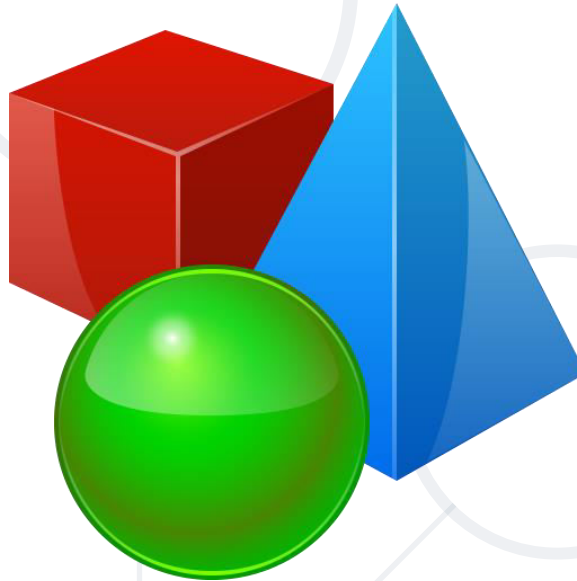


Objects and Classes

Using Objects and Classes
Defining Simple Classes



SoftUni Team
Technical Trainers



**Software
University**



**SoftUni
Foundation**



Software University

<http://softuni.bg>

1. Objects
2. Classes
3. Built in Classes
4. Defining Simple Classes
 - Fields
 - Constructors
 - Methods



Have a Question?

sli.do

#tech-java



Objects and Classes

What Is an Object? What Is a Class?

Objects

- An **object** holds a set of named values
 - E.g. **birthday** object holds day, month and year
 - Creating a birthday object:



Birthday

Day = 27

Month = 11

Year = 1996

Object
name

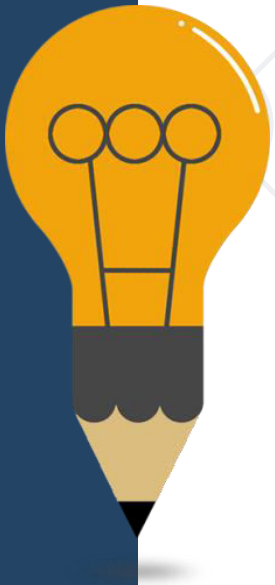
Object
properties

```
LocalDate birthday =  
    LocalDate.of(2018, 5, 5);  
System.out.println(birthday);
```

Create a new object of
type LocalDate

Classes

- In programming, **classes** provide the structure for **objects**
 - Act as **template** for **objects** of the same type
- Classes define:
 - Fields (**private variables**), e.g. **day**, **month**, **year**
 - Data, e.g. **getDay**, **setMonth**, **getYear**
 - Actions (**behavior**), e.g. **plusDays(count)**, **subtract(date)**
- One class may have many instances (objects)
 - Sample class: **LocalDate**
 - Sample objects: **PeterBirthday**, **MariaBirthday**



Objects – Instances of Classes

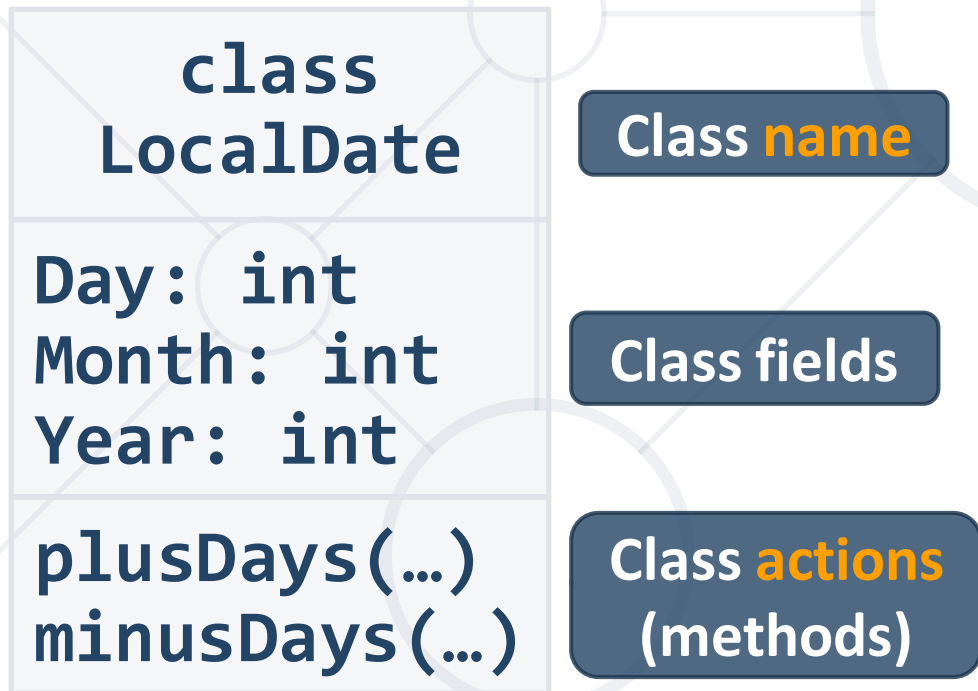
- Creating the object of a defined class is called **instantiation**
- The **instance** is the object itself, which is created runtime
- All instances have common **behaviour**



