

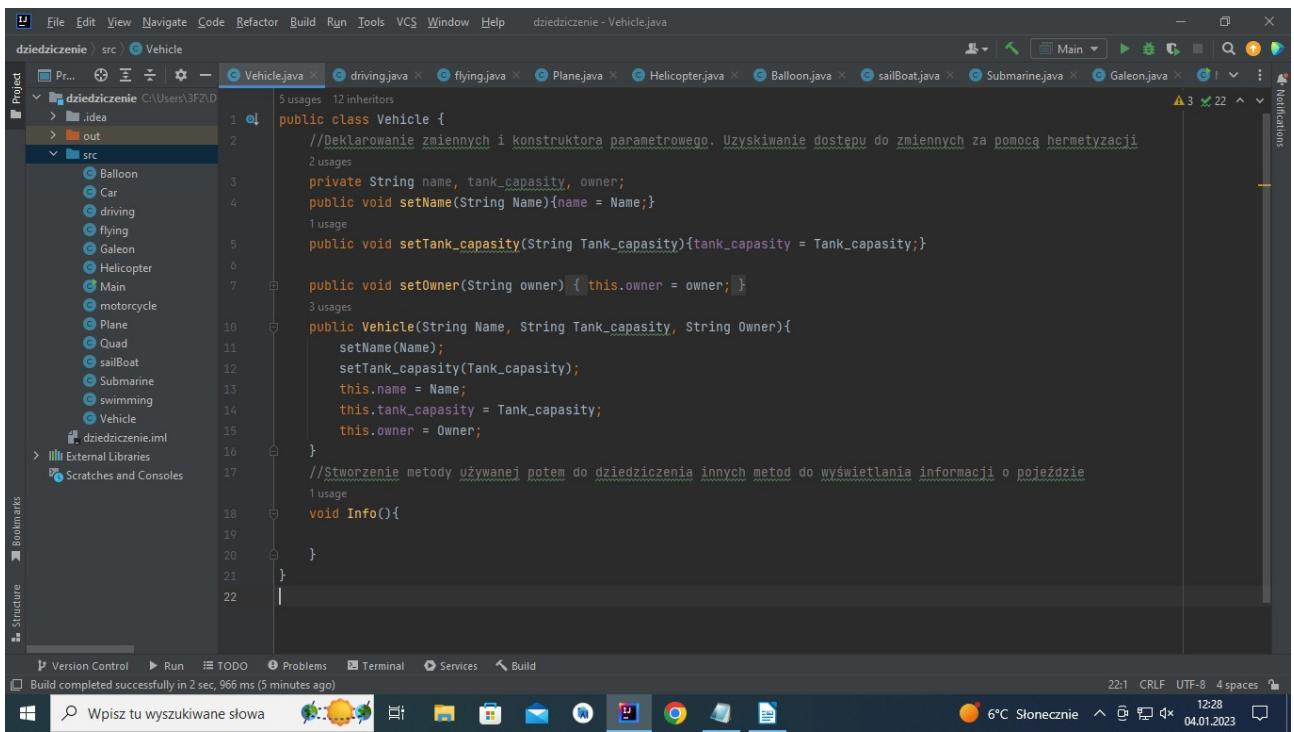
Michał Wiliński

Dokumentacja projektu z obiektowości

Spis Treści:

- Nadklaś Vahicle str. 2
- Podklaś driving i jej podklaś str. 2,5 - 4
- Podklaś flying i jej podklaś str. 4,5 – 6
- Podklaś swimming i jej podklaś 6,5 – 9
- Klaś Main i wyniki komplikacji str. 9,5

