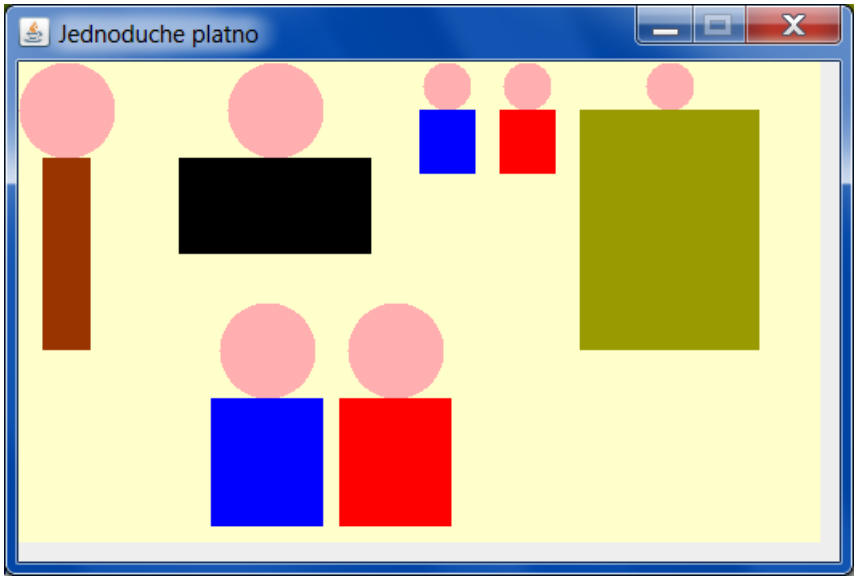
1. Domácí úloha 02

Základní informace:

- **Účel:** procvičení práce s konstruktory a getry/setry třídy `Osoba`; dále slouží pro pokročilejší orientaci ve způsobu testů; kontrola dodržení zadání pomocí Duck-testů
- **Kostra:** `02_VytvareniTridyOsoba.zip`
- **Odevzdávaný soubor:** `Osoba.java`

Zadání:

- připravte třídu `Osoba`, která dokáže vykreslit následující obrázky:



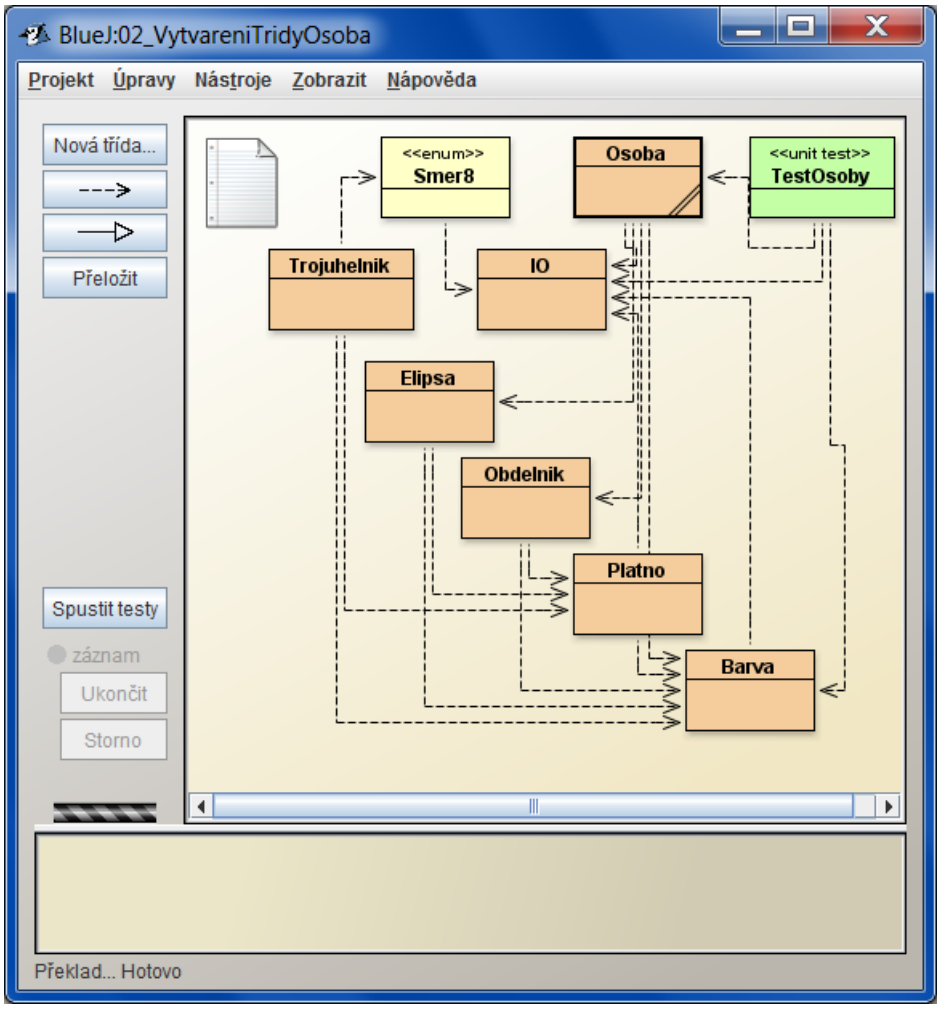
- třídu `Osoba` vytvářejte postupně dle dále uvedených pokynů
  - zatím nepište žádnou Javadoc dokumentaci, s výjimkou samého začátku souboru, kde uvedete příslušně upravené

```
 * @author    Pavel Herout
 * @version   2.00.000
```