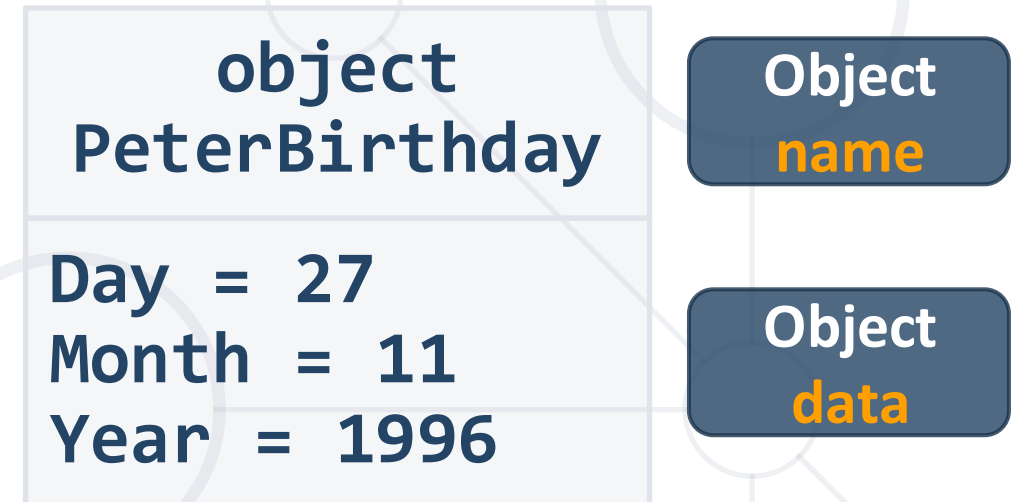
```
LocalDate date1 = LocalDate.of(2018, 5, 5);  
LocalDate date2 = LocalDate.of(2016, 3, 5);  
LocalDate date3 = LocalDate.of(2013, 3, 2);
```

Classes vs. Objects

- Classes provide structure for creating objects



- An object is a single instance of a class





`Math.max()`

Using the Built-In API Classes

Math, Random, BigInteger, ...

- Java provides ready-to-use classes
 - Organized inside Packages like:
`java.util.Scanner`, `java.util.List`, etc.
- Using static class members:

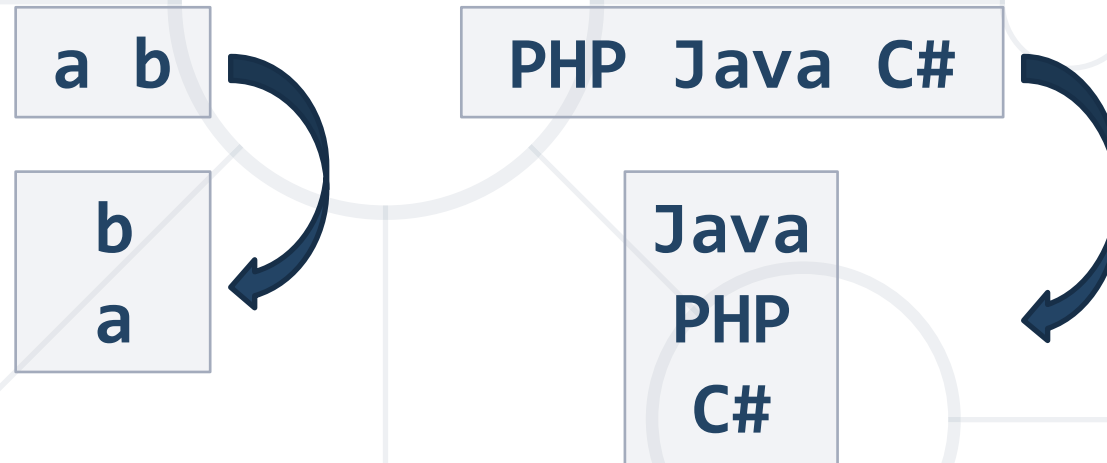
```
LocalDateTime today = LocalDateTime.now();  
double cosine = Math.cos(Math.PI);
```

- Using non-static Java classes

```
Random rnd = new Random();  
int randomNumber = rnd.nextInt(99);
```

Problem: Randomize Words

- You are given a list of words
 - Randomize their order and print each word at a separate line



**Note: the output is a sample.
It should always be different!**

Check your solution here: <https://judge.softuni.bg/Contests/1319/>

Solution: Randomize Words

```
Scanner sc = new Scanner(System.in);
String[] words = sc.nextLine().split(" ");
Random rnd = new Random();
for (int pos1 = 0; pos1 < words.length; pos1++) {
    int pos2 = rnd.nextInt(words.length);
    //TODO: Swap words[pos1] with words[pos2]
}
System.out.println(String.join(System.lineSeparator(), words));
```

Check your solution here: <https://judge.softuni.bg/Contests/1319/>

Problem: Big Factorial

- Calculate $n!$ (n factorial) for very big n (e.g. 1000)

5 → 120 10 → 3628800 12 → 479001600

50 → 3041409320171337804361260816606476884437764156
8960512000000000000

88 → 1854826422573984391147968456455462843802209689
4939934668442158098688956218402819931910014124
48045018284166335168512000000000000000000000000000

Check your solution here: <https://judge.softuni.bg/Contests/1319/>

Solution: Big Factorial

```
import java.math.BigInteger;
...
int n = Integer.parseInt(sc.nextLine());
BigInteger f =
    new BigInteger(String.valueOf(1));
for (int i = 2; i <= n; i++)
    f = f.multiply(BigInteger.valueOf(
        Integer.parseInt(String.valueOf(i))));
System.out.println(f);
```

Use the
java.math.BigInteger

N!

Check your solution here: <https://judge.softuni.bg/Contests/1319/>



Defining Classes

Creating Custom Classes

Defining Simple Classes

- Specification of a given type of objects from the real-world
- **Classes** provide structure for describing and creating objects



Keyword



Class **name**

```
class Dice {  
    ...  
}
```

Class **body**

Naming Classes

- Use PascalCase naming
- Use descriptive nouns
- Avoid abbreviations (except widely known, e.g. URL, HTTP, etc.)



```
class Dice { ... }  
class BankAccount { ... }  
class IntegerCalculator { ... }
```



