PB162 Programovanie v jazyku Java I

1. Úvod

Marek Šabo

16. septembra 2018

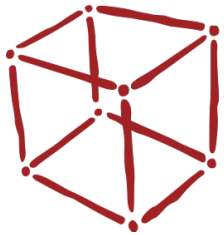
Organizačné pokyny

- základné informácie:
  - osnova v ISe
  - wiki
  - iteračný projekt – každý týdeň nové zadanie
- na riešenie nejasností používajte diskusné fórum
- cibuľová výuka – čo sa naučíme jeden týždeň použijeme v tom ďalšom
- predmet je časovo náročný, ale látka je OK

UNIXové okienko

- Pre Windows užívateľov: nainštalujte si Cygwin alebo Git Bash
- príkazy na zopakovanie: man, cd, ls, grep, ssh, vim/gedit, ...
- vedieť čo sú to prepínače, napr. `-h --help`

Integrated development environment – IDE

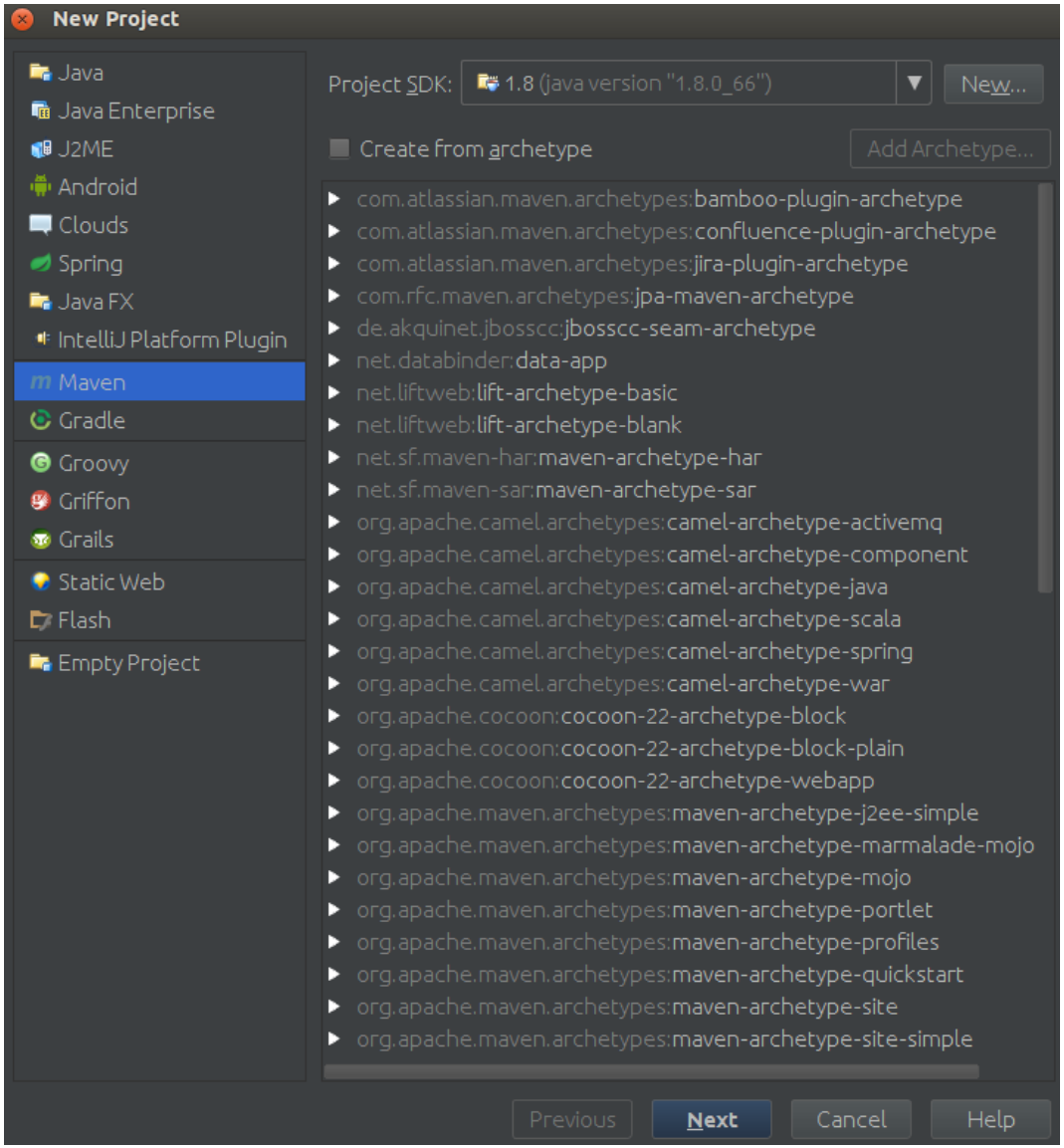


NetBeans

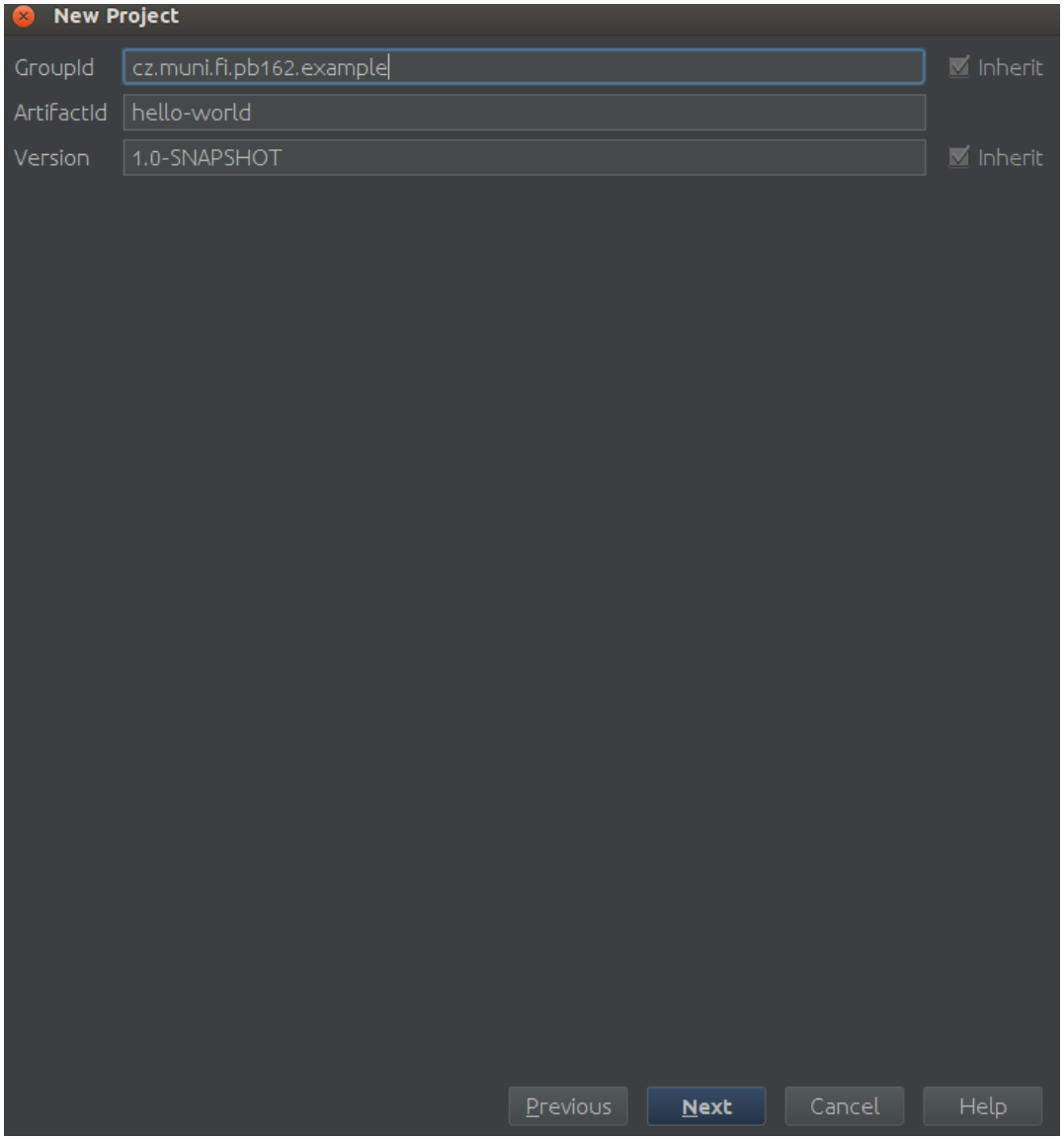
IntelliJIDEA

eclipse

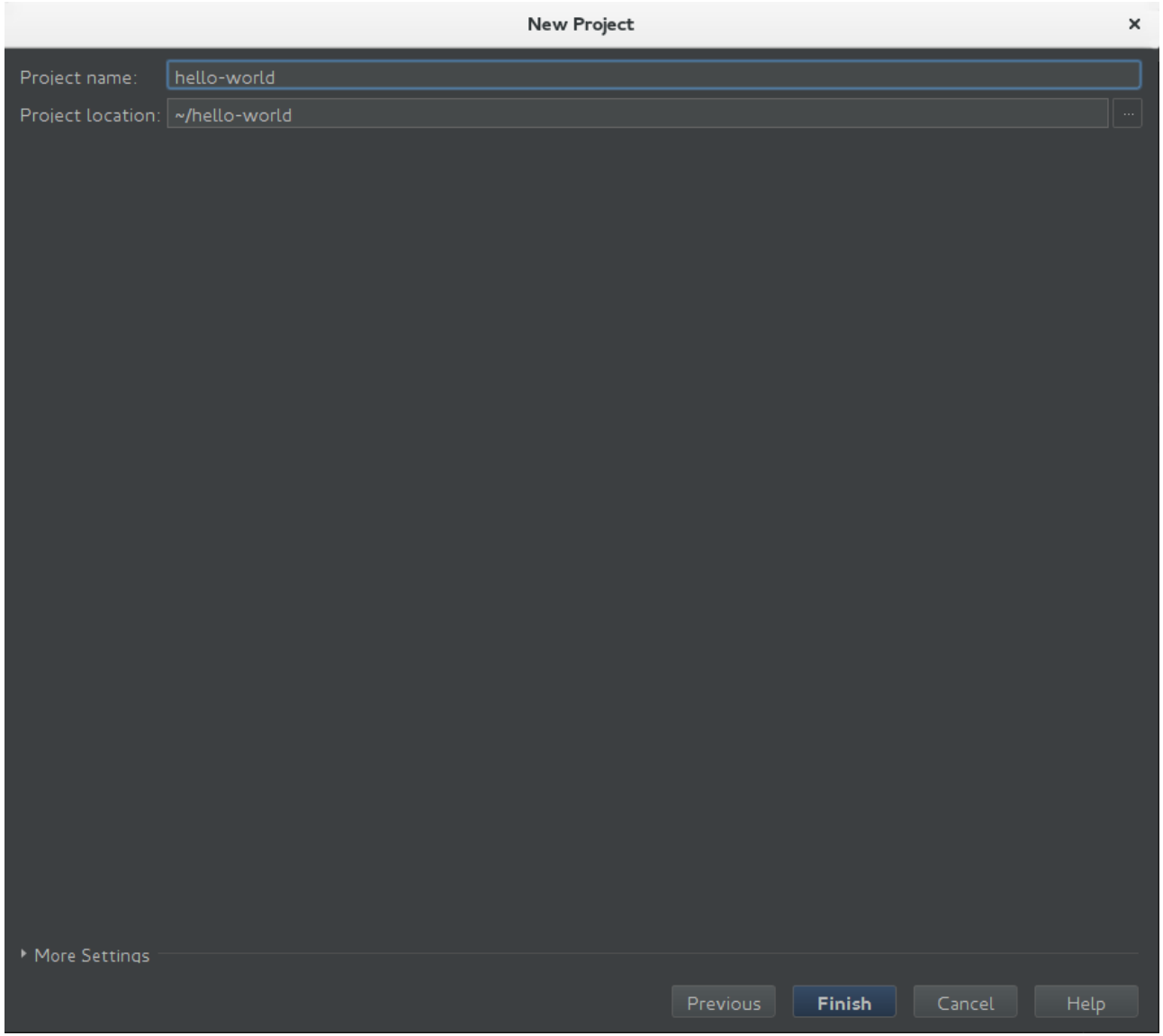
Demo I


New Project
Java
Java Enterprise
J2ME
Android
Clouds
Spring
Java FX
IntelliJ Platform Plugin
Maven
Gradle
Groovy
Griffon
Grails
