredX = stredX;  
    }  
  
    public void setStredY(double stredY) {  
        this.stredY = stredY;  
    }  
  
    public void setPolomer(double polomer) {  
        this.polomer = polomer;  
    }  
  
    public double obsah() {  
        return 3.14 * this.polomer * this.polomer;  
    }  
  
    public double obvod() {  
        return 6.28 * this.polomer;  
    }  
}
```

Class compiled - no syntax errors

saved

Vytvorenie vlastnej triedy

- Zvoľte si teleso (valec, kváder, guľa, ihlan, kužeľ,...).
- Aké vlastnosti (atribúty) má Vami zvolené teleso?
- Vytvorte novú triedu, ktorá bude toto teleso reprezentovať.



Definícia atribútov telesa

- Syntax:

```
private typ_atribútu názovAtribútu;
```

- Atribút je **VŽDY súkromný**, inak by bolo porušené zapúzdrenie – základný princíp OOP.

Definícia atribútov telesa

- Definujte atribúty pre Vaše teleso.

```
/**
 * Trieda reprezentuje gulu s danym polomerom.
 *
 * @author Michal Varga
 * @version 0.1
 */
public class Gula {

    private double polomer;

    /**
     * Constructor for objects of class Gula
     */
    public Gula() {
        // initialise instance variables
    }

}
```

Konštruktor

- Syntax:

```
public NázovTriedy([typ_param názovParam, ...]) {  
}
```

- Je zodpovedný za uvedenie objektu do správneho počiatočného stavu: priradí atribútom hodnoty z parametrov.
- Je verejný, názov má ako trieda, parametre sú voliteľné.

Konštruktor

- Prirad'ovací príkaz =.
- Na čo slúži `this`?
- Definujte konštruktor pre Vaše teleso.

```
/**  
 * Constructor for objects of class Gula  
 */  
public Gula(int polomer) {  
    // initialise instance variables  
    this.polomer = polomer;  
}
```

Getter

- Syntax:

```
public typ_atribútu getNázovAtribútu() {  
    return this.názovAtribútu;  
}
```

- Veľmi jednoduchá metóda.
- Sprístupňuje informácie o vnútornom stave inštancie.
- Návratová hodnota je takého istého typu ako je typ atribútu.

Getter

- Definujte gettery pre všetky atribúty Vášho telesa.

```
public double getPolomer() {  
    return this.polomer;  
}
```

Setter

- Syntax:

```
public void setNázovAtribútu(typ_atribútu názovParam) {  
    this.názovAtribútu = názovParametra;  
}
```

- Veľmi jednoduchá metóda.
- Mení vnútorný stav inštancie tak, že nastaví hodnotu konkrétneho atribútu.
- Nie vždy je táto metóda nutná, **niekedy je dokonca nebezpečná!**
- Nemá návratovú hodnotu!
- Typ (a názov) parametra sú zhodné s typom (a názvom) atribútu.

Setter

- Definujte settery pre vhodné atribúty Vášho telesa.

```
public void setPolomer(double polomer) {  
    this.polomer = polomer;  
}
```

Metódy

- Syntax:

```
public typ_návrat názovMetódy ([typ_param názovParam, ...]) {  
}
```

- Reakcie na správy – selektor správy je zhodný s názvom metódy.
- Názov metódy začína **malým** písmenom, každé ďalšie slovo **veľkým** písmenom (camel case).
- Parametre sú nepovinné, zátvorky musia byť uvedené aj keď metóda nemá parametre.
- Príkaz **return** ukončuje metódu a vracia výsledok (podľa jej návratového typu).
- Metódy, ktorých návratový typ je **void** neobsahujú príkaz **return**.

Metódy

- Definujte metódy na výpočet povrchu a objemu Vášho telesa.