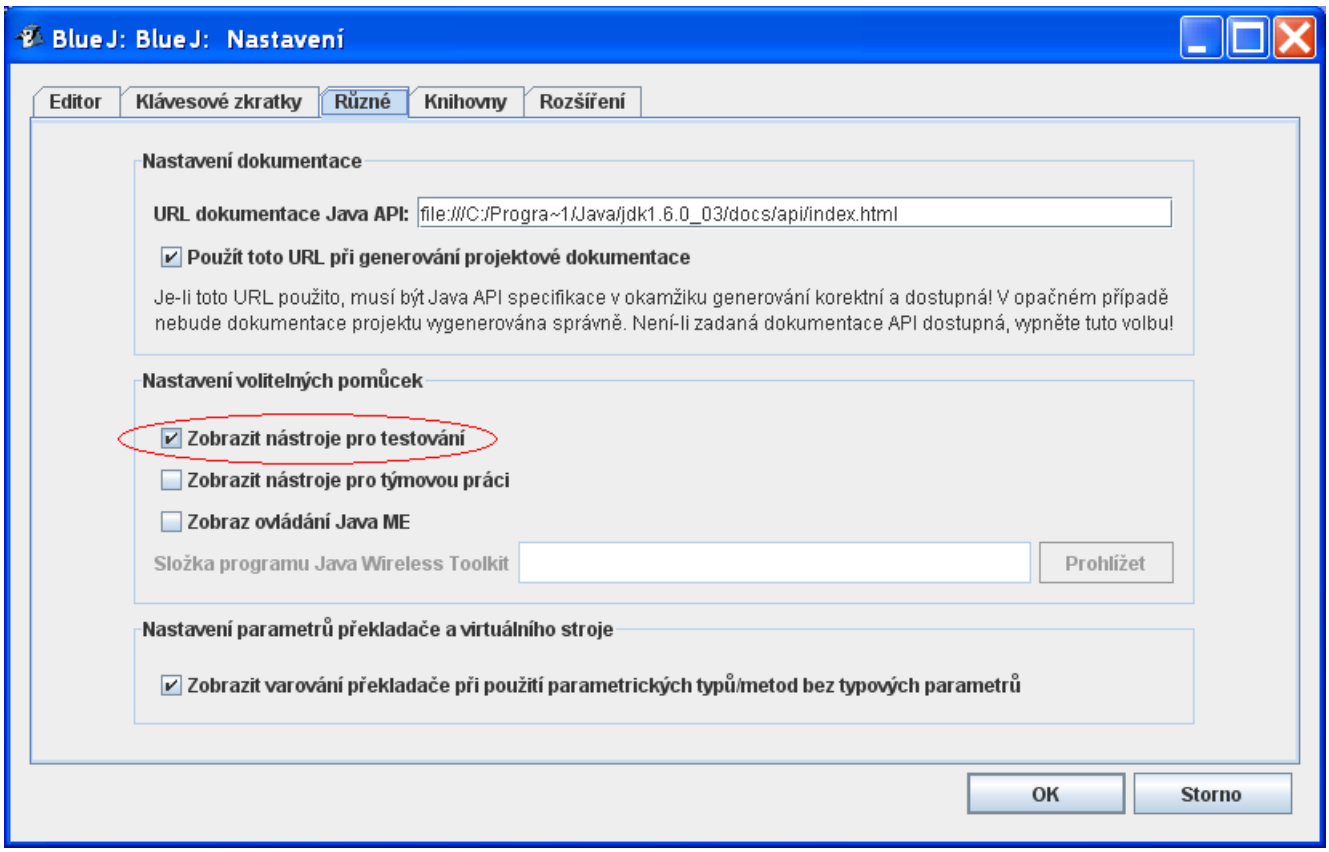
  verzi nastavte podle odhadu počtu potřebných pokusů
- do Portálu odevzdáte soubor `Osoba.java`
- třídu `TestOsoby` budete využívat, ale nebudete (nemusíte) ji nijak měnit - její implementace vám při konstrukci třídy `Osoba` příliš nepomůže