Static Web
Flash
Empty Project
Project SDK:
1.8 (java version "1.8.0_66")
New...
Create from archetype
Add Archetype...
com.atlassian.maven.archetypes:bamboo-plugin-archetype
com.atlassian.maven.archetypes:confluence-plugin-archetype
com.atlassian.maven.archetypes:jira-plugin-archetype
com.rfc.maven.archetypes:jpa-maven-archetype
de.akquinet.jbosscc:jbosscc-seam-archetype
net.databinder:data-app
net.liftweb:lift-archetype-basic
net.liftweb:lift-archetype-blank
net.sf.maven-har:maven-archetype-har
net.sf.maven-sar:maven-archetype-sar
org.apache.camel.archetypes:camel-archetype-activemq
org.apache.camel.archetypes:camel-archetype-component
org.apache.camel.archetypes:camel-archetype-java
org.apache.camel.archetypes:camel-archetype-scala
org.apache.camel.archetypes:camel-archetype-spring
org.apache.camel.archetypes:camel-archetype-war
org.apache.cocoon:cocoon-22-archetype-block
org.apache.cocoon:cocoon-22-archetype-block-plain
org.apache.cocoon:cocoon-22-archetype-webapp
org.apache.maven.archetypes:maven-archetype-j2ee-simple
org.apache.maven.archetypes:maven-archetype-marmalade-mojo
org.apache.maven.archetypes:maven-archetype-mojo
org.apache.maven.archetypes:maven-archetype-portlet
org.apache.maven.archetypes:maven-archetype-profiles
org.apache.maven.archetypes:maven-archetype-quickstart
org.apache.maven.archetypes:maven-archetype-site
org.apache.maven.archetypes:maven-archetype-site-simple
Previous
Next
Cancel
Help

Demo II


New Project
GroupId
cz.muni.fi.pb162.example
Inherit
ArtifactId
hello-world
Version
1.0-SNAPSHOT
Inherit
Previous
Next
Cancel
Help

Demo III


New Project
Project name:
hello-world
Project location:
~/hello-world
More Settings
Previous
Finish
Cancel
Help

Nastavenia Intellij

Nainštalujte si checkstyle plugin

- Ctrl + Shift + A, napíšte 'plugins' a odentrujte
