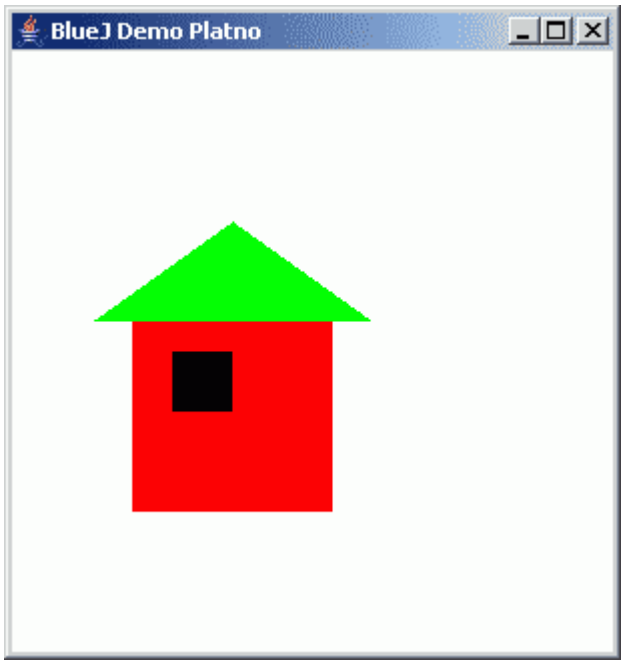
14.Projekt Obrázek

14.1. Základní popis, zadání úkolu

Pracujeme na projektu Obrázek, který je ke stažení na `http://java.vse.cz/`. Po otevření v BlueJ vytvoříme instanci třídy `Obrazek` a zavoláme metodu `kresli()`. Výsledkem je obrázek domečku:



Obrázek 14.1 Obrázek domečku z projektu Obrázek

Naším úkolem je:

- přidat do obrázku sluníčko,
- doplnit obsah metody, po jejímž zavolání projede na obrázku před domečkem auto,
- doplnit metodu pro východ slunce a metodu pro západ slunce.

Tento projekt má následující cíle:

- ukázat doplnění datového atributu (slunce) a příslušného kódu do projektu,
- na třídě `Auto` ukázat, že pro vytvoření instance (objektu) může být více konstruktorů,
- na doplnění auta do obrázku ukázat rozdíl mezi lokální proměnnou a datovým atributem,
- ukázat, jak doplnit obsah předpřipravené metody (`prujezdAuta()`),
- ukázat, jak doplnit celé metody (`vychodSlunce()` a `zapadSlunce()`),
- ukázat volání jiné metody stejné instance (volání `setCernoBily()` v metodě `zapadSlunce()`).

14.2. Struktura tříd

Projekt Obrázek se skládá ze šesti tříd:

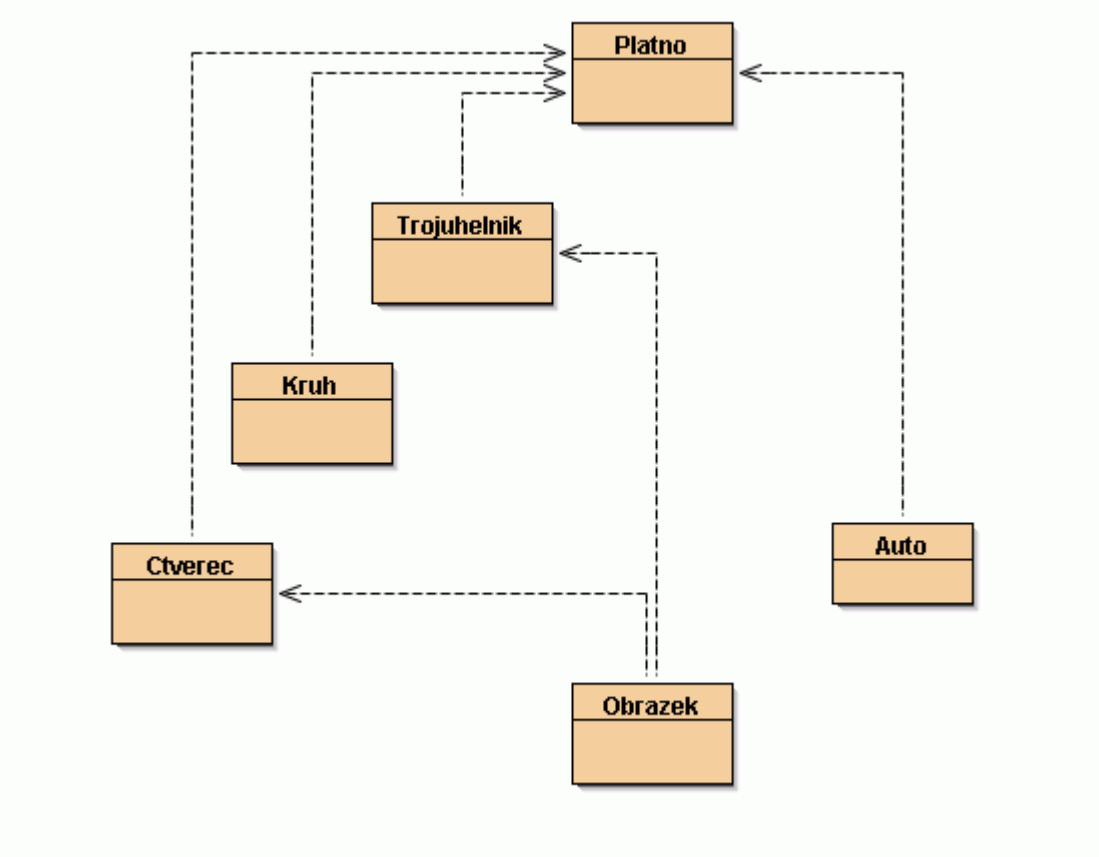
Platno – tato třída představuje prostor (plátno), na kterém se vykreslují jednotlivé tvary,

Kruh, **Ctverec**, **Trojuhelnik** – toto jsou jednotlivé tvary, které se kreslí na plátno, lze s nimi po plátně pohybovat, měnit jejich velikost a barvu,

Auto – obdoba předchozích tříd – na plátně se vykreslí auto,

Obrazek – představuje obrázek domečku – vykreslí dva čtverce (zeď a okno) a jeden trojúhelník na plátně.

Následuje diagram tříd (obrázek z BlueJ):



Obrázek 14.2 Diagram tříd projektu Obrázek, převzato z BlueJ

14.3. Popis metod jednotlivých tříd

Nejprve popíšeme metody tříd `Auto`, `Ctverec`, `Kruh` a `Trojuhelnik`. Třídu `Platno` podrobně popisovat nebudeme, třída zajišťuje plochu o velikosti 300x300 bodů pro vykreslování jednotlivých instancí tříd `Auto`, `Ctverec`, `Kruh`, `Trojuhelnik` a `Obrazek`. Jedna instance této třídy je společná pro všechny vykreslované tvary.

třída	konstruktor	popis konstruktoru
Ctverec	Ctverec()	vytvoří červený čtverec o straně 30 bodů na souřadnicích 60,50 a vykreslí ho na plátno
Kruh	Kruh()	vytvoří modrý kruh o průměru 30 bodů na souřadnicích 20,60 a vykreslí ho na plátno
Trojuhelnik	Trojuhelnik()	vytvoří zelený trojúhelník o šířce 40 a výšce 30 bodů na souřadnicích 50,15 a vykreslí ho na plátno
Auto	Auto()	vytvoří modré auto na souřadnicích 60,40 a vykreslí ho na plátno
	Auto(int xPozice,int yPozice)	vytvoří modré auto na zadaných souřadnicích