Postup řešení:

- stáhněte si soubor `02_VytvareniTridyOsoba.zip` a rozbalte jej
- v BlueJ otevřete projekt `02_VytvareniTridyOsoba` a pomocí tlačítka *Nová třída / Standardní třída* založte třídu `Osoba`



- pomocí *Nástroje* > *Nastavení* > *Různé* vyberte volbu *Zobrazit nástroje pro testování*



- pokud nebude výslovně uvedeno jinak, týkají se všechny další instrukce obsahu třídy `Osoba`

- můžete se inspirovat třídou `Strom` z přednášek [OOP-31] a dále

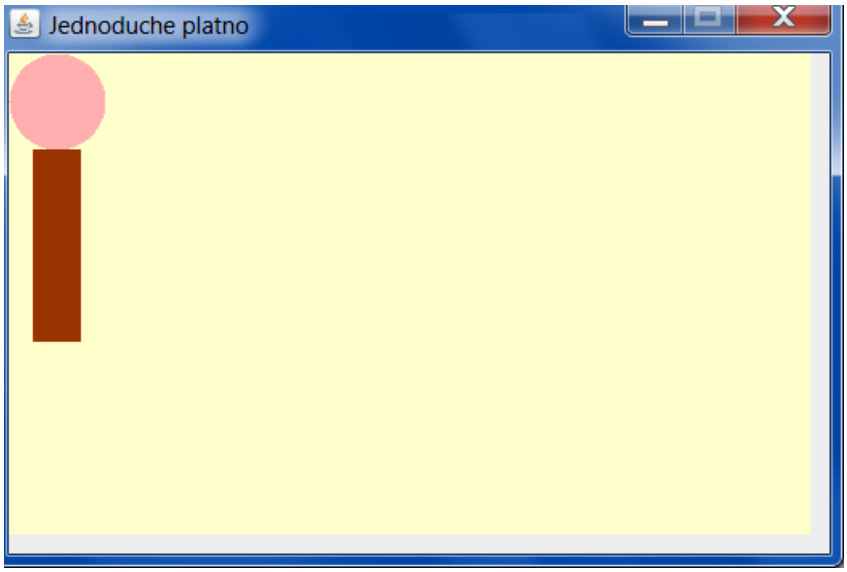
- připravte statické konstanty
  - `BARVA_HLAVY = Barva.RUZOVA`
  - `IMPL_BARVA_TELA = Barva.SEDA`
  - `IMPL_VELIKOST_HLAVY = 60`
  - `POMER_HLAVA_TELO = 6.0/8.0`
  - `POMER_TELO_VYS_SIR = 8.0/7.0`
- připravte **konstantní** atributy instancí příslušného datového typu (shodně s DU-01) s přístupovým právem `private`
  - `hlava`
  - `telo`
  - oba tyto atributy neinicializujte -- budou nastaveny až v konstruktoru
- přidejte metody `getHlava()`, `getTelo()`, které jsou používány hlavně pro účely testování
- žádné další pomocné atributy nepoužívejte (např. souřadnice `x`, `y`, `vyskaOsoby` atp.)
  - pokud je použijete, odhalí to na závěr Duck-testy (viz zcela dole) a pak budete nuceni složitě opravovat svůj kód!!!
- připravte nejsložitější konstruktor (neměl by přesáhnout délku 10 řádek), který:

```
public Osoba(int x, int y, int velikostHlavy,
             double pomerHlavaTelo, double pomerTelo,
             Barva barvaTela) {
```