- v ľavom hornom rohu napíšte do vyhľadávania 'checkstyle', ak nič nenašlo kliknite na 'Browse repositories...'
- dajte inštalovať Checkstyle-IDEA

Zmeňte defaultný text pri vytvorení triedy

- Ctrl + Shift + A, napíšte a spustite 'Code template'
- zvoľte 'File Header'
- upravte a uložte

```
/**
 * TODO: create javadoc
 * @author <your name>
 */
```

Poznámka

Zobraté z wiki: *https://gitlab.fi.muni.cz/pb162/pb162-course-info/wikis/working-with-ide*

Úvod do OOP

Motivácia: chceme rozumne ukladať štruktúry, aby sa s nimi dalo lepšie pracovať.

Chceme vytvoriť/popísať objekt pes. Čo všetko by mal obsahovať?

Úvod do OOP

Motivácia: chceme rozumne ukladať štruktúry, aby sa s nimi dalo lepšie pracovať.

Chceme vytvoriť/popísať objekt pes. Čo všetko by mal obsahovať?

to, čo pes má: hlavu, oči, 4 nohy, meno, ...

to, čo pes robí: štekanie, behanie, vrtenie chvostom, ...

Úvod do OOP

Motivácia: chceme rozumne ukladať štruktúry, aby sa s nimi dalo lepšie pracovať.

