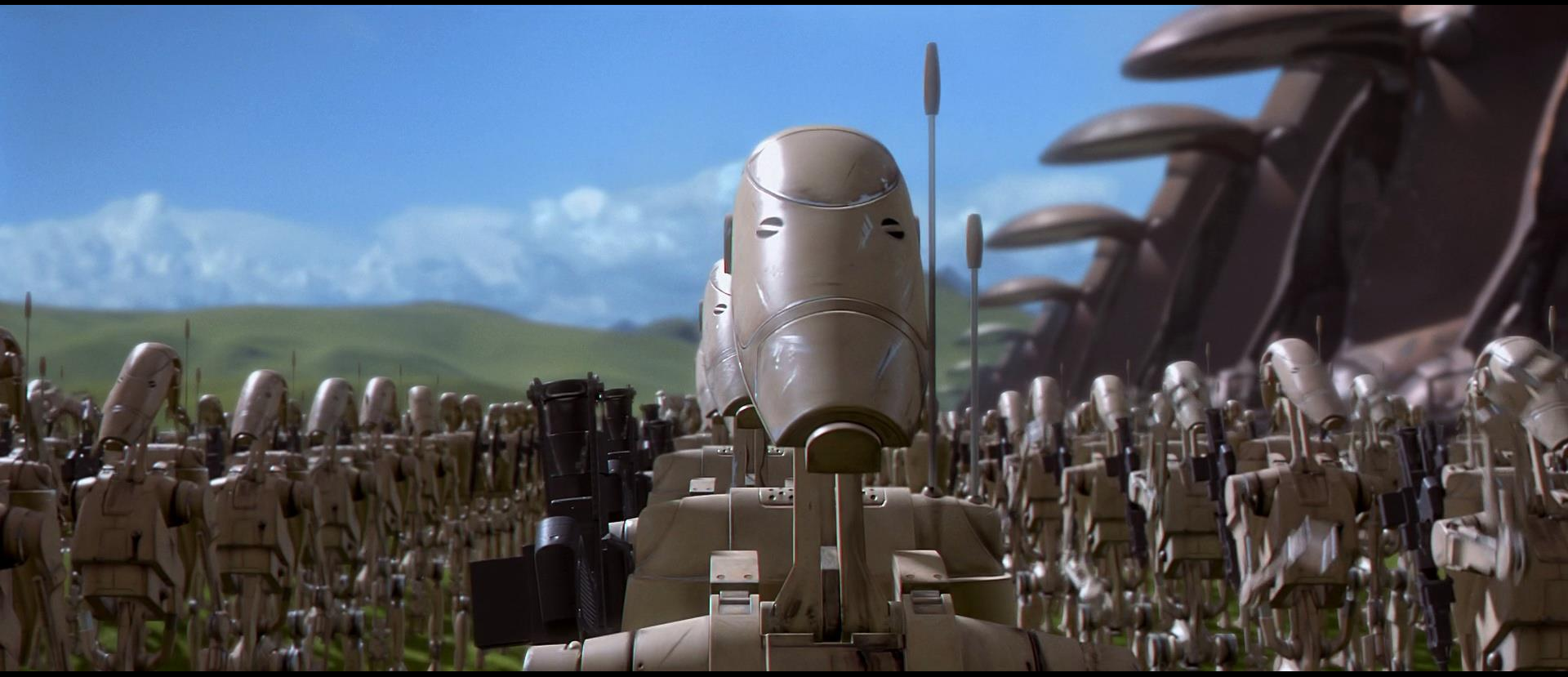
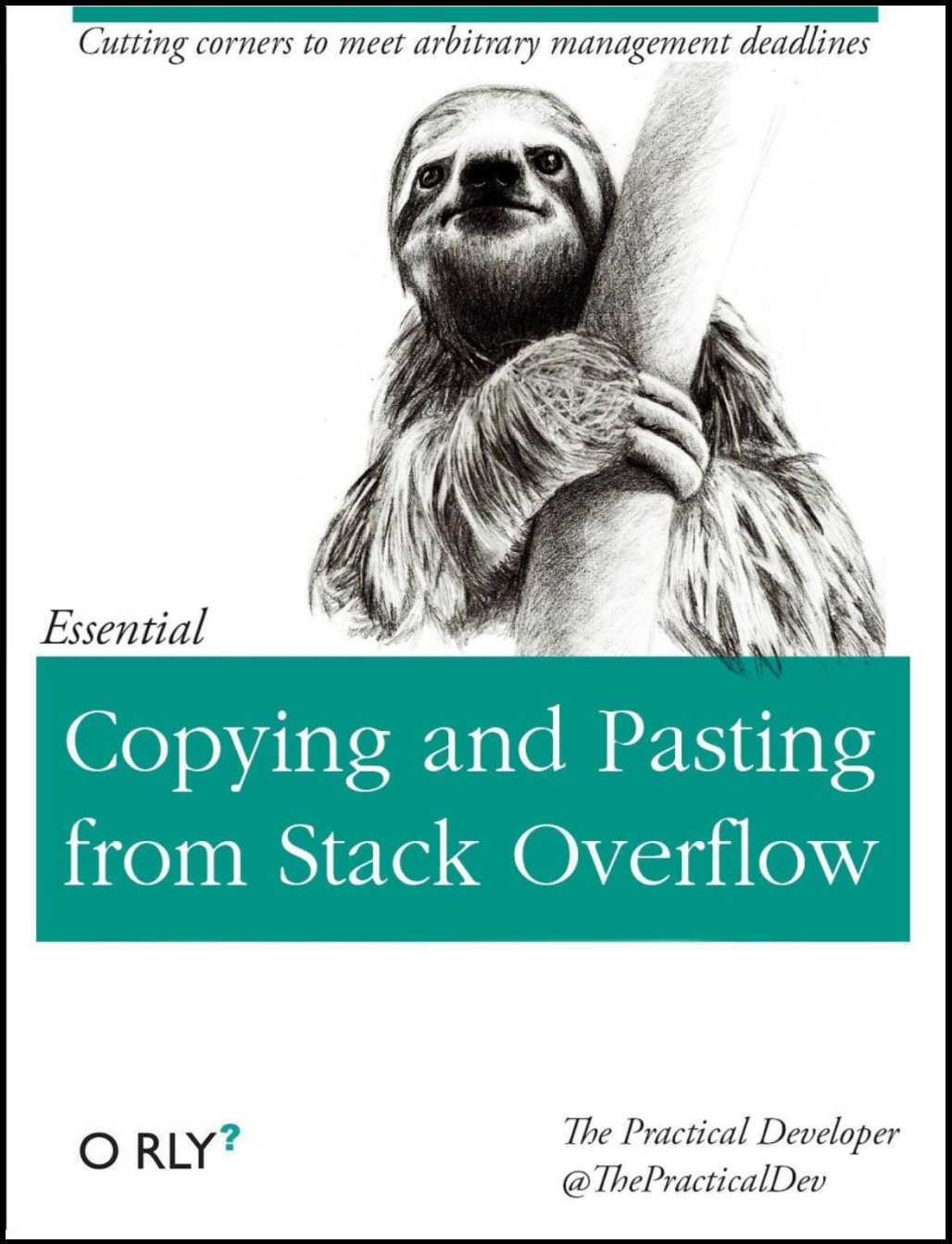
THE DARK SIDE OF THE CODE

BARTLOMIEJ GULIS
WROCLAW, 2017 IV 12



HOW MANY, ACTUALLY?



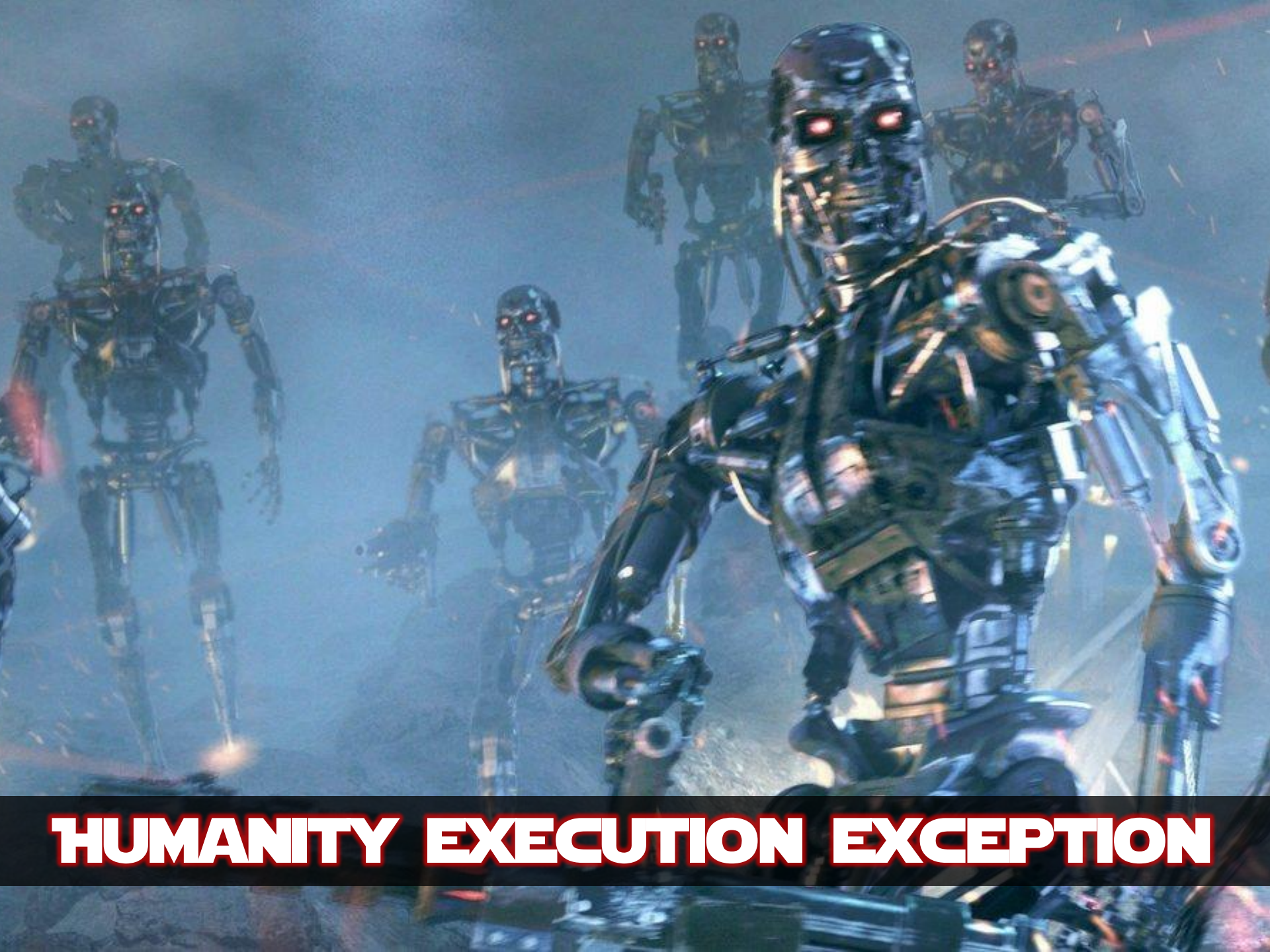
50 000 000 UNIQUE VISITS OF 7 500 000 000 TOTAL