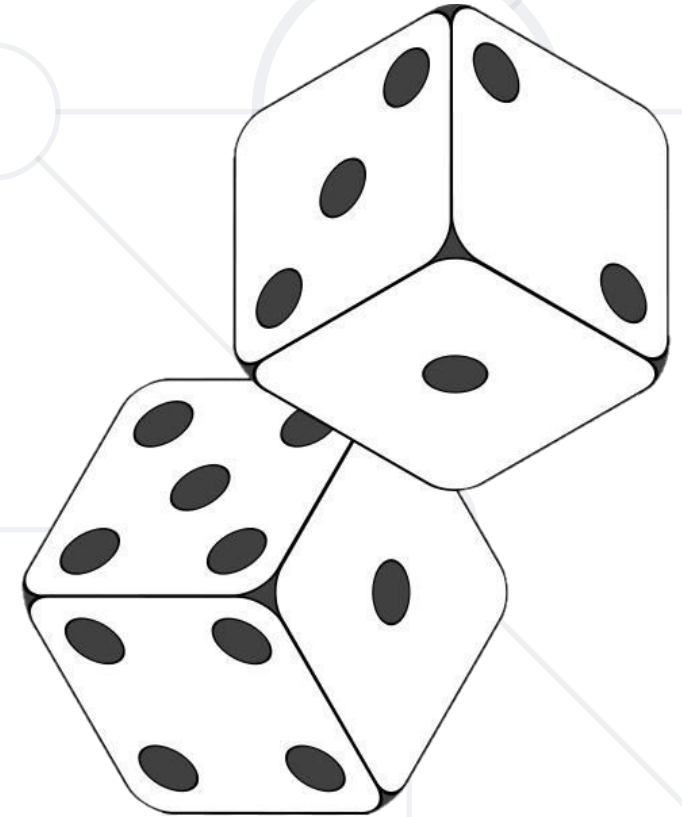
```
class TPMF { ... }  
class bankaccount { ... }  
class intcalc { ... }
```

- Class is made up of **state** and **behavior**
- Fields **store values**
- Methods **describe behaviour**

```
class Dice {  
    private int sides;  
    private String type;  
    public void roll() { ... }  
}
```

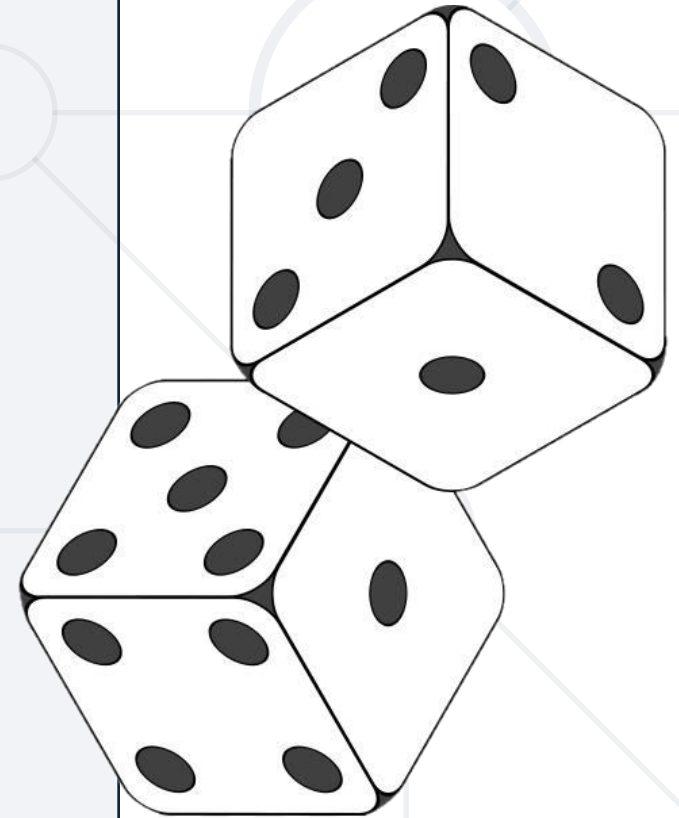
Fields

Method



```
class Dice {  
    public int getSides() { return this.sides; }  
    public void setSides(int sides) {  
        this.sides = sides;  
    }  
    public String getType() { return this.type; }  
    public void setType(String type) {  
        this.type = type;  
    }  
}
```

Properties



Creating an Object

- A class can have many instances (objects)