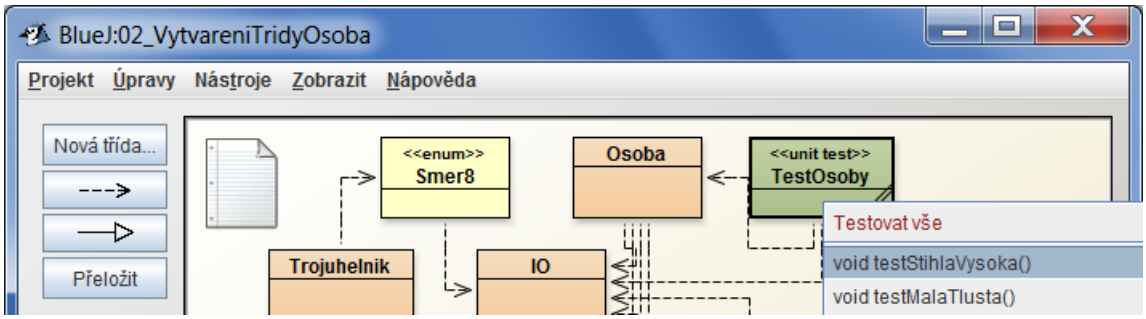
  - vytvoří na definovaných souřadnicích (levý horní roh celkového ohraničujícího obdélníka) instanci o rozměrech odvozených od velikosti hlavy se zadanou barvou těla
    - `pomerHlavaTelo` znamená poměr hlavy ku výšce těla, např. u štíhlé osoby na obrázku je to 1 / 2 (konkrétně 60 ku 120)
    - `pomerTelo` znamená poměr výšky těla ku šířce těla, např. u štíhlé osoby na obrázku je to 4 / 1 (konkrétně 120 ku 30)
    - to znamená, že z rozměru hlavy lze snadno vypočíst oba rozměry těla v daných poměrech
  - tělo a hlava jsou vystředěny podle svislé osy -- viz první obrázek štíhlé (hnědá) a tlusté (černá) osoby
    - to znamená, že musíte zjistit, zda je hlava širší než tělo či naopak, a podle výsledku posunout užší část
  - nezapomeňte provádět všechny aritmetické operace (zejména dělení) v typu `double`
  - budete potřebovat přetypování na typ `int` [PPA1-26]

- dále se vám může hodit absolutní hodnota čísla `Math.abs()` [PPA1-42]