```
public double obsah() {  
    return 4 * 3.14159 * this.polomer * this.polomer;  
}
```

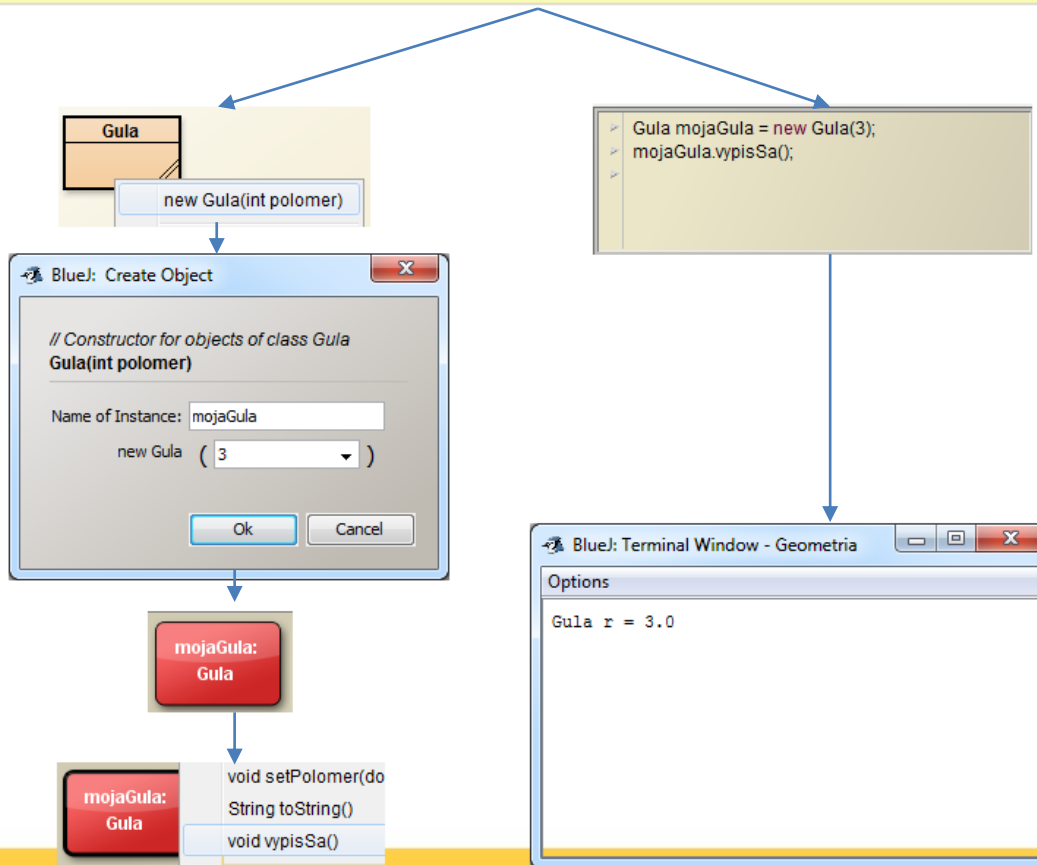
```
public double objem(){  
    return 4/3 * 3.14159 * this.polomer * this.polomer * this.polomer;  
}
```

Vypísanie informácií o telese

- Objekt zodpovedný za štandardný výstup (terminál) je `System.out`
- Je mu možné poslať rôzne správy, pre nás sú dôležité `print` a `println`.
- Reťazce a literály ("`toto je literál`") sa dajú spájať pomocou `+`.
- Vytvorte metódu `vypisSa`, ktorá pomocou `System.out` vypíše informácie o telese.

Vypísanie informácií o telese

```
public void vypisSa() {  
    System.out.print("Gula r = ");  
    System.out.println(this.polomer);  
}
```



Metóda toString

- Syntax:

```
public String toString() {  
}
```

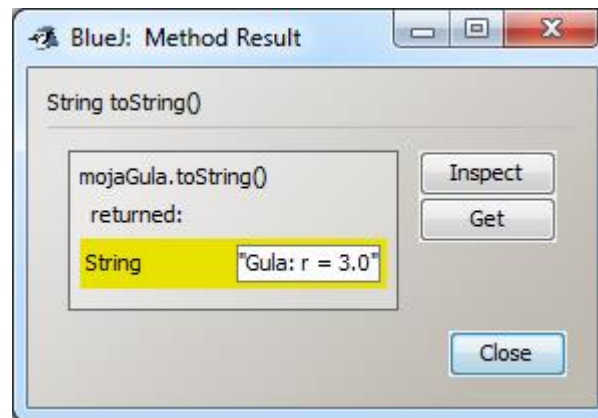
- Úlohou metódy je vrátiť vhodnú reťazcovú (`String`) reprezentáciu objektu.
- JAVA Konvertuje primitívne typy sa na reťazce automaticky.
- Na konverziu objektových typov na reťazce využíva JAVA správu `toString` - **každý** objekt vie reagovať na správu `toString`.

Metóda toString pre vytvorené teleso

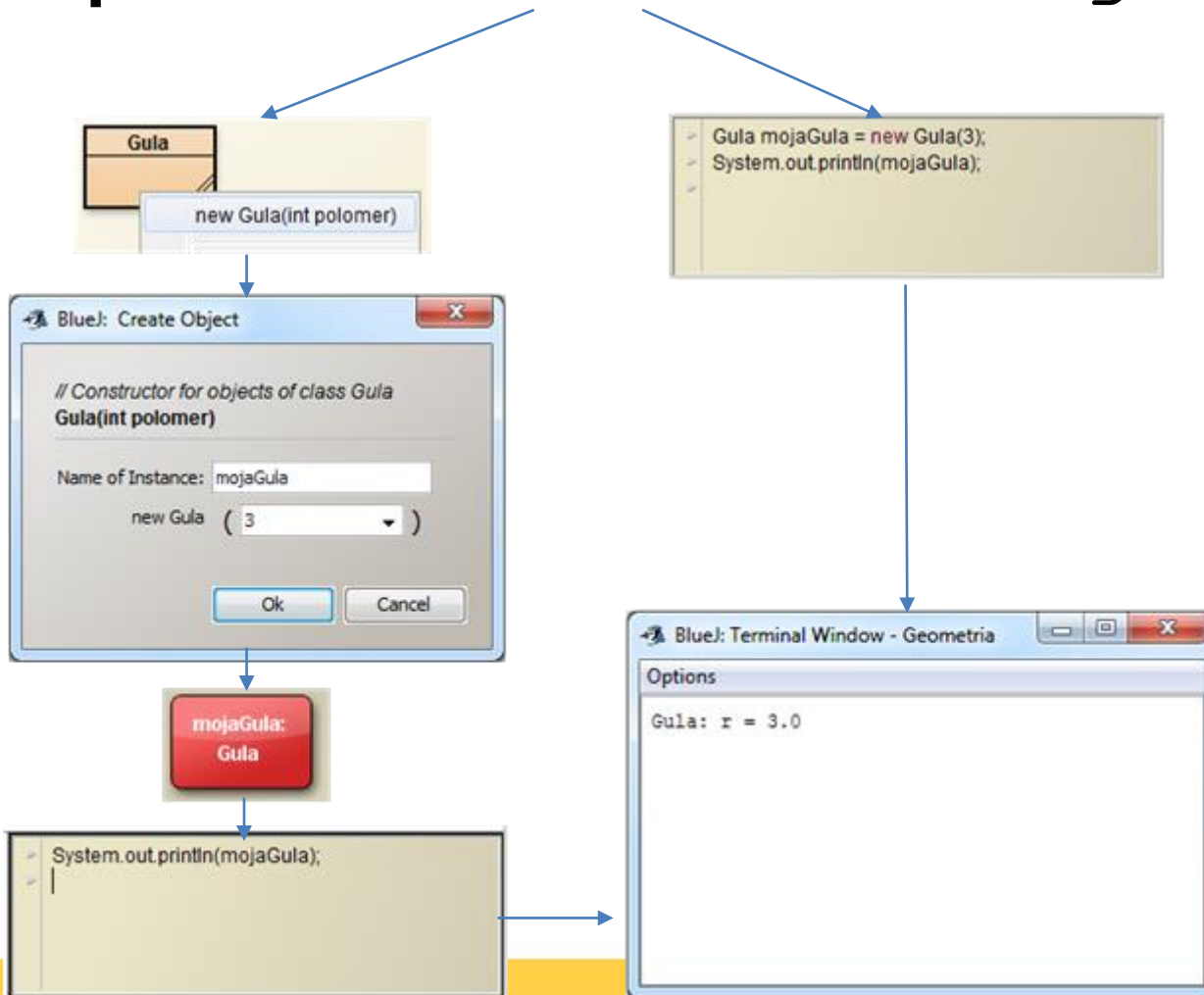
- Definujte metódu `toString` pre Vaše teleso.