0,67 %


– JUDGE ME BY MY SIZE, DO YOU?
HM? HMM...


IT SECTORS

EVERYTHING IS CONNECTED

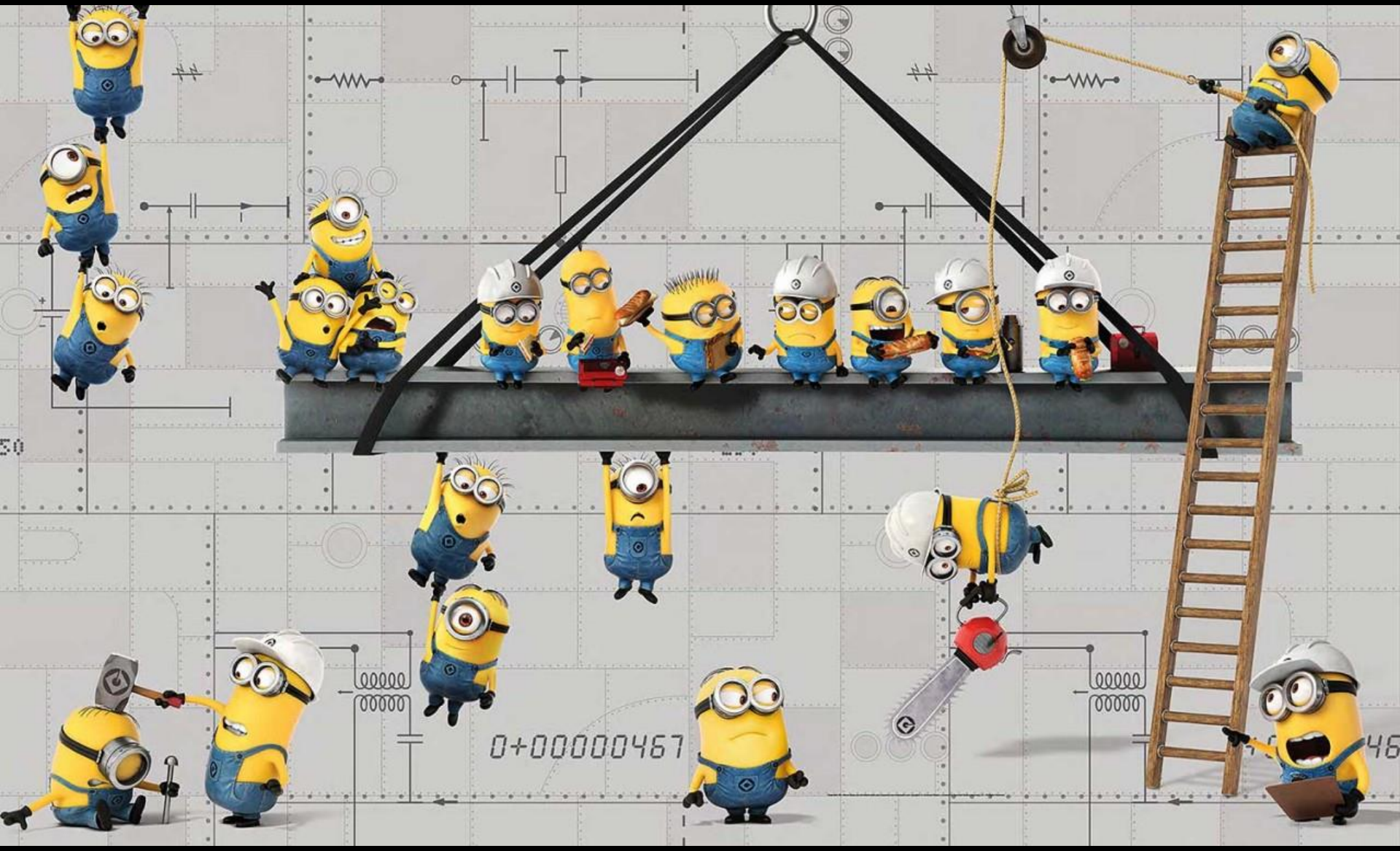

HUMANITY EXECUTION EXCEPTION

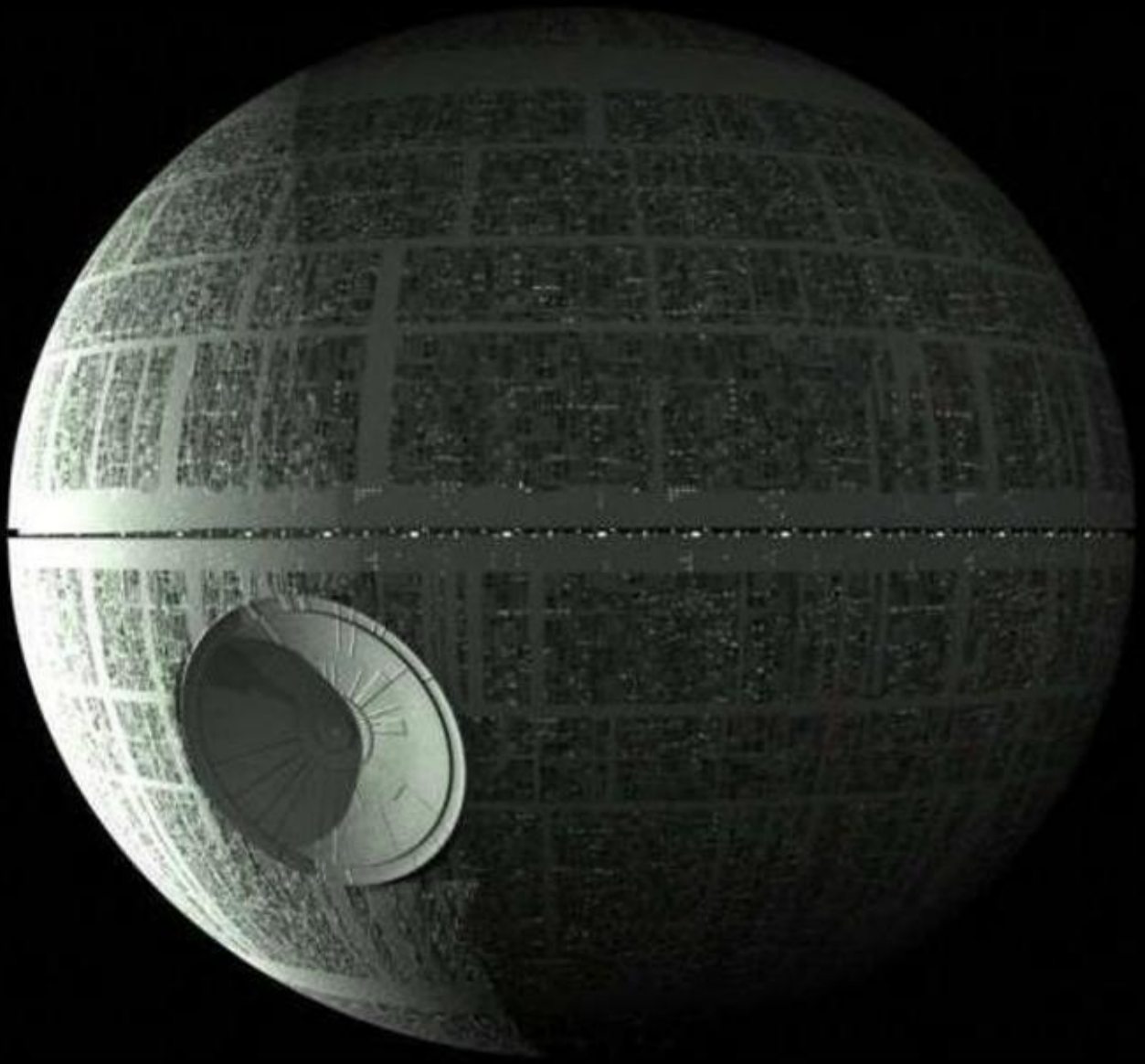

PROGRAMMERS
PROGRAMMERS EVERYWHERE
memegenerator.net

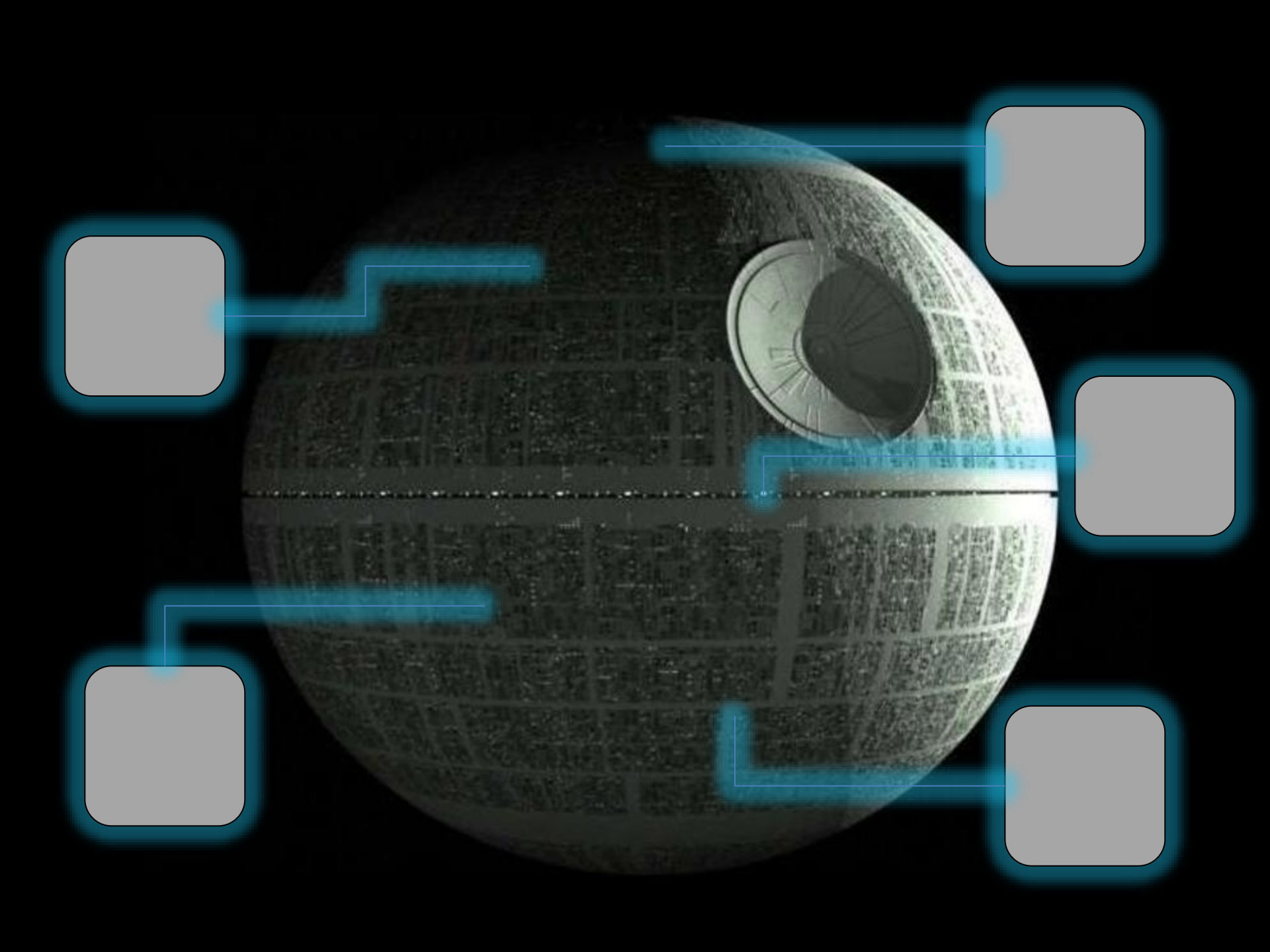

TOGETHER WE CAN RULE THE GALAXY


THE CODE IS STRONG IN THIS ONE


PWNAGE.RO


0+00000467





```java
public class ControlPanel {

    public void destroy(Planet planet) throws PlanetOutOfRangeException {
        // ...
    }

    public Connection call(Number number) throws NotEnoughCreditException {
        // ...
    }

}
```

WHAT IS WRONG WITH THIS CLASS?