Nadklaśa Vehicle:



```
public class Vehicle {
    //Deklarowanie zmiennych i konstruktora parametrycznego. Uzyskiwanie dostępu do zmiennych za pomocą hermetyzacji
    private String name, tank_capacity, owner;
    public void setName(String Name){name = Name;}
    public void setTank_capacity(String Tank_capacity){tank_capacity = Tank_capacity;}
    public void setOwner(String owner) {this.owner = owner;}
    public Vehicle(String Name, String Tank_capacity, String Owner){
        setName(Name);
        setTank_capacity(Tank_capacity);
        this.name = Name;
        this.tank_capacity = Tank_capacity;
        this.owner = Owner;
    }
    //Stworzenie metody używanej potem do dziedziczenia innych metod do wyświetlania informacji o pojeździe
    void Info(){
    }
}
```

Podklaśa driving(dziedzicząca od Vehicle)

The screenshot shows the IntelliJ IDEA interface with the project 'dziedziczenie' open. The 'driving.java' file is the active editor. The code defines a class 'driving' that extends 'Vehicle'. It includes methods for setting the engine and speed, and a constructor that calls the superclass constructor and initializes the engine. The code is annotated with comments explaining the purpose of each part.

```
public class driving extends Vehicle{
    //Tworzenie zmiennych i uzyskiwanie do nich dostępu za pomocą hermetyzacji
    private String engine;
    private int speed;
    public void setEngine(String Engine){engine = Engine;}
    public void setSpeed(int Speed){speed = Speed;}
    //Odwzorowanie się do nadklasta za pomocą znacznika super i tworzenie własnego konstruktora parametrywego
    public driving(String name, String tank_capacity, String Owner, int speed, String engine){
        super(name, tank_capacity, Owner);
        setEngine(engine);
        setSpeed(speed);
        this.speed = speed;
        this.engine = engine;
    }
}
```

Podklaśa Quad(dziedzicząca od klasy driving)

The screenshot shows the IntelliJ IDEA interface with the project 'dziedziczenie' open. The 'Quad.java' file is the active editor. The code defines a class 'Quad' that extends 'driving'. It includes a constructor that calls the superclass constructor and sets the daperType. It also contains an 'Info' method that prints information about the Quad. The code is annotated with comments explaining the purpose of each part.

```
public class Quad extends driving{
    //Tworzenie zmiennych i uzyskiwanie do nich dostępu
    private String daperType;
    public void setDaperType(String DaperType){daperType = DaperType;}
    //Tworzenie konstruktora i odwoływanie się do konstruktora nadklasta
    public Quad(String name, String tank_capacity, String Owner, int speed, String engine, String DaperType){
        super(name, tank_capacity, Owner, speed, engine );
        setDaperType(DaperType);
        daperType = DaperType;
        Info(name, tank_capacity,speed, engine, daperType, Owner);
    }
    //Tworzenie metody wyświetlającej informacje na temat Quadu
    public void Info(String name, String tank_capacity, int speed, String engine , String daperType, String owner){
        System.out.println("Właściciel : " + owner + ". Quad o nazwie: " + name + ". Pojemność silnika :" + tank_capacity + ". Wy ciąga");
    }
}
```

Podklaśa Car(dziedzicząca od Vehicle)