Tabulka 14.1 Přehled konstruktorů tříd projektu

metoda	popis metody
posunVpravo()	posune daný tvar o 20 bodů vpravo
posunVlevo()	posune daný tvar o 20 bodů vlevo
posunNahoru()	posune daný tvar o 20 bodů nahoru
posunDolu()	posune daný tvar o 20 bodů dolu
posunHorizontalne(int vzdalenost)	posune daný tvar o zadaný počet bodů (+ vpravo, − vlevo)
posunVertikalne(int vzdalenost)	posune vertikálně daný tvar o zadaný počet bodů (+ dolů, − nahoru)
pomaluPosunHorizontalne(int vzdalenost)	pomalé posunutí tvaru o zadaný počet bodů horizontálně
pomaluPosunVertikalne(int vzdalenost)	pomalé posunutí tvaru o zadaný počet bodů vertikálně
zmenVelikost(int novaVelikost)	změní velikost daného tvaru, tato metoda neexistuje u třídy Auto (plátno umí nakreslit jen jednu velikost auta)
zmenBarvu(String novaBarva)	změní barvu zadaného tvaru

Tabulka 14.2 Přehled metod tříd Kruh, Ctverec, Trojuhelnik a Auto

Vysvětlíme si následující část kódu s instancí třídy `Ctverec`:

```
        Ctverec zed;
        zed = new Ctverec();
        zed.posunVertikalne(80);
        zed.zmenVelikost(100);
```

Na prvním řádku je deklarace proměnné `zed` typu `Ctverec`, na druhém řádku se vytvoří instance třídy `Ctverec` (zavolá se konstruktor) a odkaz se uloží do proměnné `zed` – čtverec se nakreslí na plátno. Na třetím řádku se instanci čtverce pošle zpráva, aby se posunula vertikálně o 80 bodů doprava – u instance, na kterou odkazuje proměnná `zed` se zavolá metoda `posunVertikalne(80)`. Na čtvrtém řádku se zvětší velikost strany čtverce na velikost 100 bodů.

14.4. Kód třídy Obrazek

```
 1 /**
 2  * Tato třída reprezentuje jednoduchý obrázek.
 3  * Obrázek se nakreslí po zavolání metody kresli.
 4  * Obrázek může být změněn - můžete ho upravit na černobílý
 5  * a zpět na barevný.
 6  * Tato třida je napsána jako jeden z prvních příkladů
 7  * pro výuku Javy v BlueJ.
 8  *
 9  * @author    Michael Kolling
10  * @author    Luboš Pavlíček
11  * @version 1.0  (15 July 2000)
12  * @version 1.1cz (30 July 2004)
13 */
```

```
14 public class Obrazek {
15   private Ctverec zed;
16   private Ctverec okno;
17   private Trojuhelnik strecha;
18 
19   /**
20    * Konstruktor pro vytvoření instance třídy Obrazek
21    */
22   public Obrazek() {
23   // zde není žádný obsah
24   // datové atributy mají automaticky počáteční hodnotu null
25   // a vlastní kreslení obrázku
26   //(a tim i inicializace datových atributů)
27   //je v metodě kresli()
28   }
29   /**
30    * Nakreslí obrázek.
31    */
32   public void kresli() {
33       zed = new Ctverec();
34       zed.posunVertikalne(80);
35       zed.zmenVelikost(100);
36 
37       okno = new Ctverec();
38       okno.zmenBarvu("cerna");
39       okno.posunHorizontalne(20);
40       okno.posunVertikalne(100);
41 
42       strecha = new Trojuhelnik();
43       strecha.zmenVelikost(50, 140);
44       strecha.posunHorizontalne(60);
45       strecha.posunVertikalne(70);
46 }
47 
48   /**
49    * změní obrázek na černobílý
50    */
51   public void setCernoBily() {
52       if (zed != null) { // pouze pokud je již nakreslen
53            zed.zmenBarvu("cerna");
54            okno.zmenBarvu("bila");
55            strecha.zmenBarvu("cerna");
56       }
57   }
58 
59   /**
60    * změní obrázek zpět na barevný
61    */
62   public void setBarevny() {
63       if (zed != null) { // pouze pokud je již nakreslen domeček
64            zed.zmenBarvu("cervena");
65            okno.zmenBarvu("cerna");
66            strecha.zmenBarvu("zelena");
67       }
68   }
69 
```

```
70   /**
71    * při zavolání této metody by mělo před domečkem projet auto
72    */
73   public void prujezdAuta() {
74   if (zed != null) {  // pouze pokud je již nakreslen domeček
75      // ZDE DOPLNIT PŘÍSLUSNÝ KÓD
76   }
77   }
78 }
```