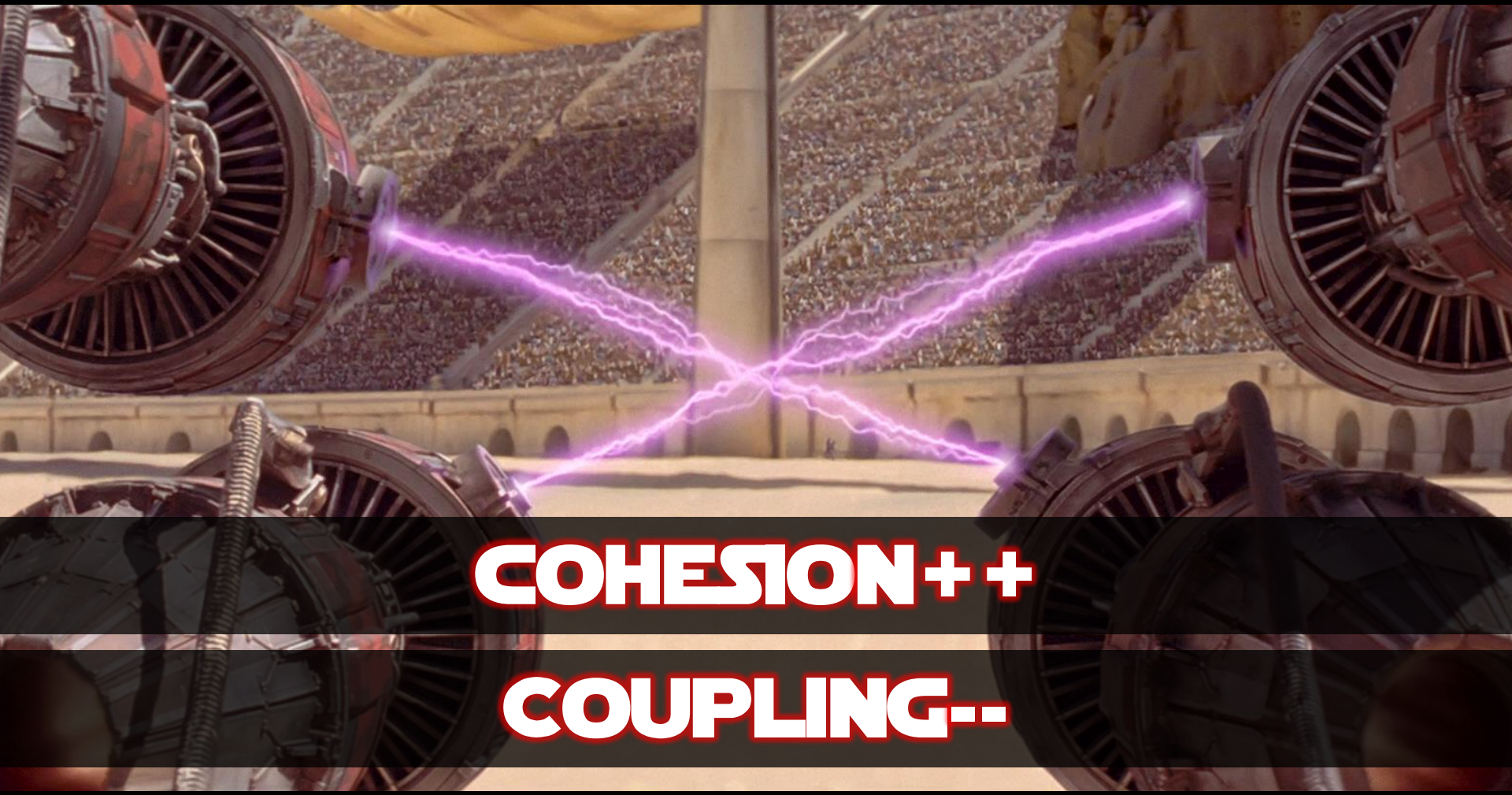
TOO MANY RESPONSIBILITIES

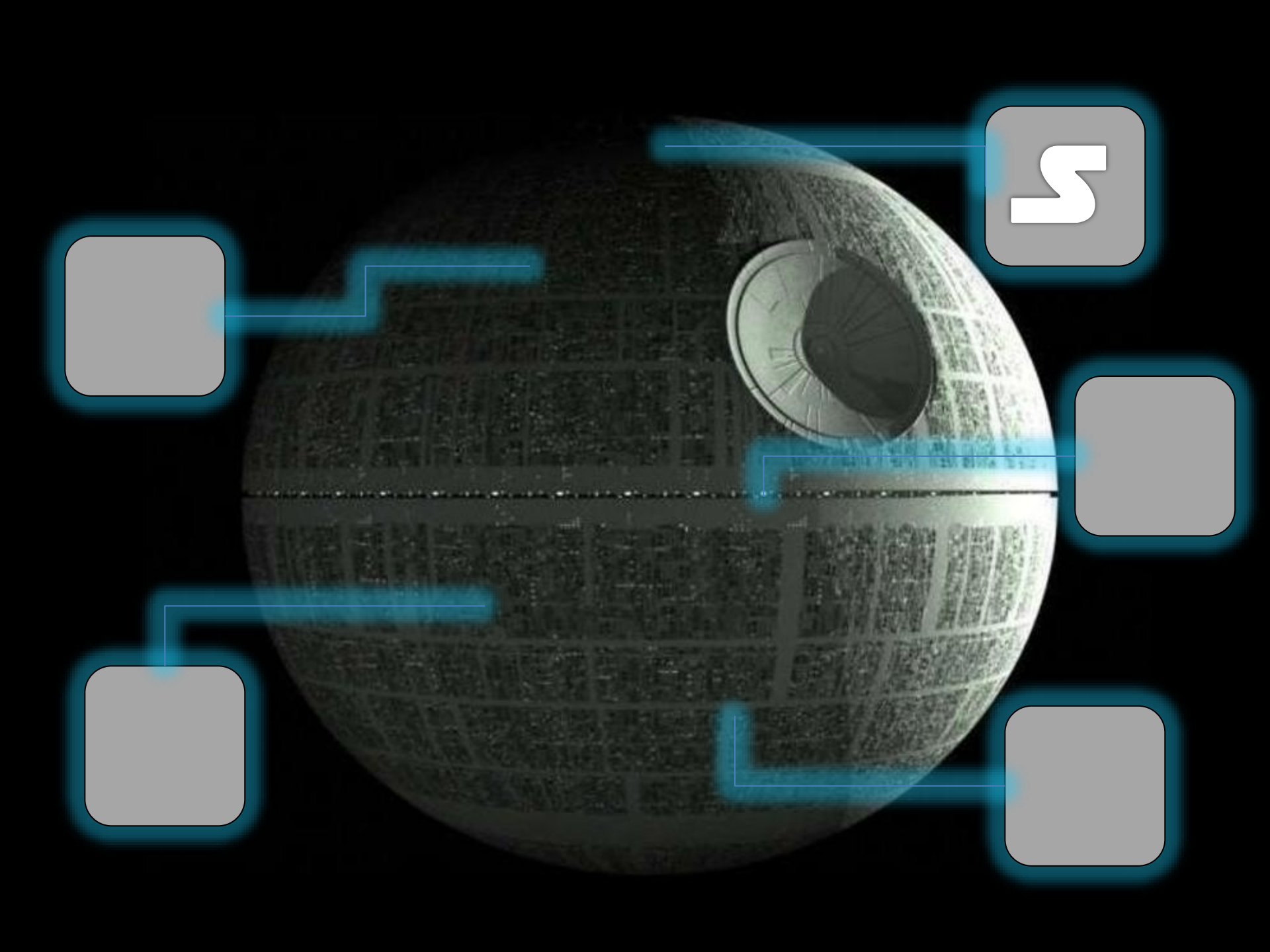
```java
public class ControlPanel {

    public void destroy(Planet planet) throws PlanetOutOfRangeException {
        // ...
    }

    public Connection call(Number number) throws NotEnoughCreditException {
        // ...
    }

}
```

```java
public class SuperLaser {

    public void destroy(Planet planet)
        throws PlanetOutOfRangeException {
        // ...
    }

}
```

```java
public class HolographicPhone {

    public Connection call(Number number)
        throws NotEnoughCreditException {
        // ...
    }

}
```

SINGLE RESPONSIBILITY PRINCIPLE


COHESION++
COUPLING--


S


```java
public class StormTrooperIdCard {
    private static final Rank RANK = Rank.StormTrooper;

    private String firstName;
    private String lastName;

}
```

```java
public class GunnerIdCard {
    private static final Rank RANK = Rank.Gunner;

    private String firstName;
    private String lastName;

    // ...
}
```

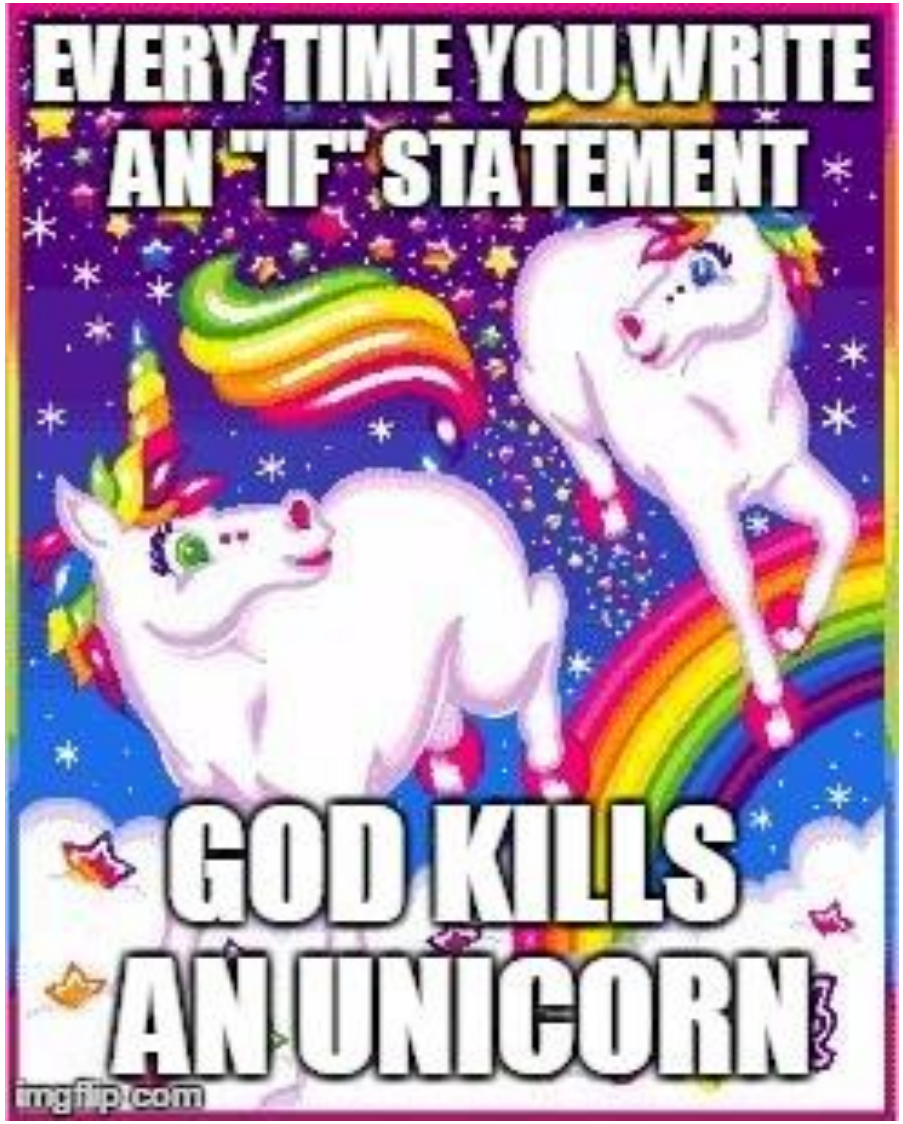
```java
public class PilotIdCard {
    private static final Rank RANK = Rank.Pilot;

    private String firstName;
    private String lastName;

    // ...
}
```

ID CARD

```
public class Canteen {
```

```
public class LockerRoom {

    public Armor giveArmor(Object idCard) throws UnknownRankException {
        if(idCard instanceof StormTrooperIdCard)
            return new StormTrooperArmor();
        else if(idCard instanceof
            return new GunnerArmor
        else if(idCard instanceof
            return ne
        else
            throw ne
    }

    // ...
}
```

```
public class CommandorIdCard {
    private static final Rank RANK = Rank.Commandor;
    private String firstName;
    private String lastName;
    // ...
}
```