The screenshot shows an IDE interface with the project 'dziedziczenie' open. The 'src' folder contains several classes: Balloon, Car, driving, flying, Galeon, Helicopter, Main, motorcycle, Quad, Plane, sailBoat, Submarine, swimming, Vehicle. The 'Car.java' file is selected and its code is displayed:

```
dziedziczenie > src > Car > Info
```

```
dziedziczenie | src | Pr... | File Edit View Navigate Code Refactor Build Run Tools VCS Window Help dziedziczenie - Car.java
```

```
Project | Bookmarks | Structure | Version Control | Run | TODO | Problems | Terminal | Services | Build | Notifications
```

```
1 usage
public class Car extends driving{
    //Tworzenie zmiennych i uzyskiwanie do nich dostepu
    3 usages
    private int number_of_passagers;
    1 usage
    public void setNumber_of_passagers(int NumberOfPassagers){number_of_passagers = NumberOfPassagers;}
    //Tworzenie konstruktora oraz odwoływanie się do konstruktora nadklasy
    1 usage
    public Car(String name, String tank_capacity, String Owner, int speed, String engine, int Number_of_passagers){
        super(name, tank_capacity, Owner, speed, engine );
        setNumber_of_passagers(Number_of_passagers);
        number_of_passagers = Number_of_passagers;
        Info(name, tank_capacity, speed, engine, number_of_passagers, Owner);
    }
    //Tworzenie metody wyświetlającej wszystkie informacje na temat samochodu
    public void Info(String name, String tank_capacity, int speed, String engine, int number_of_passagers, String owner){
        System.out.println("Własciciel : " + owner + ". Samochód o nazwie: " + name + ". Pojemność baku: " + tank_capacity + ". Maksymalna prędkość: " + speed + ". Silnik: " + engine + ". Liczba pasażerów: " + number_of_passagers);
    }
}
```

Below the code editor, the status bar shows: Typo: In word 'samochodu'. The system tray indicates it's 12:30, 6°C, Słonecznie, and the date is 04.01.2023.

Podklaśa motorcycle(dziedzicząca od driving)

The screenshot shows an IDE interface with the project 'dziedziczenie' open. The 'src' folder contains several classes: Balloon, Car, driving, flying, Galeon, Helicopter, Main, motorcycle, Quad, Plane, sailBoat, Submarine, swimming, Vehicle. The 'motorcycle.java' file is selected and its code is displayed:

```
dziedziczenie > src > motorcycle
```

```
dziedziczenie | src | Pr... | File Edit View Navigate Code Refactor Build Run Tools VCS Window Help dziedziczenie - motorcycle.java
```

```
Project | Bookmarks | Structure | Version Control | Run | TODO | Problems | Terminal | Services | Build | Notifications
```

```
1 usage
public class motorcycle extends driving{
    //Tworzenie zmiennych i uzyskiwanie do nich dostepu
    3 usages
    private String suspension;
    1 usage
    public void setSuspension(String Suspension){suspension = Suspension;}
    //Tworzenie konstruktora i odwoływanie się do konstruktora nadklasy
    1 usage
    public motorcycle(String name, String tank_capacity, String Owner, int speed, String engine, String Suspension){
        super(name, tank_capacity, Owner, speed, engine );
        setSuspension(Suspension);
        suspension = Suspension;
        Info(name, tank_capacity, speed, engine, suspension, Owner);
    }
    //Tworzenie metody wyświetlającej wszystkie informacje o Motocyklu
    public void Info(String name, String tank_capacity, int speed, String engine, String suspension, String owner){
        System.out.println("Własciciel : " + owner + ". Motocykl o nazwie: " + name + ". Pojemność baku: " + tank_capacity + ". Maksymalna prędkość: " + speed + ". Silnik: " + engine + ". Wyposażenie: " + suspension);
    }
}
```

Below the code editor, the status bar shows: Build completed successfully in 2 sec, 966 ms (8 minutes ago). The system tray indicates it's 12:31, 6°C, Słonecznie, and the date is 04.01.2023.

Podklaśa Flying(dziedzicząca od Vehicle)

```
public class flying extends Vehicle{
    //Tworzenie zmiennych i uzyskiwanie do nich dostepu
    private int max_height_flight;

    public void setMax_height_flight(int max_height_flight) { this.max_height_flight = max_height_flight; }

    public flying(String name, String tank_capacity, String Owner, int Max_height_flight){
        super(name, tank_capacity, Owner);
        setMax_height_flight(Max_height_flight);
        max_height_flight = Max_height_flight;
    }
}
```