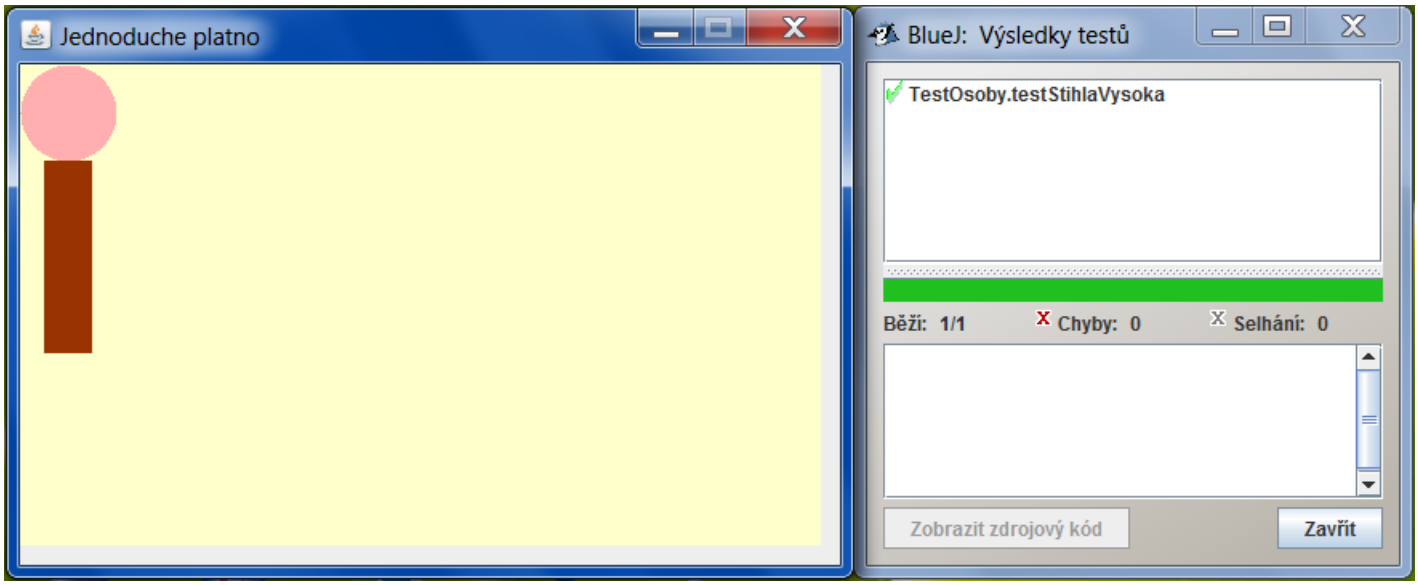
- příkazem z místní nabídky třídy `TestOsoby` *Testovací přípravek* > *Zásobník Odkazů* zobrazte instanci `osStihlaVysoka`:



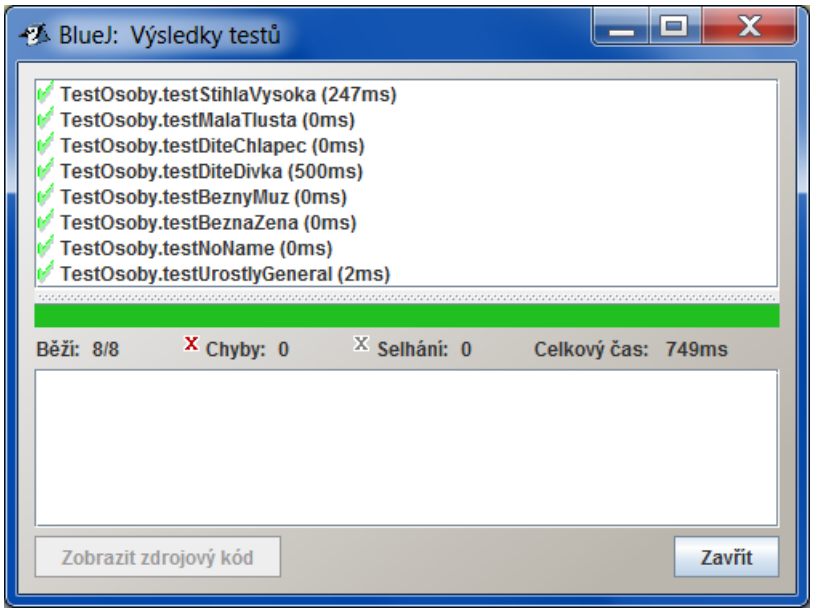
- upravujte zdrojový kód konstruktoru tak dlouho, dokud nedostanete tento obrázek
- dále přidejte metody `getX()`, `getY()`, `getSirka()`, `getVyska()`
  - tyto metody budou potřebné údaje vždy znovu vypočítávat z hodnot poskytovaných instancemi `hlava` a `telo`, které byly nastaveny v konstruktoru
  - pro výpočet se vám budou hodit metody `Math.min()` a `Math.max()` [PPA1-42]
  - uvědomte si, že hlava může být užší nebo širší než tělo
  - pozice [x, y] je levý horní roh ohraničujícího obdélníka celé osoby
- přidejte statickou proměnnou `pocet` a instanční konstanty `PORADI` a `NAZEV` [OOP-35]
- přidejte metodu `getNazev()`
- zvolte *Zobrazit / Ukázat výsledky testů*
- z místní nabídky třídy `TestOsoby` spusťte *testStihlaVysoka()*