Chceme vytvoriť/popísať objekt pes. Čo všetko by mal obsahovať?

to, čo pes má: hlavu, oči, 4 nohy, meno, . . . **atribúty**

to, čo pes robí: štekanie, behanie, vrtenie chvostom, . . . **metódy**

Pseudopes – intuícia

```
class Dog {

 String name;
 int age;

 String welcomeMaster() {
   return "Woof I am " + name;
 }

 String barkOthers() {
   return "Wrr HAF HAF";
 }
}
```

Pseudopes – intuícia

```
class Dog {

 String name;
 int age;

 String welcomeMaster() {
   return "Woof I am " + name;
 }

 String barkOthers() {
   return "Wrr HAF HAF";
 }
}
```

Čo ak

- chceme overiť, aby sa vek vždy nastavil na nezáporné číslo?
- nechceme zverejniť meno nášho psa?
- chceme obmedziť viditeľnosť metód?

Pseudopes – pokračovanie

```
public class Dog {

  private String name;
  private int age;

  public String welcomeMaster() {
    return "Woof I am " + name;
  }

  public String barkOthers() {
    return "Wrr HAF HAF";
  }

  public void setAge(int newAge) {
    if newAge > 0
      age = newAge;
  }

  public int getAge() { return age; }
}
```

Trieda je verejná – každý môže vyrábať psov.

Atribúty sú privátne, iba metódy v triede môžu s nimi pracovať.

Metódy na prácu s atribútmi nazývame gettery a settery.

Ako však vytvoríme konkrétneho psa?

```
<viditeľnosť> <návratový typ> <názov>(<parametre>)
```

Konštruktory

1. Konštruktor bez parametrov (vytvorí sa sám, ak neexistuje žiadny)

```
public Dog() { } // v triede Dog


Dog staryDunco = new Dog(); // v inej triede
staryDunco.setName("Dunco");
staryDunco.setAge(13);
```

2. Konštruktor s parametrami

```
public Dog(String ourName, int ourAge) {
  name = ourName;
  age = ourAge; // privatne atributy su name a age
}

Dog staryDunco = new Dog("Dunco", 13);
```

Premenná staryDunco sa nazýva **inštanciou triedy** Dog.

Vytváranie primitívnych typov vs. vytváranie objektov

Primitívne typy

- *int, boolean, char, double, . . .*
- premenná môže byť vytvorená "ihneď"
- každý typ má defaultnú hodnotu: int má 0, double 0.0, atď.