Podkلاza Plane(dziedzicząca od klasy flying)

```
public class Plane extends flying{
    //Tworzenie zmiennych i uzyskiwanie do nich dostepu
    private String type;

    public void setType(String Type){type = Type;}

    public Plane(String name, String tank_capacity, String Owner, int max_height_flight, String Type){
        super(name, tank_capacity, Owner, max_height_flight);
        setType(Type);
        type = Type;
        Info(name, tank_capacity, max_height_flight, type, Owner);
    }

    void Info(String name, String tank_capacity, int max_height_flight, String type, String owner){
        System.out.println("Wlasnosc : " + owner + ". Samolot o nazwie: " + name + ". Pojemnosc baku: " + tank_capacity + ". Maksymalna wysokosc lotu: " + max_height_flight + ". Typ: " + type);
    }
}
```

Podklaza Helicopter(dziedzicząca od klasy flying)

The screenshot shows the IntelliJ IDEA interface with the project 'dziedziczenie' open. The 'src' directory contains several classes: Balloon, Car, driving, flying, Galeon, Helicopter, Main, motorcycle, Plane, Quad, sailBoat, Submarine, swimming, and Vehicle. The 'Helicopter.java' file is currently selected and displayed in the editor. The code defines a class 'Helicopter' that extends 'flying'. It includes methods for setting propeller length and displaying information about the helicopter.

```
public class Helicopter extends flying{
    //Tworzenie zmiennych i uzyskiwanie do nich dostepu
    private int Propeller_lenght;

    //Tworzenie konstruktora i odwoływanie się do konstruktora nadklasy
    public Helicopter(String name, String tank_capacity, String Owner, int max_height_flight, int Propeller_lenght){
        super(name, tank_capacity, Owner, max_height_flight);
        setType(Propeller_lenght);
        Propeller_lenght = Propeller_lenght;
        Info(name, tank_capacity, max_height_flight, Propeller_lenght, Owner);
    }

    //Tworzenie metody do wyświetlania wszystkich informacji o Helikopterze

    void Info(String name, String tank_capacity, int max_height_flight, int Propeller_lenght, String owner){
        System.out.println("Własciciel : " + owner + ". Samolot o nazwie: " + name + ". Pojemność baku: " + tank_capacity + ". Maksymalna wysokość lotu: " + max_height_flight + ". Długość śmigła: " + Propeller_lenght);
    }
}
```

Podkласa Balloon(dziedzicząca od klasy flying)

The screenshot shows the IntelliJ IDEA interface with the project 'dziedziczenie' open. The 'src' directory contains the same set of classes as the previous screenshot. The 'Balloon.java' file is currently selected and displayed in the editor. The code defines a class 'Balloon' that extends 'flying'. It includes methods for setting torch power and displaying information about the balloon.

```
public class Balloon extends flying{
    //Tworzenie zmiennych i uzyskiwanie do nich dostepu
    private int Torch_power;

    //Tworzenie konstruktora i odwoływanie się do konstruktora nadklasy
    public Balloon(String name, String tank_capacity, String Owner, int max_height_flight, int Torch_Power){
        super(name, tank_capacity, Owner, max_height_flight);
        setType(Torch_Power);
        Torch_power = Torch_Power;
        Info(name, tank_capacity, max_height_flight, Torch_power, Owner);
    }

    //Tworzenie metody do wyświetlania wszystkich informacji o balonie

    void Info(String name, String tank_capacity, int max_height_flight, int Torch_Power, String owner){
        System.out.println("Własciciel : " + owner + ". Samolot o nazwie: " + name + ". Pojemność baku: " + tank_capacity + ". Maksymalna wysokość lotu: " + max_height_flight + ". Długość torcza: " + Torch_Power);
    }
}
```

Podklasa Swimming(dziedzicząca od klasy Vehicle)