EVERY TIME YOU WRITE AN "IF" STATEMENT
GOD KILLS AN UNICORN
imgflip.com

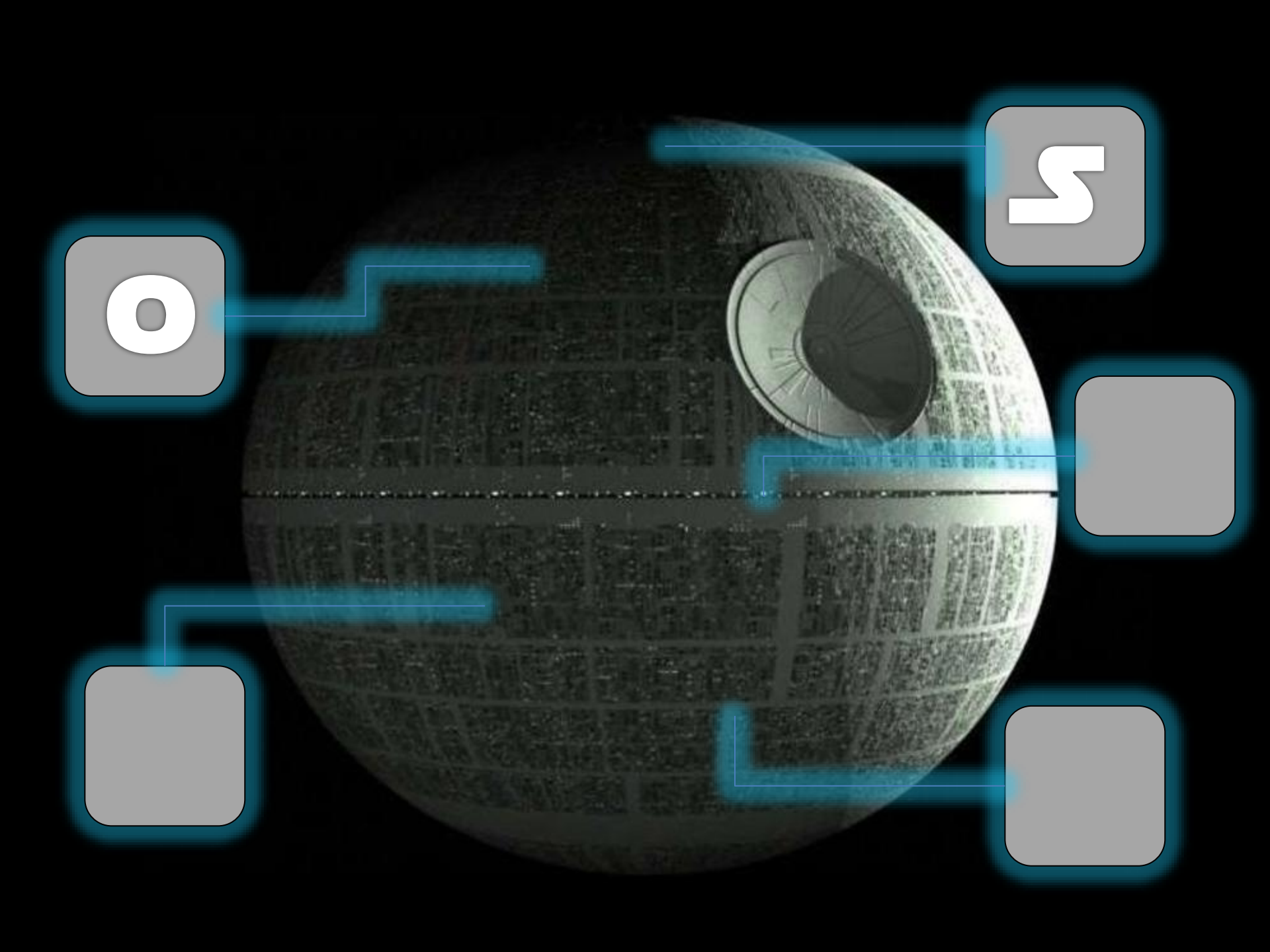

SAVE THE
UNICORN
FOUNDATION
Membership Card

```java
public abstract class IdCard {
    protected final Rank rank;

    protected String firstName;
    protected String lastName;

    protected IdCard(Rank rank) {
        this.rank = rank;
    }

    public abstract Credit giveSalary();
    public abstract Portion giveAllowedPortion();
    public abstract Armor giveArmor();

    // ...
}
```

```java
public class StormTrooperIdCard extends IdCard {
    public StormTrooperIdCard() {
        super(Rank.StormTrooper);
    }

    public Credit giveSalary() {
        return new Credit(2000);
    }

    public Portion giveAllowedPortion() {
        return new StormTrooperPortion();
    }

    public Armor giveArmor() {
        return new StormTrooperArmor();
    }

    // ...
}
```

```java
public class CreditMachine {

    public Credit payout(IdCard card) {
        return card.giveSalary();
    }

    // ...
}
```

OPEN – CLOSE PRINCIPLE


S
O

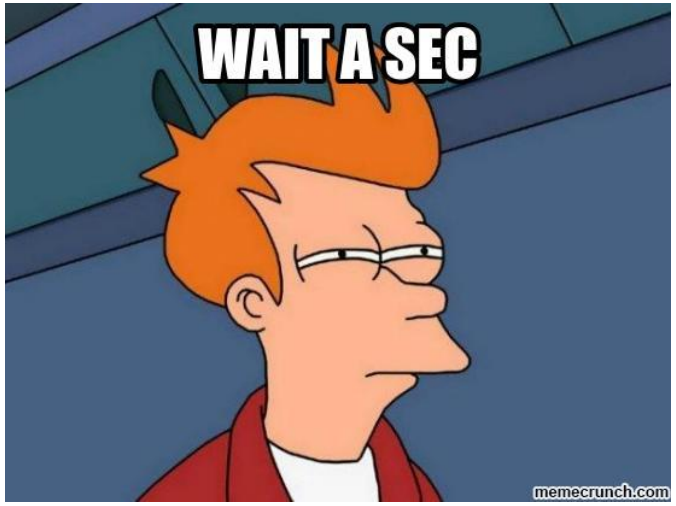

CHARACTER DETAILS
482K
ATTACKS & ABILITIES
Hindering Sh
2
Level 1 (Basic)
The Order Relentl
1
Level 2 (Special)
Return Fire
Level 3 (Unique)
First Order Stormtrooper
Find
Find
POWER: 947
Dark Side
First Order tank that uses Advantage to disrupt enemies
First Order, Human, Tank
SHARDS: 4 / 25
Collect additional Shards to promote to Three Star
FIND
BUY DATA CARDS
Gear Lvl II
Gear Lvl III
Fill all gear slots to upgrade
Lvl 35 10200/10925 XP
TRAIN
1


STORMTROOPER SELFIE
SURFINGTROOPER.COM

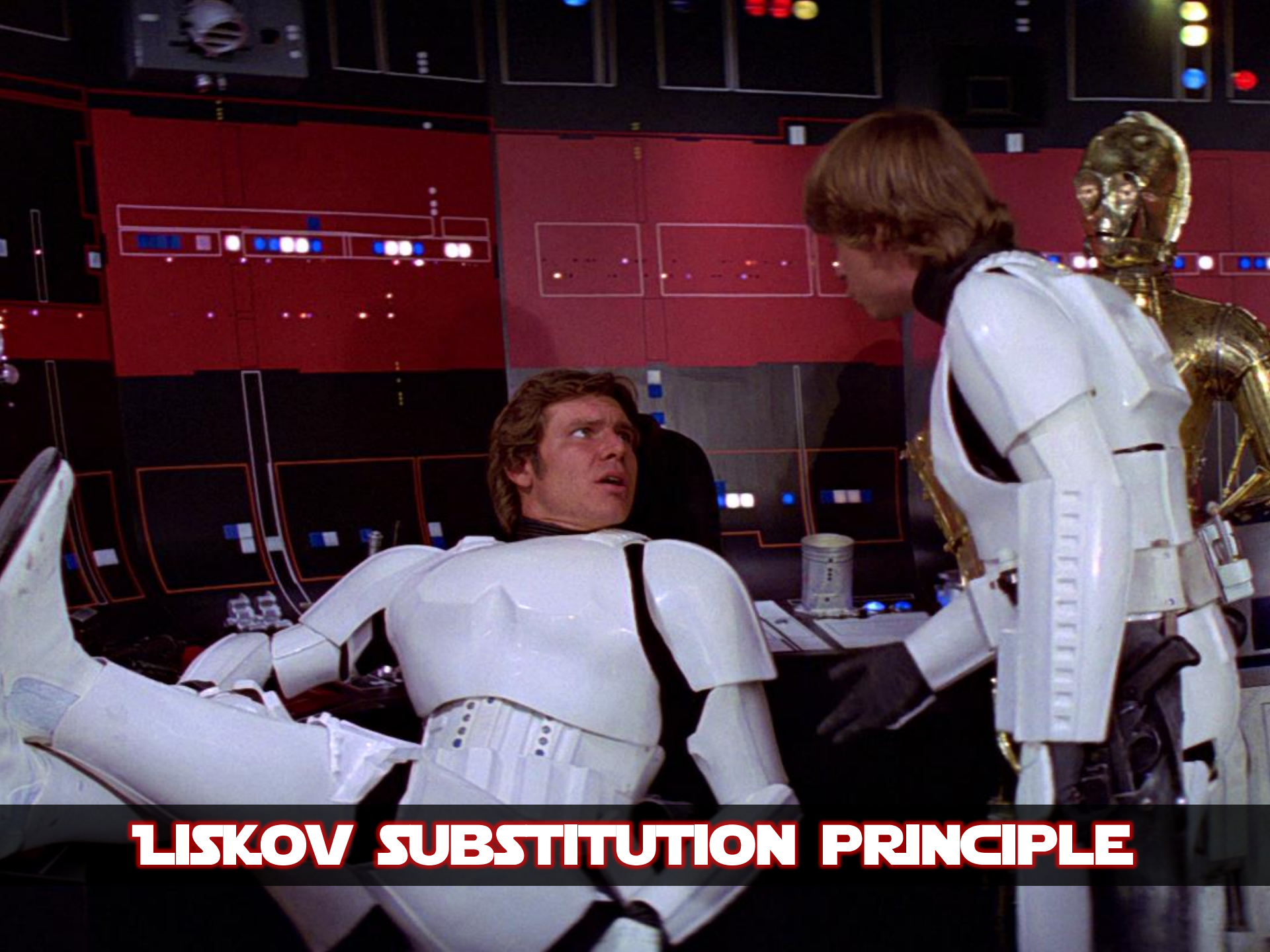
```java
public class Stormtrooper {

    private Weapon weapon;

    public void shoot(Target target) {
        weapon.decreaseEnergy();
    }

    // ...
}
```