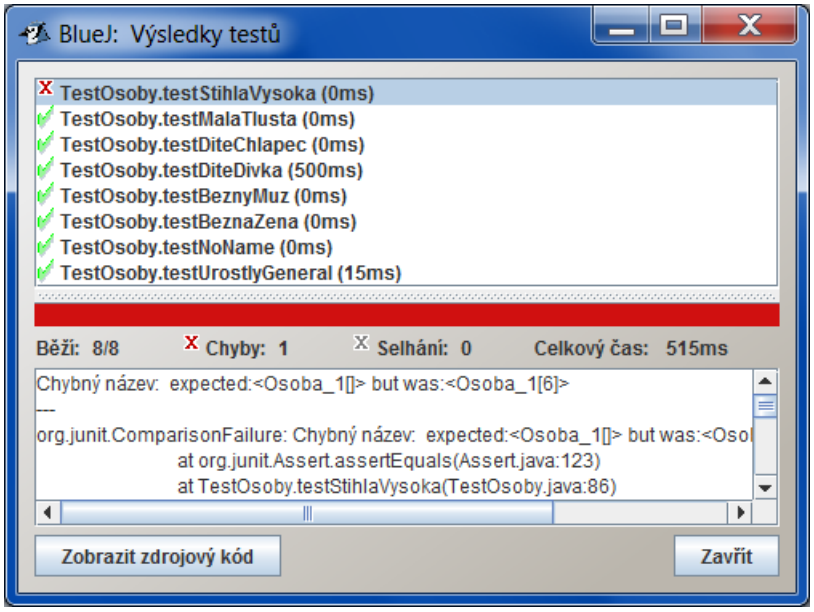
- upravujte výše zmíněné metody tak dlouho, dokud nedostanete následující výsledek
  - postup při výskytu chyby viz v zadání DU-01



- ve třídě `TestOsoby` odkomentujte metodu `testMalaTlusta()`
  - z místní nabídky třídy `TestOsoby` spusťte *testMalaTlusta()* a opět odstraňte případné problémy
- přidejte konstruktor `public Osoba(int x, int y, int velikostHlavy, Barva barvaTela)`, který využívá volání předchozího konstruktoru pomocí `this`
  - chybějící skutečné parametry `pomerHlavaTelo` a `pomerTelo` nahradí odpovídajícími symbolickými konstantami
- přidejte metody `getBarvaTela()` a `setBarvaTela()`
- ve třídě `TestOsoby` odkomentujte metody `testDiteChlapec()` a `testDiteDivka()`
- spusťte testy *testDiteChlapec()* a *testDiteDivka()*
  - u *testDiteDivka()* je časová prodleva 500 ms mezi přebarveními těla - nebuďte tím překvapeni
- přidejte konstruktor `public Osoba(int x, int y, Barva barvaTela)`, který využívá volání předchozího konstruktoru pomocí `this`
  - odkomentujte příslušné metody a spusťte testy *testBeznyMuz()* a *testBeznaZena()*
- přidejte konstruktor `public Osoba()`, který využívá volání předchozího konstruktoru pomocí `this` a vytvoří osobu na souřadnicích [0, 0] implicitních barev a implicitních velikostí
  - odkomentujte příslušnou metodu a spusťte *testNoName()* - pozor - na plátně se částečně překryje štíhlá osoba
- posledním testem je *testUrostlyGeneral()*, se kterým by již neměly být žádné potíže, protože netestuje nic dosud neotestovaného
- na závěr můžete tlačítkem *Spustit testy* (nebo *Testovat vše* z místní nabídky) spustit všechny připravené testy
  - před každým testem se mění plátno, takže na něm uvidíte vždy jen dvě osoby
  - výsledek by měl být



