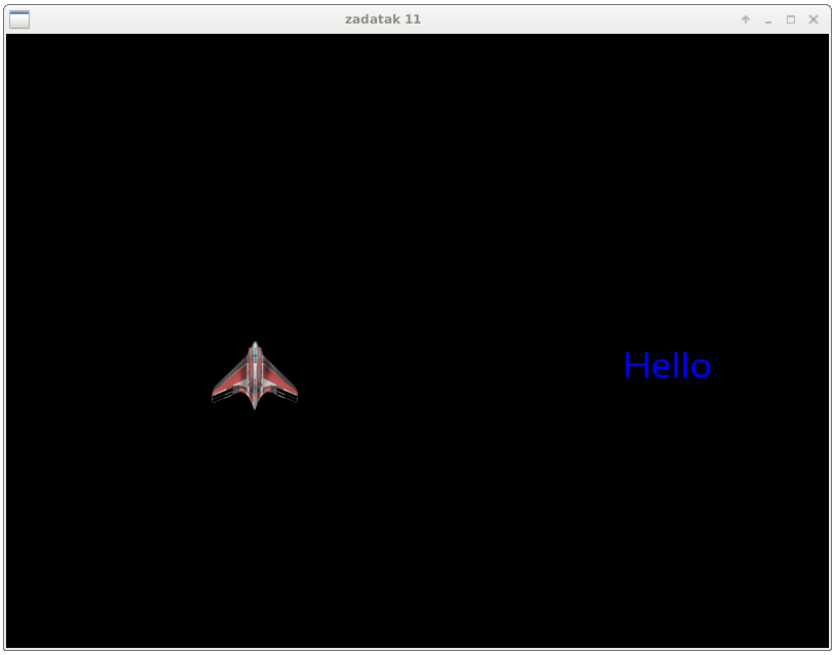
Zadatak

U ovom zadatku je zadan gotov program koji koristi SFML biblioteku *(Simple and Fast Multimedia Library*, www.sfml-dev.org). Program otvara jedan prozor s dva objekta, kao na slici:



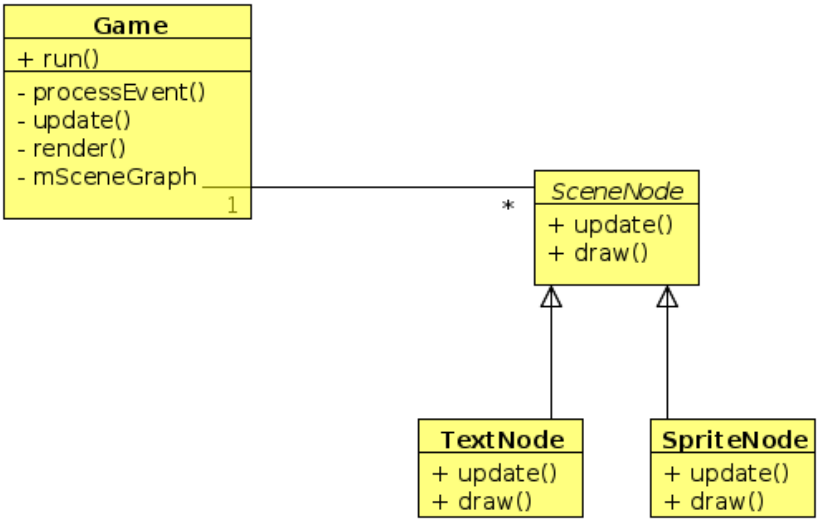
Na ekranu su dva objekta: slika aviona i tekst "Hello". S tipkom Ctrl-R možemo rotirati tekst, a s Alt-R rotiramo sliku. Program završava zatvaranjem ekrana. U praktikumima je potrebno zadatak **raditi pod Linuxom** jer je tamo biblioteka SFML instalirana jedino pod Linuxom.

Cijeli program je zapakiran u Commands.tar.gz. Nakon otvaranja naredbom

```
tar xvzf Commands.tar.gz
```

bit će kreiran direktorij **Commands**. U Code::Blocks IDE treba zatim otvoriti projekt **Commands/Commands.cbp**.

Dijagram klasa koje sudjeluju u programu dan je na ovoj slici.



Program funkcionira na sljedeći način:

1. U `Game::run()` metodi je beskonačna petlja u kojoj se procesiraju događaji, aktualizira i iscrtava scena.
2. Metoda `Game::processEvents()` preuzima od prozora (`mWindow`) događaje koji su se desili i postavljanjem lokalnih varijabli (`mRotate`) daje informaciju koji objekt treba, odnosno ne treba, rotirati.
3. Metoda `Game::update()` sada zna koje objekte treba rotirati i poziva njihove `update()` metode. Kod za rotaciju objekata je smješten u podklase `SceneNode` klase, u `TextNode::update()` i `SpriteNode::update()`.
4. Metoda `Game::render()` iscrtava scenu pozivajući `draw()` metodu svih objekata.

Nedostatak ovog pristupa se vidi ako pokušamo dodati nove manipulacije objektima na sceni (na primjer translacije). Kod za translatiranje trebali bi smjestiti u `TextNode::update()` i `SpriteNode::update()` metode, i `update()` metode bi trebale dobiti parametar koji bi signalizirao operaciju koja se zahtijeva.