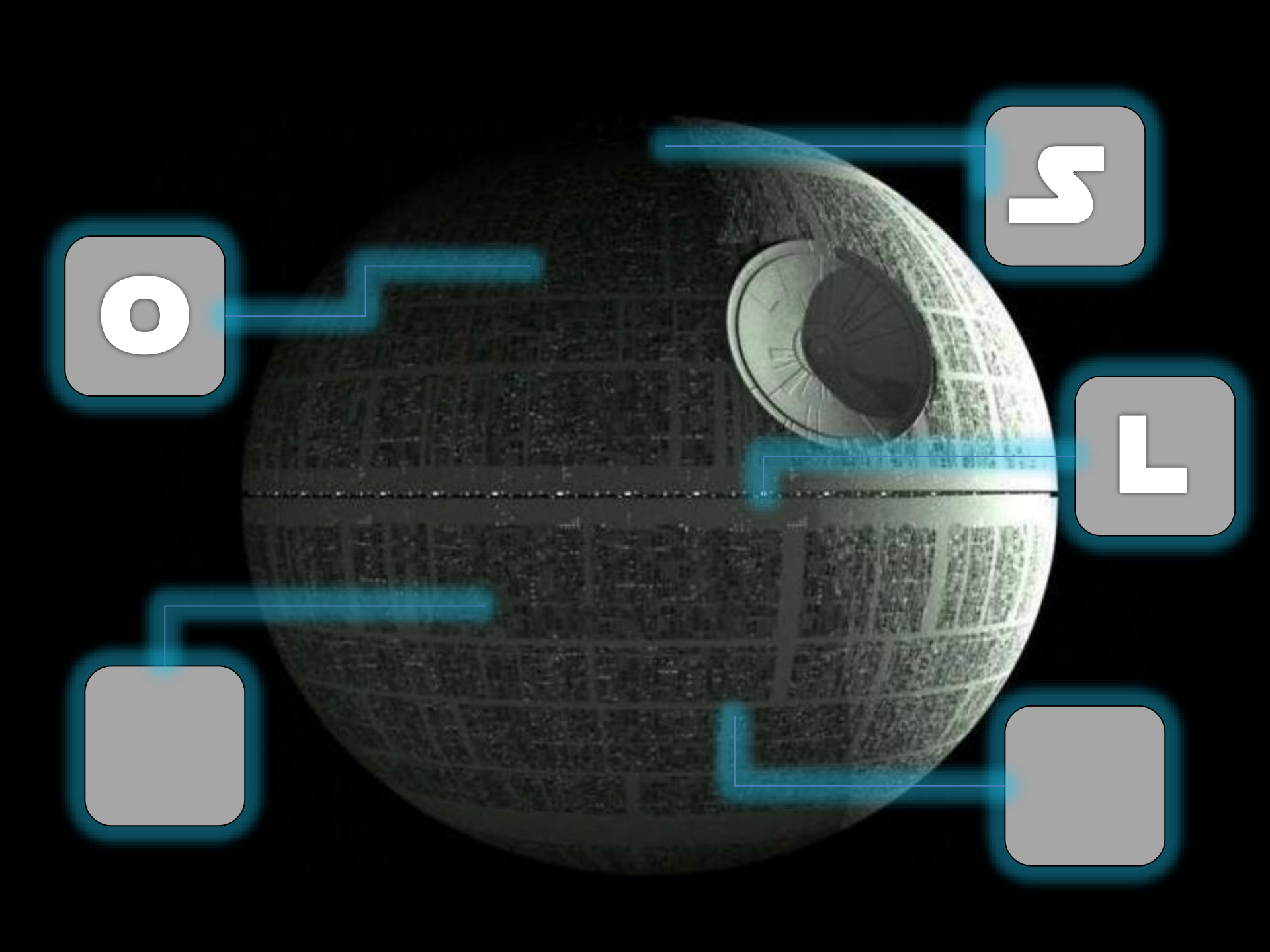
```java
public class StormtrooperTest {

    private static final MAX_HEALTH = 100;

    private Stormtrooper trooper = new Stormtrooper();

    @Test
    public void shoul                    nShootingTarget() {
        // given
        Target targe                     TH);

        // when
        trooper.shoot(

        // then
        assertEquals(MAX_HEALTH, target.getHealth());
    }

    // ...
}
```

```java
public class HanSolo extends Stormtrooper {

    private Weapon weapon;

    @Override
    public void shoot(Target target) {
        weapon.decreaseEnergy();
        target.hit(weapon.getDamage());
    }

    // ...
}
```




```java
public class StormtrooperTest {

    private static final MAX_HEALTH = 100;

    private Stormtro                nSolo();

    @Test
    public void                     otingTarget() {
        // give
        Target

        // when
        trooper.

        // then
        assertEquals(                   getHealth());
    }

    // ...
}
```


LISKOV SUBSTITUTION PRINCIPLE


S
O
L

TIE FIGHTER

ATAT WALKER

```java
public interface Movable {

    void fly(Distance distance);
    void walk(Distance distance);

    // ...
}
```

```java
public class TieFighter implements Movable {

    private Position position;

    public void fly(Distance distance) {
        position.move(distance);
    }

    public void walk(Distance distance) {
        throw new NotImplementedException("Me have no legs");
    }
}
```

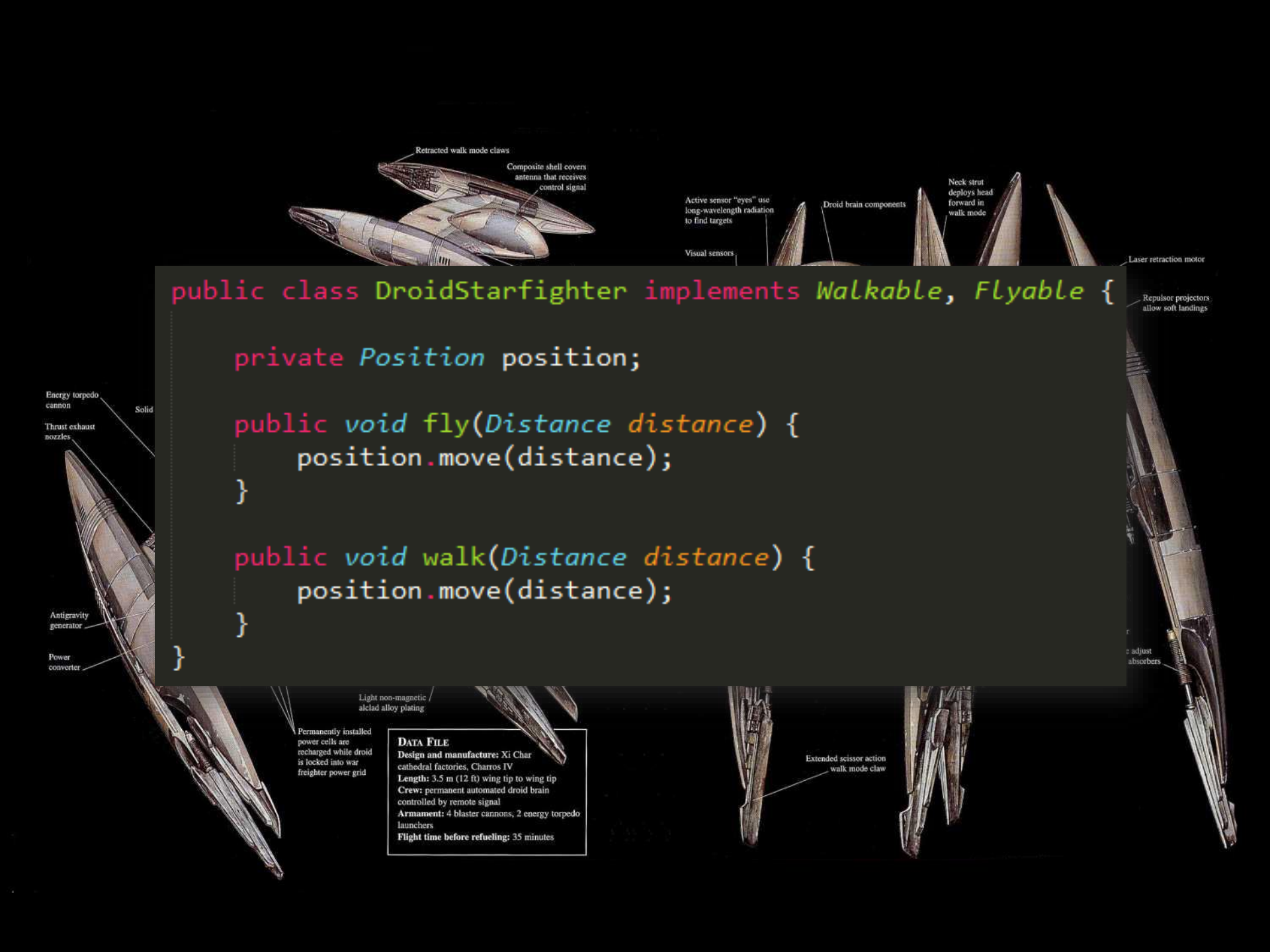
```java
public class AtAtWalker implements Movable {

    private Position position;

    public void fly(Distance distance) {
        throw new NotImplementedException("Me have no wings");
    }

    public void walk(Distance distance) {
        position.move(distance);
    }
}
```

```java
public interface Movable {

    void fly(Distance distance);
    void walk(Distance distance);

    // ...
}
```