The screenshot shows a Java project named 'dziedziczenie' with a source file 'swimming.java'. The code defines a class 'swimming' that extends 'Vehicle'. It includes a constructor that takes parameters like name, tank capacity, owner, and max weight, and sets them using a constructor delegation call. A method 'setMax_weight' is also defined.

```
public class swimming extends Vehicle{
    //tworzenie zmiennych i uzyskiwanie do nich dostepu
    private int max_weight;

    public swimming(String name, String tank_capacity, String Owner, int Max_weight){
        super(name, tank_capacity, Owner);
        setMax_weight(Max_weight);
        max_weight = Max_weight;
    }

    public void setMax_weight(int max_weight) { this.max_weight = max_weight; }
}
```

Podkласa SailBoat(dziedzicząca od klasy swimming)

The screenshot shows the same Java project with a new source file 'sailBoat.java'. This class extends the 'swimming' class and adds its own specific attribute 'number_of_sails'. It includes a constructor that initializes this attribute and a method 'Info' that prints out details about the sailboat.

```
public class sailBoat extends swimming{
    private int number_of_sails;

    public void setNumber_of_sails(int number_of_sails) { this.number_of_sails = number_of_sails; }

    public sailBoat(String name, String tank_capacity, String Owner, int max_weight, int Number_of_sails){
        super(name, tank_capacity, Owner, max_weight);
        setNumber_of_sails(Number_of_sails);
        number_of_sails = Number_of_sails;
        Info(name, tank_capacity, max_weight, number_of_sails, Owner);
    }

    public void Info(String name, String tank_capacity, int max_weight, int number_of_sails, String owner){
        System.out.println("Wlasnosc : " + owner + ". Zaglowka o nazwie: " + name + ". Pojemosc baku: " + tank_capacity + ". Maksymalna "
    }
}
```

Podklasa Submarine(dziedzicząca od klasy swimming)

```
public class Submarine extends swimming{
    //Tworzenie zmiennych i uzyskiwanie do nich dostępu
    private int max_Depth;

    public void setNumber_of_sails(int Max_Depth) { max_Depth = Max_Depth; }

    public Submarine(String name, String tank_capacity, String Owner, int max_weight, int Max_Depth){
        super(name, tank_capacity, Owner, max_weight);
        setNumber_of_sails(Max_Depth);
        max_Depth = Max_Depth;
        Info(name, tank_capacity, max_weight, max_Depth, Owner);
    }

    public void Info(String name, String tank_capacity, int max_weight, int Max_Depth, String owner){
        System.out.println("Własciciel : " + owner + ". Zagłówka o nazwie: " + name + ". Pojemność baku: " + tank_capacity + ". Maksymalna głębokość: " + max_Depth);
    }
}
```

Podklasa Galeon(dziedzicząca od klasy swimming)

```
public class Galeon extends swimming{
    //Tworzenie zmiennych i uzyskiwanie do nich dostępu
    private String Jolly_roger;

    public void setNumber_of_sails(String Jolly_Roger) { Jolly_roger = Jolly_Roger; }

    public Galeon(String name, String tank_capacity, String Owner, int max_weight, String Jolly_Roger){
        super(name, tank_capacity, Owner, max_weight);
        setNumber_of_sails(Jolly_Roger);
        Jolly_roger = Jolly_Roger;
        Info(name, tank_capacity, max_weight, Jolly_roger, Owner);
    }

    public void Info(String name, String tank_capacity, int max_weight, String Jolly_roger, String owner){
        System.out.println("Własciciel : " + owner + ". Zagłówka o nazwie: " + name + ". Pojemność baku: " + tank_capacity + ". Maksymalna głębokość: " + max_weight);
    }
}
```