Zadatak je redizajnirati sustav prijenosa naredbi korištenjem oblikovnog obrasca **Command**. Obrazac se sastoji u sljedećem: metoda `Game::processEvents()` će za svaki događaj koji traži aktualizaciju nekog objekta na sceni generirati jedan objekt tipa `Command` koji će nositi dvije informacije -- informaciju o scenskom objektu koju treba aktualizirati i samu funkciju koju treba izvršiti na objektu. Budući da može biti više događaja koji čekaju procesiranje metoda `Game::processEvents()` ima formu:

```
void Game::processEvents(){
    // ..
      sf::Event event;
      while(mWindow.pollEvent(event))
      {
          // procesiraj sve događaje koji čekaju u redu događaja.
      }
}
```

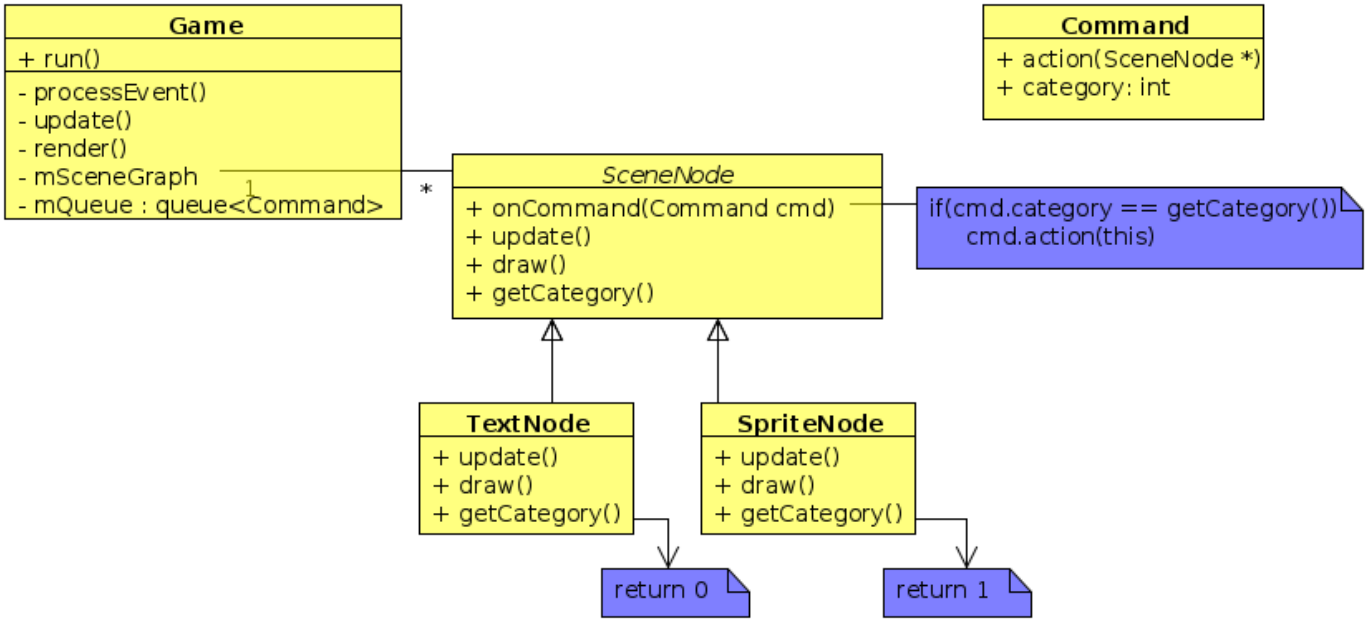
Stoga ćemo `Command` objekte koje generiramo unutar while-petlje u `Game::processEvents()` smještati u jedan red (tip `std::queue<Command>`) kako se niti jedan događaj ne bi "izgubio".

Klasa `Command` neka ima formu:

```
class SceneNode;
struct Command
{
        std::function<void(SceneNode*)> action;
        int category= -1;
};
```

`category` je cijeli broj koji selektira scenski objekt (npr. 0 za tekst "Hello" i 1 za avion). `action` je funkcija koja uzima pokazivač na `SceneNode` i obavlja traženi posao (rotaciju). Zamotali smo je u `std::function` kako bismo mogli koristiti lambda-izraze za implementaciju akcije (korištenje lambdi je nužno za 100 bodova).

Novi kod ima ovu strukturu:



Kod funkcionira na ovaj način:

- Metoda `Game::processEvents()` generira komande za rotaciju objekata i gura ih u red komadi (`mQueue`).
- Metoda `Game::update()` na svim objektima scene zove `onCommand()` metodu predajući im jednu po jednu sve naredbe iz `mQueue`. Pri tome je važno napraviti `pop()` svake naredbe nakon što je predana svim objektima kako bi se `mQueue` na kraju ispraznio.
- `TextNode::update()` i `SpriteNode::update()` metode sada mogu ostati prazne. U složenijim situacijama one bi obavljale ostalu aktualizaciju scenskih objekta.
- `onCommand()` metoda će obaviti traženu akciju (rotaciju) na objektu ako se njegova kategorija podudara s kategorijom u naredbi. Inače se naredba ignorira.

Drugi dio zadatka. Nakon što rotiranje ponovo dobro funkcionira treba dodati naredbe za translatiranje objekata. Neka **Ctrl-Up** pomiće sliku prema gore, a **Alt-Up** neka pomiće tekst prema gore; pri tome je Up strelica prema gore. Analogno, napravite pomicanje u svim smjerovima oba objekta.

Uputa: Za strelicu prema gore je traženi kod `sf::Keyboard::Up`, za dolje `sf::Keyboard::Down`, lijevo `sf::Keyboard::Left` i desno `sf::Keyboard::Right`. Imajte na umu da y-koordinata raste prema dolje.

Pomicanje objekta se vrši pomoći naredbe `move(x,y)` koja je definirana u `sf::Transformable` klasi pa je automatski nasljeđuju scenski objekti. Koordinate `x` i `y` su tipa `float` i predstavljaju pomak objekta u pixelima. Možete uvijek vršiti pomicanje za 10 piksela.

Last Modified: *February 26, 2015*

<>