```
public String toString() {  
    return "Gula: r = " + this.polomer;  
}
```

- Vytvorte inštanciu Vášho telesa a pošlite mu správu `toString`

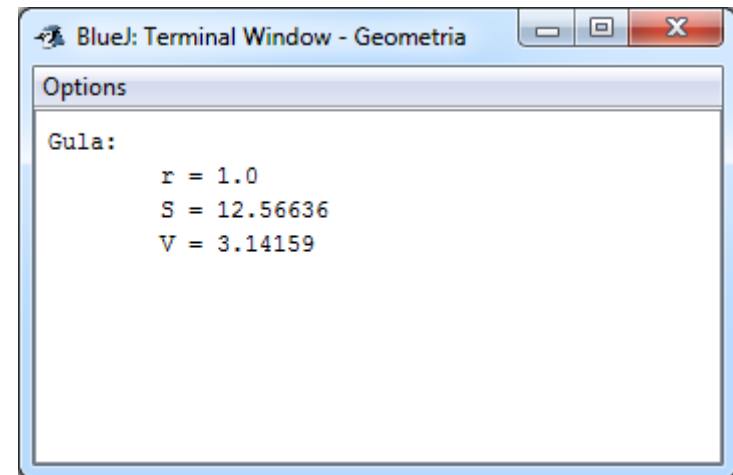


Vypísanie informácií o telese pomocou toString



Metóda toString

- Reťazec je možné rozšíriť o znaky nového riadka, tabulátora atď:
 - `\n` – nový riadok.
 - `\t` – tabulátor.
- Upravte metódu `toString` tak, aby vracala naformátovaný reťazec o všetkých informáciách o telese.
- Aký je rozdiel medzi metódami `toString` a `vypisSa`?