Objekty

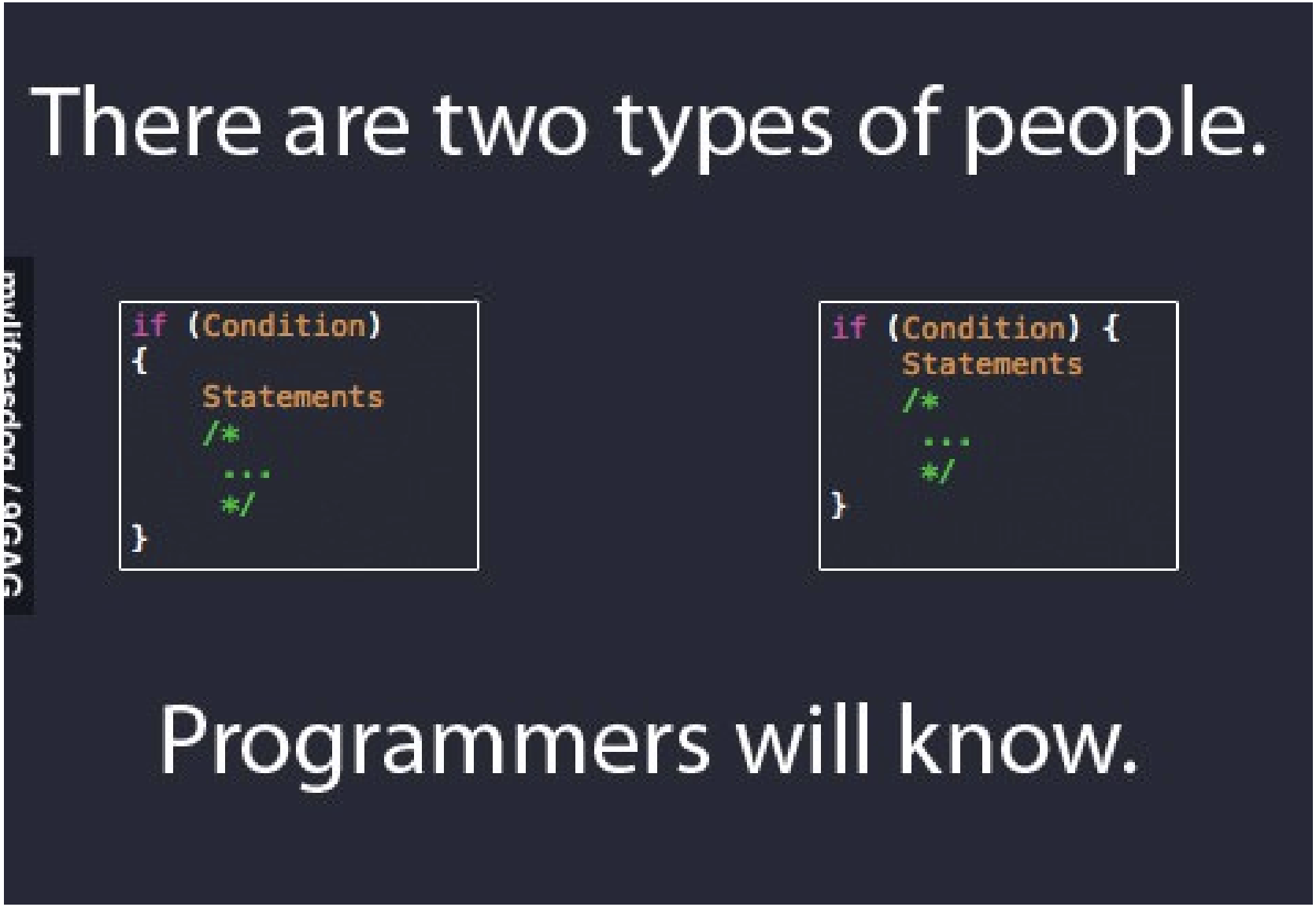
- zvyčajne si vytvárame vlastné – *Dog, Car, Wheel, Main, Math, . . .*
- premenná musí byť najprv **skonštruovaná**, napr. auto si musí najprv vytvoriť kolesá
- objektová premenná má defaultne hodnotu `null`
- premenné sú prakticky ukazatele, `null` hovorí "ukazuj na nič"

Poznámka

Objekt začína na rozdiel od primitívnych typov veľkým písmenom.

Dog vs. int

Konvencie v Jave



C# vs. Java

Balíčky alebo Čo vás napadne pri slove jaguár?

Balíčky alebo Čo vás napadne pri slove jaguár?

Balíčky alebo Čo vás napadne pri slove jaguár?

Balíčky alebo Čo vás napadne pri slove jaguár?

Jaguar

Jaguar

Balíčky alebo Čo vás napadne pri slove jaguár?

animal.wild.Jaguar

vehicle.car.Jaguar

Balíčky alebo Čo vás napadne pri slove jaguár?

animal.wild.Jaguar

vehicle.car.Jaguar

Aj Android takýmto spôsobom odlišuje aplikácie s rovnakým názvom.

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

  Na obmedzenie viditeľnosti (nechceme aby hocikto nastavil daný atribút).

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

  Na obmedzenie viditeľnosti (nechceme aby hocikto nastavil daný atribút).
- Môžu mať metódy viditeľnosť typu `private`?

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

  Na obmedzenie viditeľnosti (nechceme aby hocikto nastavil daný atribút).
- Môžu mať metódy viditeľnosť typu `private`?

  Môžu, ale nie je bežné.

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

  Na obmedzenie viditeľnosti (nechceme aby hocikto nastavil daný atribút).
- Môžu mať metódy viditeľnosť typu `private`?

  Môžu, ale nie je bežné.
- Čo ak začnem používať inštanciu objektu bez jej vytvorenia?

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

  Na obmedzenie viditeľnosti (nechceme aby hocikto nastavil daný atribút).
- Môžu mať metódy viditeľnosť typu `private`?

  Môžu, ale nie je bežné.
- Čo ak začnem používať inštanciu objektu bez jej vytvorenia?