Třída `Obrazek` má tři datové atributy (viz řádky 15 až 17 kódu):

- `zed` a `okno` typu `Ctverec`,
- atribut `strecha` typu `Trojuhelnik`.

Tyto datové atributy vyjadřují základní prvky, ze kterých je sestaven obrázek. Na rozdíl od tříd `Ctverec`, `Kruh` a `Trojuhelnik` se instance třídy `Obrazek` nezobrazí na plátně po zavolání konstruktoru, ale až po zavolání metody `kresli()`. Konstruktor třídy `Obrazek` je proto prázdný. Metoda `kresli()` vytvoří instance jednotlivých částí obrázku, nastaví jejich umístění, velikost a barvu pomocí příslušných metod.
Metody pro změnu na černobílý (metoda `setCernoBily()` na řádcích 51 až 57) a barevný (metoda `setBarevny()` na řádcích 62 až 67) překreslují již vytvořený obrázek. Podmínka na začátku každé z těchto metod zjišťuje, zda je již nakreslena zeď (tj. zda byla vytvořena instance v metodě `kresli()` a uložena do proměnné `zed`). Pokud zavoláte metodu `setBarevny()` nebo metodu `setCernoBily()` před zavoláním metody `kresli()`, nic se nestane.

14.5. Postup řešení

14.5.1. Dokreslení obrázku (přidání slunce)

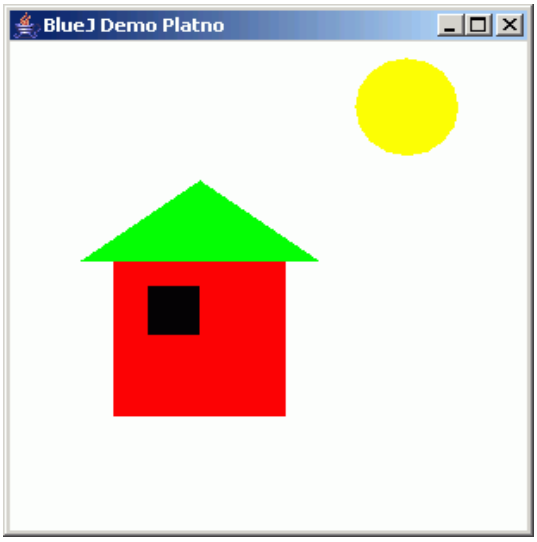
Naším prvním úkolem je přidat do obrázku slunce, které bude v této fázi řešení na obrázku neustále na jednom místě. Při změně na černobílý obrázek nebude slunce zobrazeno (je noc) – barva slunce se změní na bílou.
Slunce bude reprezentováno datovým atributem typu `Kruh`. Ve třídě `Obrazek` v části deklarace datových atributů (řádky 15 až 17 kódu) přibude následující řádek:

```
    private Kruh slunce;
```

Metoda `kresli()` se doplní o vytvoření instance třídy `Kruh` a změnu barvy, velikosti a umístění této instance. Kód by mohl vypadat takto:

```
    slunce = new Kruh();
    slunce.zmenBarvu("zluta");
    slunce.posunHorizontalne(180);
    slunce.posunVertikalne(-50);
    slunce.zmenVelikost(60);
```

Po přeložení, vytvoření instance třídy `Obrazek` a spuštění metody `kresli()` se zobrazí následující obrázek:



Obrázek 14.3 Vzhled obrázku po přidání slunce

Po úspěšném přeložení třídy `Obrazek` si všimněte, že se změnil diagram tříd zobrazený v BlueJ – automaticky přibyla vazba užití od třídy `Obrazek` ke třídě `Kruh`, neboť ve třídě `Obrazek` se nyní používá třída `Kruh`.
Ještě je potřeba upravit metody `setCernoBily()` a `setBarevny()`. Do metody `setCernoBily()` přidáme řádek se změnou barvy slunce na bílou (nebude na bílém pozadí vidět) a v metodě `setBarevny()` řádek se změnou barvy slunce na žlutou.

14.5.2. Průjezd auta

Pro řešení druhého úkolu byla již ve zdrojovém kódu třídy `Obrazek` předpřipravena metoda `prujezdAuta()`, při jejím zavolání by před domečkem mělo projet auto.
Je potřeba se rozhodnout, zda vytvořená instance třídy `Auto` bude datovým atributem či lokální proměnnou. Auto bude na obrázku pouze tehdy, když bude spuštěna tato metoda; nebude potřeba v žádné jiné metodě, při novém zavolání metody může vzniknout a projet nové auto. Z toho vyplývá, že instance třídy `Auto` může být lokální proměnná metody.
V metodě `prujezdAuta()` vytvoříme instance třídy `Auto` a zavoláme metodu pro jeho pomalý posun. Kód metody `prujezdAuta()` vidíte v následujícím výpisu:

```
1  public void prujezdAuta() {
2    if (zed != null) {   // pouze pokud je již nakreslen domeček
3        Auto ford = new Auto(0,240);
4        ford.pomaluPosunHorizontalne(300);
5    }
6  }
```

Animace auta je velmi nedokonalá – při průjezdu auto bliká, při nesprávném umístění bude auto umazávat domeček atd. V rámci tohoto jednoduchého příkladu však tyto problémy nebudeme řešit, jejich odstranění by vyžadovalo použití vícevrstvého plátna.
Po úspěšném přeložení třídy `Obrazek` se do diagramu tříd zobrazeného v BlueJ promítne další změna, přibude vazba užití od třídy `Obrazek` ke třídě `Auto`.

14.5.3. Východ a západ slunce

Pro práci se sluncem si lze vytvořit dvě představy:

- každé ráno vznikne nové slunce, při západu slunce zaniká (pohled „země-placka“) – v této variantě v metodě `vychodSlunce()` vzniká instance třídy `Kruh`, která představuje slunce,

v metodě `zapadSlunce()` tato instance zaniká. Instance kruhu se vytváří také na začátku (v metodě `kresli()`), neboť se začíná dnem a ne nocí,

- slunce obíhá kolem dokola, večer zapadne, ráno vyjde (pohled „geocentrický“[34]) – v této variantě vzniká pouze jedna instance, při západu slunce se příslušný kruh přesune mimo viditelnou část plátna, při východu se přesune na viditelnou část plátna.

Řešení metod ve variantě **„země-placka“** by mohlo vypadat následovně:

```
 1 /**
 2  * Při zavolání metody vyjde slunce, pokud již není na obloze.
 3  */
 4 public void vychodSlunce() {
 5  if (slunce == null) {//zapadlé slunce se pozná dle hodnoty null
 6  slunce = new Kruh();
 7  slunce.zmenBarvu("zluta");
 8  slunce.posunHorizontalne(-60);
 9  slunce.posunVertikalne(-10);
10  slunce.zmenVelikost(60);
11  slunce.pomaluPosunHorizontalne(60);
12  setBarevny();     // slunce je již na obrázku, nastává den
13  slunce.pomaluPosunHorizontalne(180);
14  }
15 }
16
17 /**
18  * Při zavolání metody zapadne slunce, pokud je na obloze.
19  */
20 public void zapadSlunce () {
21  if (slunce != null)  {    // slunce již musí existovat
22      slunce.pomaluPosunHorizontalne(90);
23      setCernoBily();
24      slunce.pomaluPosunHorizontalne(30);
25      slunce=null;        // zde se zruší odkaz na instanci slunce
26  }
27 }
```