- pokud se vám objeví (po rozkliknutí chyby)



- je to způsobeno tím, že *testStihlaVysoka()* je závislý na pořadí (není úplně vhodný)
- v tomto případě restartujte virtuální stroj a spusťte testy znovu - další běh testů by měl být v pořádku

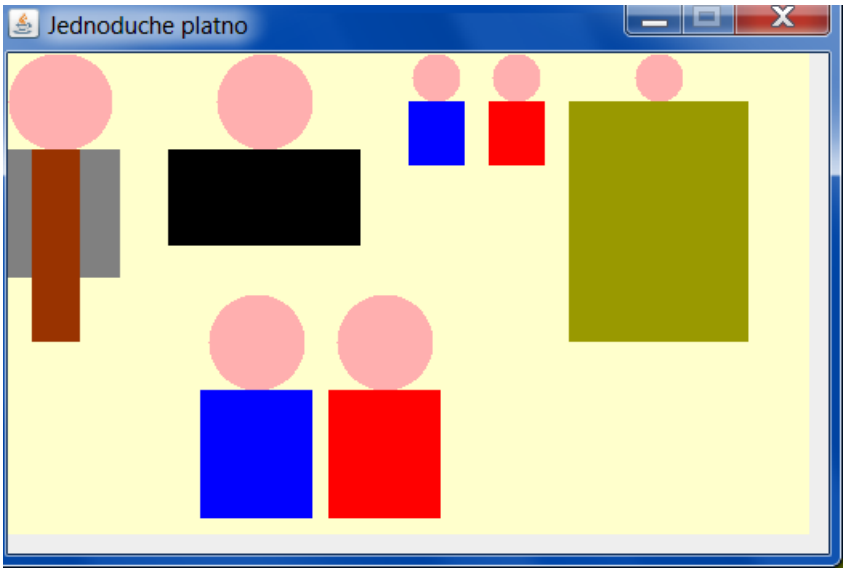
Varování

Testy jsou sice velmi podrobné, ale z principu nemohou nalézt všechny potenciální problémy. Porovnejte proto důsledně výsledný obrázek na plátně se vzorovým obrázkem v tomto zadání. Pokud si nebudou odpovídat, opravujte třídu `Osoba` tak dlouho, dokud nedojde ke shodě. Odevzdáte-li třídu `Osoba` s chybou, dostanete se do problémů v dalších DÚ, kdy bude tato třída dále používána.

Přístup: „Teď to prošlo a pak se uvidí.“ je spolehlivou cestou do problémů v budoucnu.

Poznámka

Po provedení všech testů uvidíte obrázek, ve kterém je zobrazena i osoba `NoName`. Protože se překrývá s osobou `StihlaVysoka`, nebyla na úvodním obrázku uvedena. Měli byste tedy vidět:



Může se stát, že šedá osoba překrývá hnědou osobu. Je to způsobeno nedokonalostí testů a tento případ neřešte.

- hotovou třídu `Osoba` otestujte pomocí Duck-testů
  - návod viz v samostatném souboru
  - pokud Duck-testy odhalí problém, nedodrželi jste zadání a je nutné příslušnou část kódu opravit
  - kvalitu každé opravy ověříte tlačítkem *Spustit testy* -- viz výše
  - pokračujte tak dlouho, dokud Duck-testy neprojdou bez chyb
- finální verzi třídy `Osoba` budete odevzdávat