Klasa Main oraz wynik kompilacji

The screenshot shows an IDE interface with a Java project named "dziedziczenie". The "src" package contains several classes: Balloon, Car, driving, flying, Galeon, Helicopter, Main, motorcycle, Plane, Quad, sailBoat, Submarine, swimming, and Vehicle. The "Main.java" file is open and contains the following code:

```
public static void main(String[] args) {
    //Utworzenie Szeregu Heterologicznego od nadzby Vahicle
    Vehicle v[] = new Vehicle[10];
    v[0] = new Quad( name: "Quad1", tank_capacity: "100L", Owner: "Jarowslaw", speed: 80, engine: "tłokowy", DaperType: "terenowe");
    v[1] = new motorcycle( name: "motor1", tank_capacity: "50L", Owner: "Mateusz", speed: 50, engine: "tłokowy", Suspension: "10");
    v[2] = new Car( name: "Auto1", tank_capacity: "100L", Owner: "Michał", speed: 150, engine: "tłokowy", Number_of_passagers: 5);
    v[3] = new Plane( name: "Samolot1", tank_capacity: "1000L", Owner: "Jakub", max_height_flight: 2000, Type: "Odrzutowy");
    v[4] = new sailBoat( name: "Zaglowka1", tank_capacity: "200L", Owner: "Filip", max_weight: 5000, Number_of_sails: 2);
    v[5] = new Helicopter( name: "Helikopter1", tank_capacity: "500L", Owner: "Maksym", max_height_flight: 1000, Propeller_Lenght: 8);
    v[6] = new Balloon( name: "Balon1", tank_capacity: "300kg", Owner: "Kamil", max_height_flight: 800, Torch_Power: 70);
    v[7] = new Submarine( name: "Podwodna1", tank_capacity: "800L", Owner: "Marcel", max_weight: 10000, Max_Depth: 5000);
    v[8] = new Galeon( name: "Galeon1", tank_capacity: "700L", Owner: "Mateusz", max_weight: 10000, Jolly_Roger: "Czaszka Piracka");
    //Wyswietlanie w petli informacji o kazdym obiekcie zapisany w tablicy
    for (int i = 0; i < v.length; i++){
        v[i].info();
    }
}
```

The "Run" tab shows the output of the program's execution:

```
Wlasciciel : Jarowslaw. Quad o nazwie: Quad1. Pojemnosc silnika:100L. Wyciągajacy maks predkosc: 80km/h. O silniku: tlokowy. Typ amortyzatorow: terenowe
Wlasciciel : Mateusz. Motocykl o nazwie: motor1. Pojemnosc baku: 50L. Maksymalna predkosc: 50km/h. Typ silnika: tlokowy. Wysokosc zawieszenia: 10cm
Wlasciciel : Michal. Samochód o nazwie: Auto1. Pojemnosc baku: 100L. Maksymalna predkosc: 150km/h. Typ silnika: tlokowy. Maksymalna liczba pasazerow: 5
Wlasciciel : Jakub. Samolot o nazwie: Samolot1. Pojemnosc baku: 1000L. Maksymalna wysokosc lotu: 2000m.n.p.m. Typ samolotu: Odrzutowy
Wlasciciel : Filip. Zaglowka o nazwie: Zaglowka1. Pojemnosc baku: 200L. Maksymalny dozwolony ciezar na pokladzie: 5000kg. Liczba zagli: 2
Wlasciciel : Maksym. Samolot o nazwie: Helikopter1. Pojemnosc baku: 500L. Maksymalna wysokosc lotu: 1000m.n.p.m. DLugosc smigiel helikoptera: 8m
Wlasciciel : Kamil. Samolot o nazwie: Balon1. Pojemnosc baku: 300kg. Maksymalna wysokosc lotu: 800m.n.p.m. Sila ognia w balonie: 70
Wlasciciel : Marcel. Zaglowka o nazwie: Podwodna1. Pojemnosc baku: 800L. Maksymalny dozwolony ciezar na pokladzie: 10000kg. Maksymalne zanuzenie pod woda: 5000m
Wlasciciel : Mateusz. Zaglowka o nazwie: Galeon1. Pojemnosc baku: 700L. Maksymalny dozwolony ciezar na pokladzie: 10000kg. Flaga statku: Czaszka Piracka
```

The status bar at the bottom indicates: "Build completed successfully in 2 sec. 389 ms (moments ago)".