V metodě `vychodSlunce()` nejprve zjistíme, jestli je slunce na obrázku nebo ne (řádek 5 výpisu). Pokud na obrázku žádné slunce není (proměnná slunce odkazuje na hodnotu `null`), vytvoříme instanci třídy `Kruh`, nastavíme barvu, posuneme kruh na kraj plátna a změníme jeho velikost (řádky 6 až 10 výpisu). Poté spustíme animaci východu slunce pomocí metody `pomaluPosunHorizontalne()`. Slunce popojde kousek po obrázku a nastane den – pro obarvení obrázku zavoláme metodu `setBarevny()`. Na řádku 13 slunce pokračuje ve svém pohybu po obloze.

Metoda pro západ slunce je velmi podobná. Pokud je slunce na obrázku (proměnná slunce odkazuje na instanci, tj. nemá hodnotu `null`), začne se pohybovat, obrázek se změní s barevného na černobílý a slunce zapadne úplně. Poté je instance zrušena, protože odkaz na ni je nastaven na hodnotu `null`.

[34] Pohled „heliocentrický“ je pro naši úlohu obtížně použitelný, neboť obrázek je kreslen z perspektivy uživatele na zemi.

Povšimněte si, že pokud spouštíme jinou metodu z téže třídy, uvádíme pouze název metody a případné parametry. Metoda bude spuštěna na té instanci, která spouští metodu obsahující toto volání (instance pošle zprávu sama sobě). Na řádku 12 výpisu se obrázek změní z černobílého na barevný – provedou se na tomto místě všechny příkazy obsažené v metodě `setBarevny()`.
Volání metody stejné instance by správně mělo obsahovat odkaz na sebe sama pomocí slova **this**:

```
this.setBarevny();
```

Java programátorovi usnadňuje práci tím, že nevyžaduje zápis pomocí `this` – překladač toto přiřazení do výsledného kódu doplní sám.

V **geocentrické variantě** je potřeba pomocí dalšího datového atributu sledovat, zda je den či noc – použijeme datový atribut `jeDen` typu `boolean`, který bude mít hodnotu `true`, pokud je den a `false`, pokud je noc. Instance třídy `Kruh` představující slunce vzniká pouze jednou, a to v metodě `kresli()`. V noci (na konci metody `zapadSlunce()`) se slunce přesune kolem zeměkoule na svoji výchozí pozici na východě. Deklarace a inicializace by mohla vypadat následovně:

```
    private boolean jeDen;   //  datový atribut
    .....
    public void kresli() {
    .....
         jeDen = true; // nastavení počáteční hodnoty
                       // v metodě kresli
    .....
    }
```

Vlastní kód metod `zapadSlunce()` a `vychodSlunce()` bude vypadat následovně:

```
 1 public void zapadSlunce () {
 2   if (jeDen)  {    // je den
 3       slunce.pomaluPosunHorizontalne(80);
 4       setCernoBily();
 5       slunce.pomaluPosunHorizontalne(20);
 6       slunce.posunHorizontalne(-360); // posunu do výchozí pozice
 7       jeDen = false; // poznačím si, že je noc
 8   }
 9 }
10 public void vychodSlunce() {
11   if (! jeDen) {   // je noc
12   slunce.pomaluPosunHorizontalne(20);
13   setBarevny();     // slunce je již na obrázku, nastává den
14   slunce.pomaluPosunHorizontalne(240);
15   jeDen = true;     // poznačím si, že je den
16   }
17 }
```

14.6. Domácí úkoly

1. Přidejte do projektu metody pro příjezd a odjezd auta.

2. Nakreslete si vlastní obrázek, např. sněhuláka či stromek.