  Nastane výnimočná situácia, ktorá spôsobí pád programu.

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

  Na obmedzenie viditeľnosti (nechceme aby hocikto nastavil daný atribút).
- Môžu mať metódy viditeľnosť typu `private`?

  Môžu, ale nie je bežné.
- Čo ak začnem používať inštanciu objektu bez jej vytvorenia?

  Nastane výnimočná situácia, ktorá spôsobí pád programu.
- String sa píše veľkým písmenom, znamená to, že je to objekt?

Otázky na zamyslenie

- Aké sú ďalšie výhody používania balíčkov?

  Triedy majú svoju hierarchiu, štruktúru.
- Prečo je dobré mať privátne atribúty?

  Na pridanie kódu pri nastavovaní (overenie validného vstupu).

  Na obmedzenie viditeľnosti (nechceme aby hocikto nastavil daný atribút).
- Môžu mať metódy viditeľnosť typu `private`?

  Môžu, ale nie je bežné.
- Čo ak začnem používať inštanciu objektu bez jej vytvorenia?

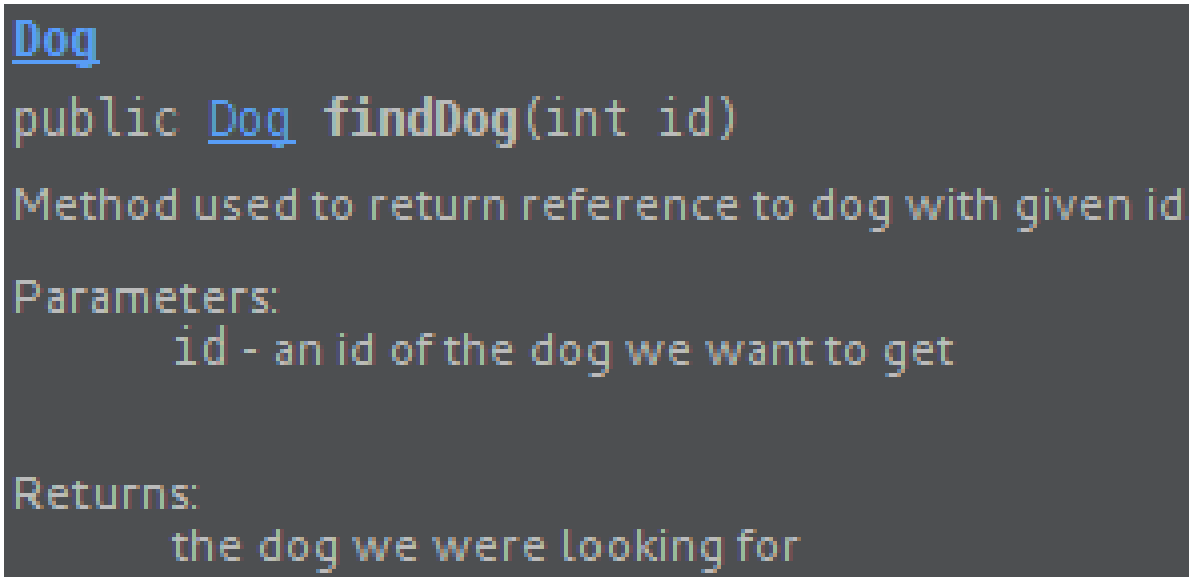
  Nastane výnimočná situácia, ktorá spôsobí pád programu.
- String sa píše veľkým písmenom, znamená to, že je to objekt?

  Áno, ale môžme ho vytvárať aj inak ako použitím `new String("Dunco")`, stačí `"Dunco"`.

Komentáre sú užitočné

Javadoc

```java
/**
 *
 * Method used to return reference to dog with given id.
 *
 * @param   id    an id of the dog we want to get
 * @return        the dog we were looking for
 */
public Dog findDog(int id) {
        return dogs[id];
}
```



Na doma

Zistite si, čo je to `Git`. Skúste sa s ním pohrať na try.github.io.
Pre lepšiu vizualizáciu a pokročilejšiu prácu je tu learngitbranching.js.org.

Na ďalšom cvičení si prejdeme návod na nastavenie vášho súkromného repozitára. Spravte si krok 0 a 0.1.