```
Options
Gula:
    r = 1.0
    S = 12.56636
    V = 3.14159
```

Tvorba tried

DRUHÁ ČASŤ

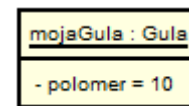
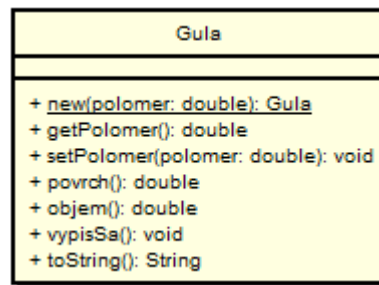
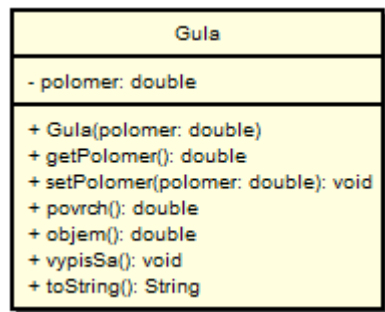


Cieľ cvičenia

- Základy práce s UML.FRI 2.
- Vytvorenie UML diagramu triedy reprezentujúcej teleso.
- Vytvorenie jednoduchkej triedy na základe UML.

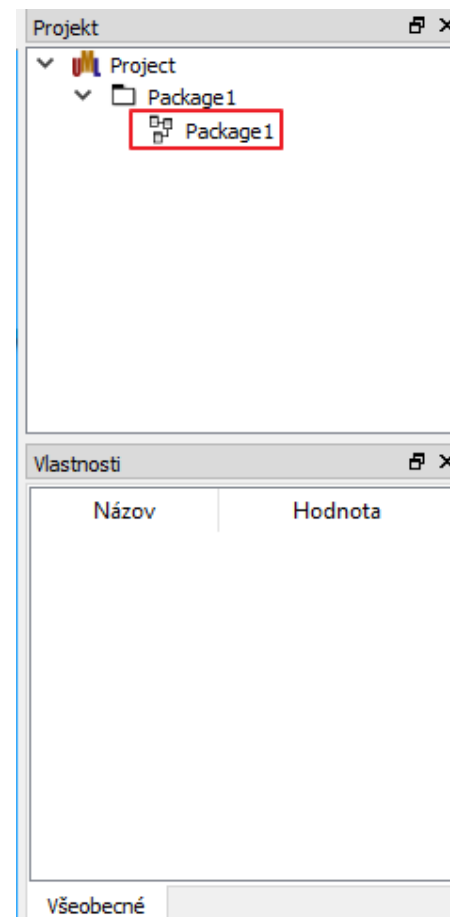
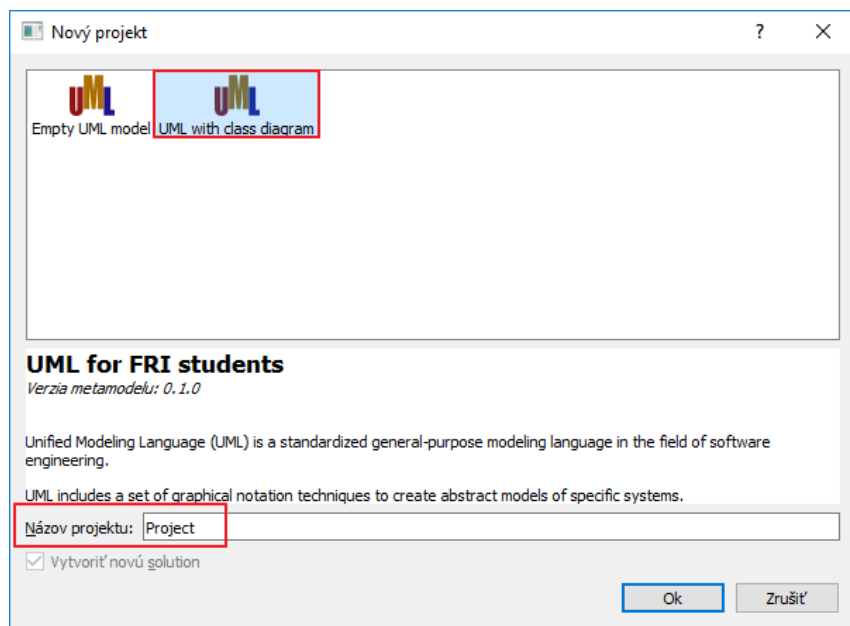
UML diagrams

- Aké diagramy sú zobrazené na nasledujúcich obrázkoch?





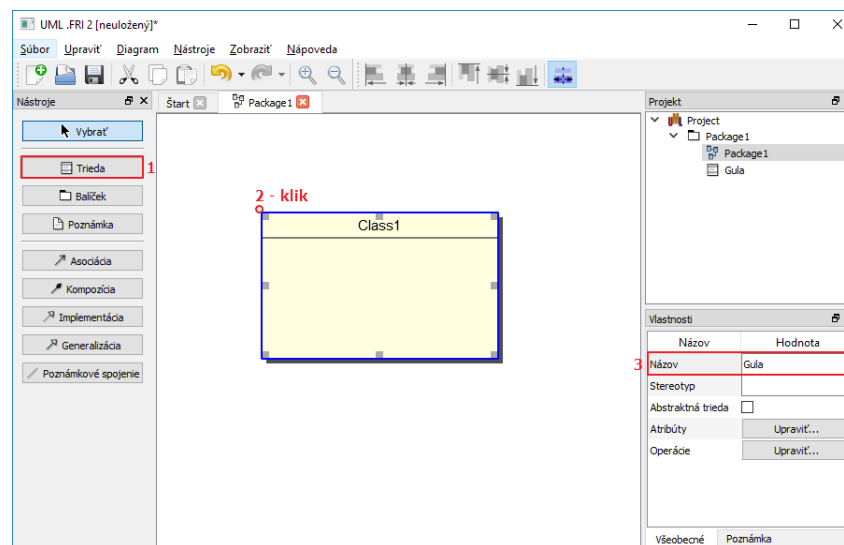
Vytvorenie nového projektu





Pridanie tried

1. Z menu vľavo vyberte položku trieda.
 2. Kliknite ľavým myšítkom do plátna.
 3. V panely vlastností vpravo zmeňte názov triedy.
- Pridajte triedu reprezentujúcu triedu Vášho telesa.



Editácia atribútov

- Dvojklik na triedu.
- Vyberte záložku Atribúty.
- Kliknite na Nový.
- Vypíšte meno a typ atribútu.
- Kliknite na Uložiť!
- Pre pridanie ďalšieho atribútu je potrebné opäť kliknúť na Nový!
- Nakoniec kliknite na Ok.
- Pridajte všetky atribúty z Vašej triedy.

Editácia metód

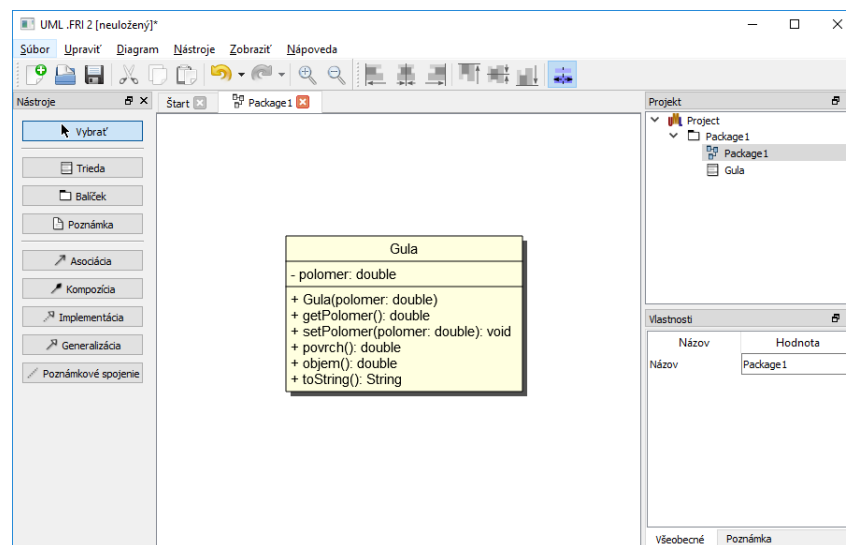
- Dvojklik na triedu.
- Vyberte záložku Operácie.