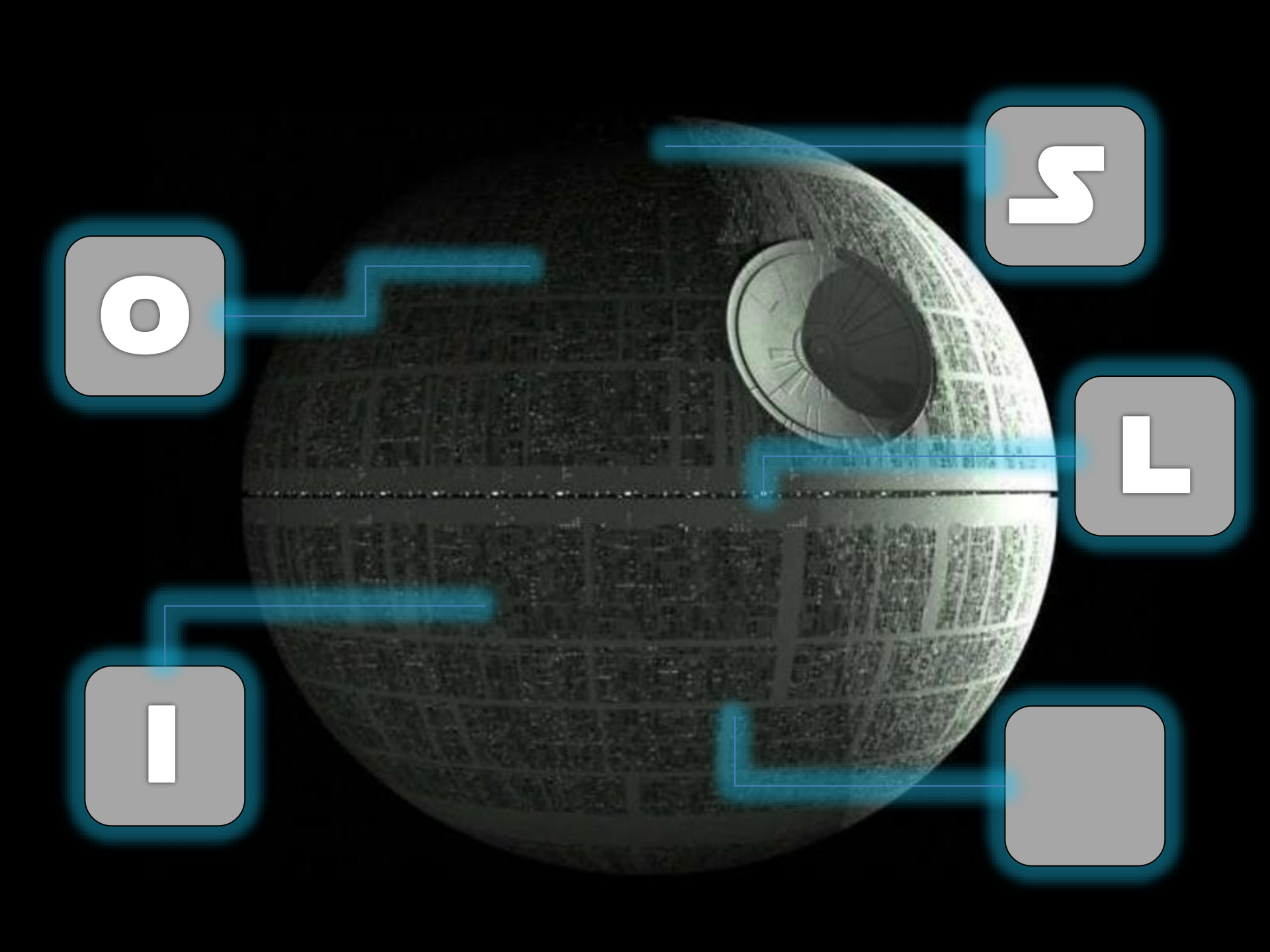
```java
public interface Walkable {

    void walk(Distance distance);

    // ...
}
```

```java
public interface Flyable {

    void fly(Distance distance);

    // ...
}
```

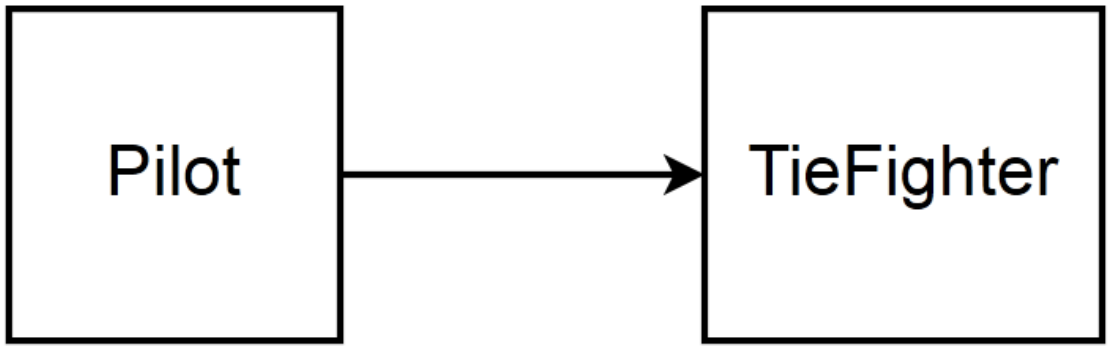
INTERFACE SEGREGATION PRINCIPLE

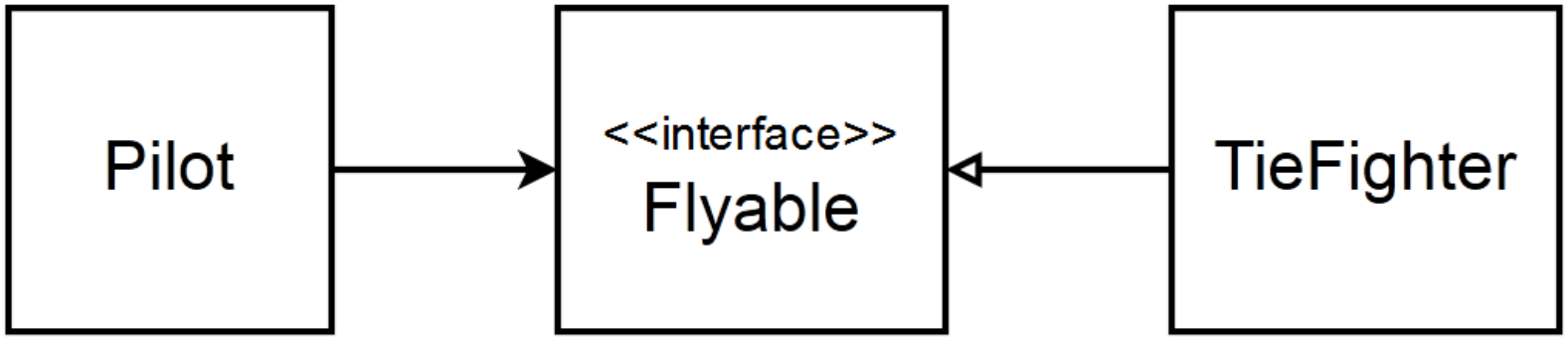



```java
public class DroidStarfighter implements Walkable, Flyable {

    private Position position;

    public void fly(Distance distance) {
        position.move(distance);
    }

    public void walk(Distance distance) {
        position.move(distance);
    }
}
```


S
O
L
I

```java
public class Pilot {
    private TieFighter tieFighter;

    public void fly(Destination destination) {
        while(!tieFighter.getPosition().equals(destination)) {
            tieFighter.fly(calculateDistance());
        }
    }

    // ...
}
```