- Kliknite na Nový.
- Vypíšte meno a návratový typ metódy.
- Pre editáciu parametrov kliknite na tlačidlo Upraviť...(postup editácie je rovnaký ako pri atribútoch).
- Kliknite na Uložiť!
- Pre pridanie ďalšej metódy je potrebné opäť kliknúť na Nový!
- Nakoniec kliknite na Ok.
- Pridajte všetky metódy z Vašej triedy.

Názov	Typ návratovej hodnoty	Parametre	Stereotyp	Viditeľnosť



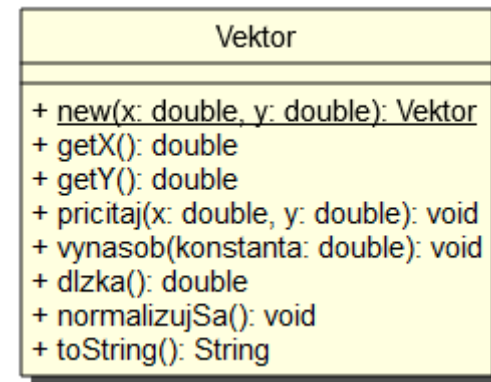
Výsledný UML diagram tried

- Vnútrotný pohľad na triedu.
- Niekedy nežiadúce – zadá sa len vonkajší pohľad, vnútrotný je možné jednoducho vydedukovať.



Vytvorenie triedy na základe UML

- Vytvorte triedu na základe uvedeného UML diagramu.
- Najskôr vytvorte triedu a definície metód.
- Aké atribúty bude mať trieda?
- Implementujte telá metód.



Upozornenie

- Tieto študijné materiály sú určené výhradne pre študentov predmetu 5BI137 Informatika 1 na Fakulte riadenia a informatiky Žilinskej univerzity v Žiline.
- Reprodukovanie, šírenie (i častí) materiálov bez písomného súhlasu autora nie je dovolené.

Ing. Michal Varga, PhD.
Katedra informatiky
Fakulta riadenia a informatiky
Žilinská univerzita v Žiline
Michal.Varga@fri.uniza.sk