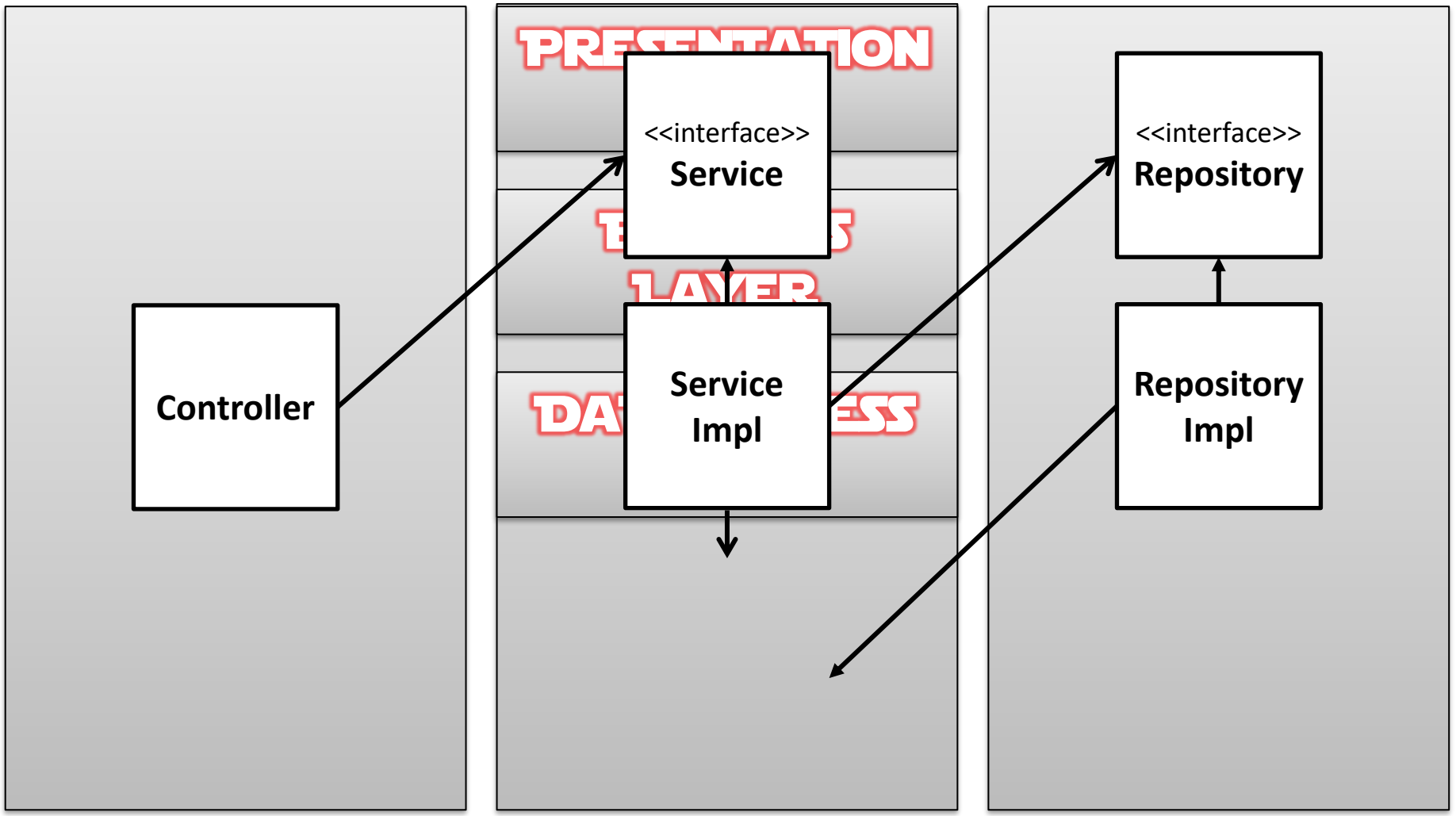

Pilot
TieFighter

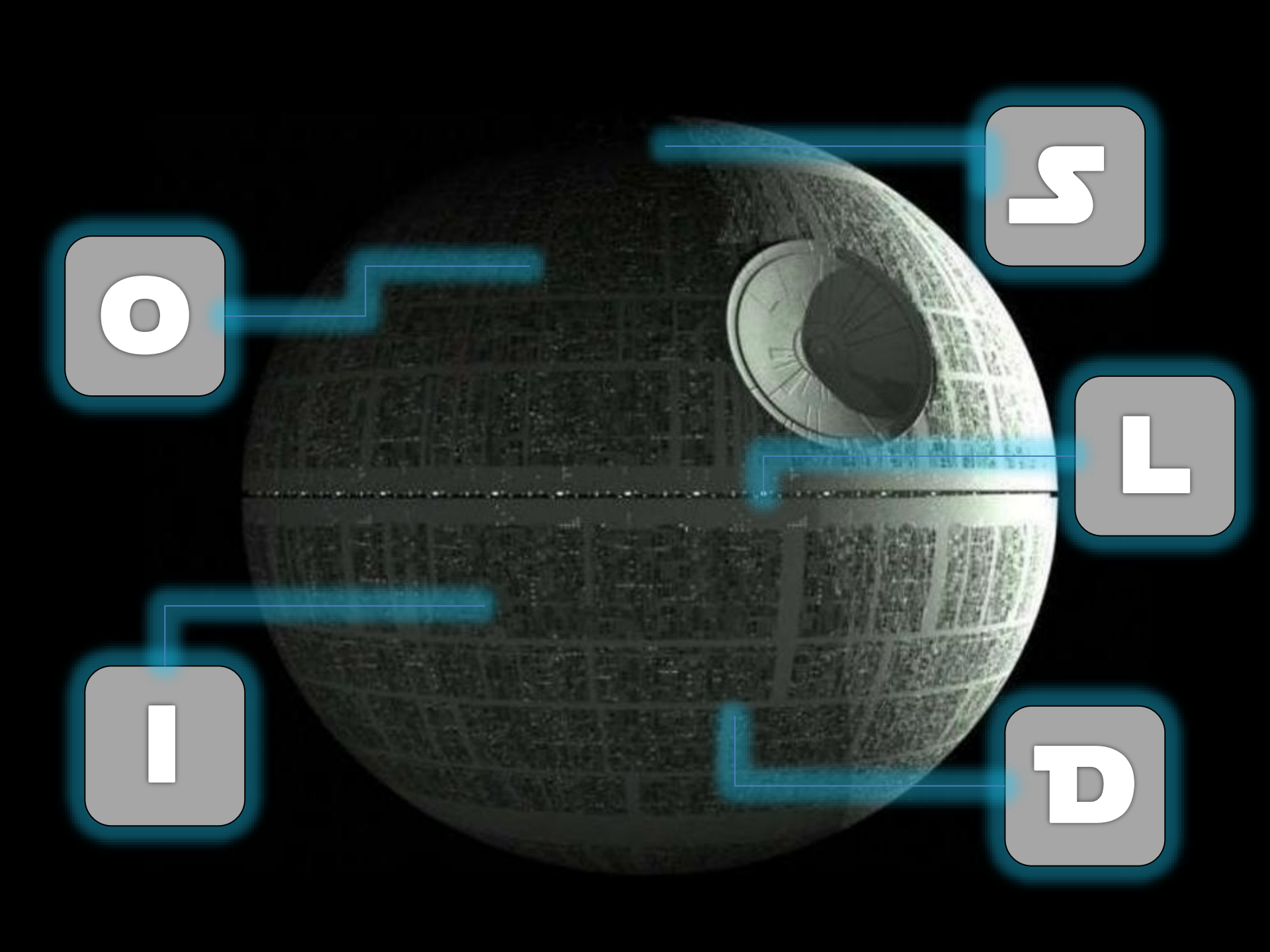

Pilot
<<interface>>
Flyable
TieFighter

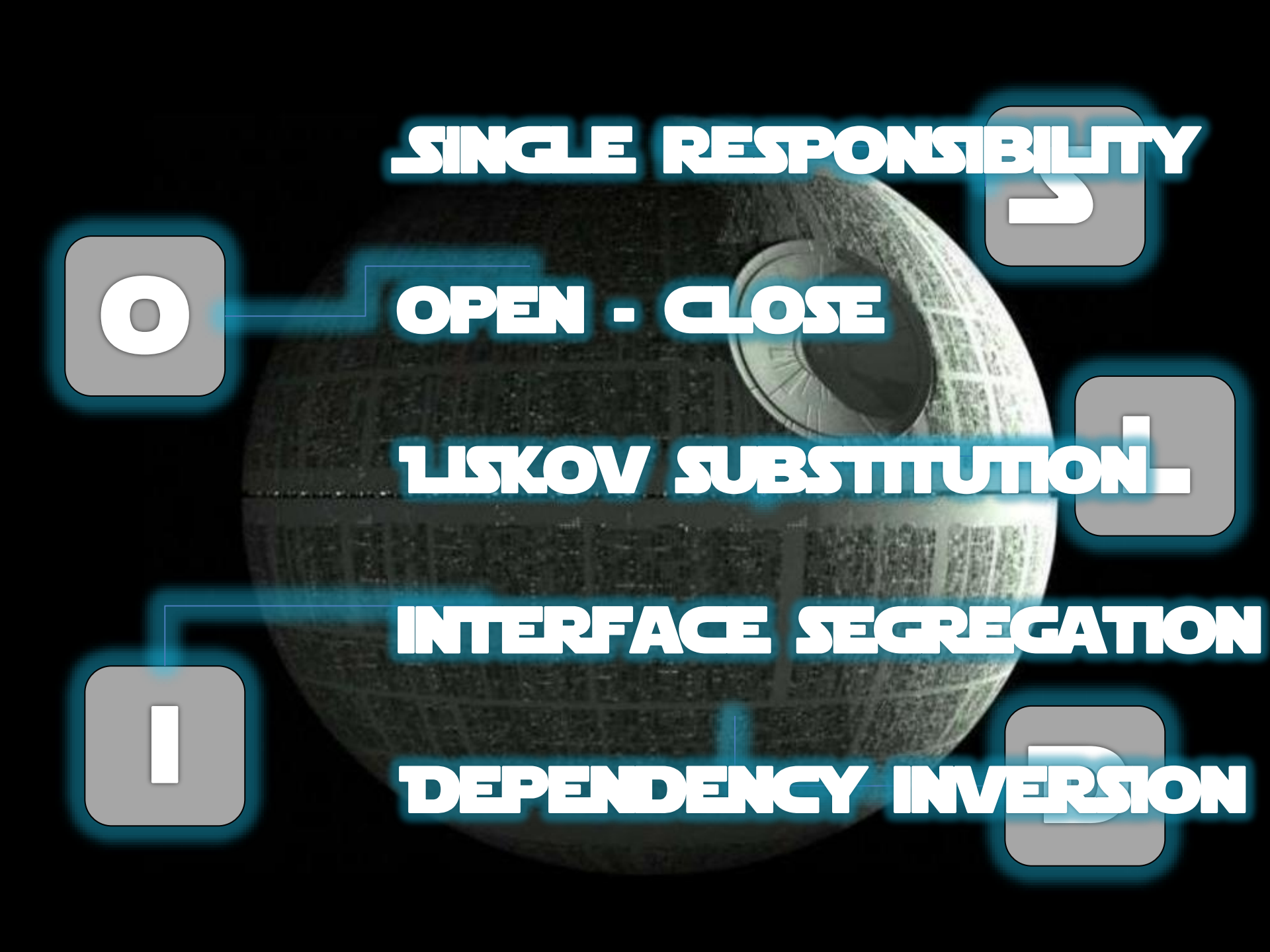
```
public class Pilot {
    private TieFighter tieFighter;

    public void fly(Destination destination) {
        while(!tieFighter.getPosition().equals(destination)) {
            tieFighter.fly(calculateDistance());
        }
    }

    // ...
}
```

```
public class Pilot {
    private Flyable flyingVehicle;

    public void fly(Destination destination) {
        while(!flyingVehicle.getPosition().equals(destination)) {
            flyingVehicle.fly(calculateDistance());
        }
    }

    // ...
}
```

DEPENDENCY INVERSION PRINCIPLE


PRESENTATION
<<interface>>
Service
Controller
Service
Impl
<<interface>>
Repository
Repository
Impl


S
O
L
I
D


S
SINGLE RESPONSIBILITY
O
OPEN - CLOSE
L
LISKOV SUBSTITUTION
I
INTERFACE SEGREGATION
D
DEPENDENCY INVERSION

SOLID

KISS

GoF

PSR

GRASP

YAGNI

DRY


Robert C. Martin Series
Clean Code
A Handbook of Agile Software Craftsmanship
Foreword by James O. Coplien
Robert C. Martin Series
The Clean Code
A Code of Conduct for Professional Programmers
Robert C. Martin
REFACTORING
IMPROVING THE DESIGN OF EXISTING CODE
MARTIN FOWLER
Foreword by Erich Gamma
Object Technology International, Inc.
Joshua Bloch
Effective Java
Second Edition
The Java Series
...from the Source
Java

MUCH TO LEARN, YOU STILL HAVE


THE
END