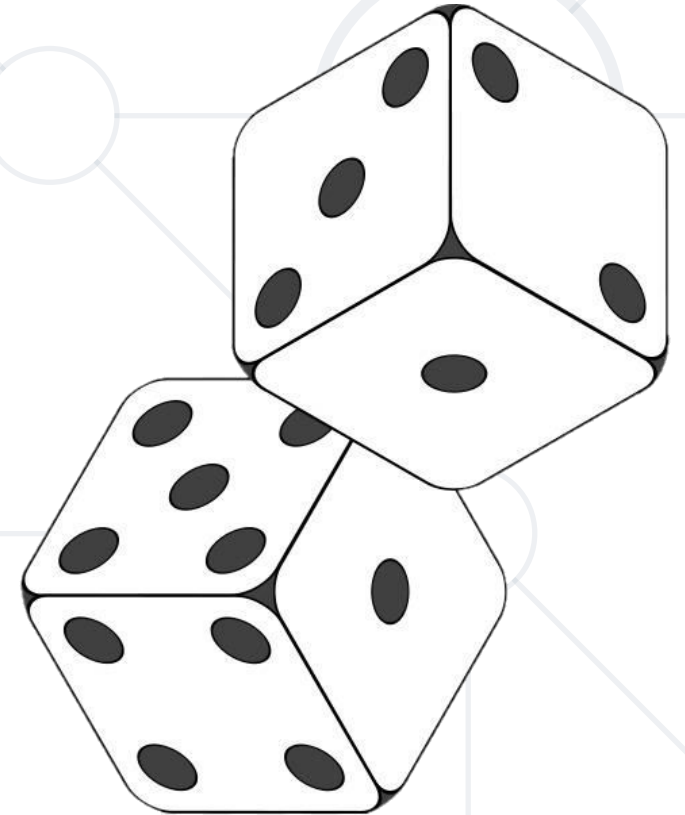
```
class Program {  
    public static void main(String[] args) {  
        Dice diceD6 = new Dice();  
        Dice diceD8 = new Dice();  
    }  
}
```

Use the **new**
keyword

Variable stores a
reference

- Store executable code (algorithm)

```
class Dice {  
    public int sides;  
    public int roll() {  
        Random rnd = new Random();  
        int sides = rnd.nextInt(this.sides + 1);  
        return sides;  
    }  
}
```

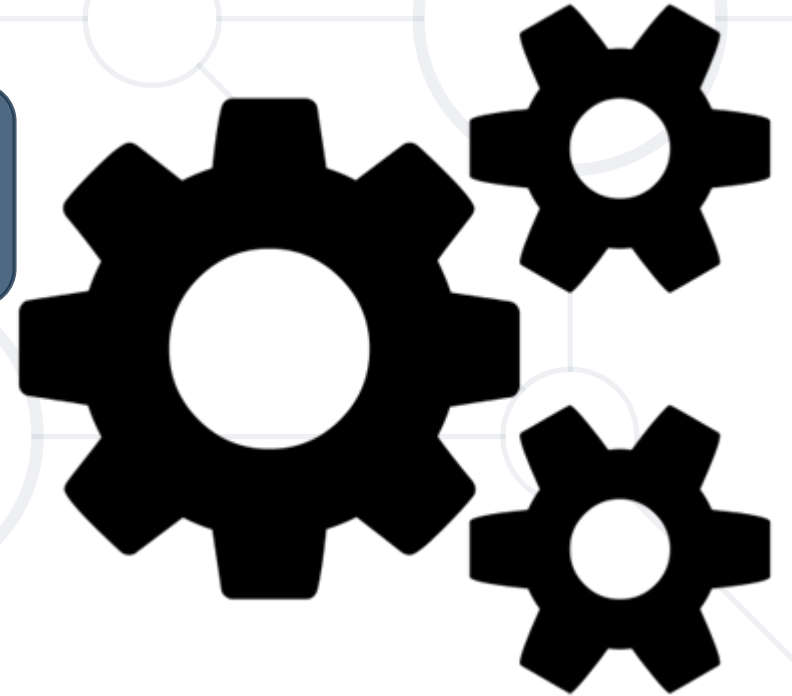


- Special methods, executed during object creation

```
class Dice {  
    public int sides;  
    public Dice() {  
        this.sides = 6;  
    }  
}
```

Overloading default
constructor

Constructor name is
the same as the name
of the class



Constructors (2)

- You can have multiple constructors in the same class

```
class Dice {  
    public int sides;  
    public Dice() { }  
    public Dice(int sides) {  
        this.sides = sides;  
    }  
}
```

```
class StartUp {  
    public static void main(String[] args) {  
        Dice dice1 = new Dice();  
        Dice dice2 = new Dice(7);  
    }  
}
```

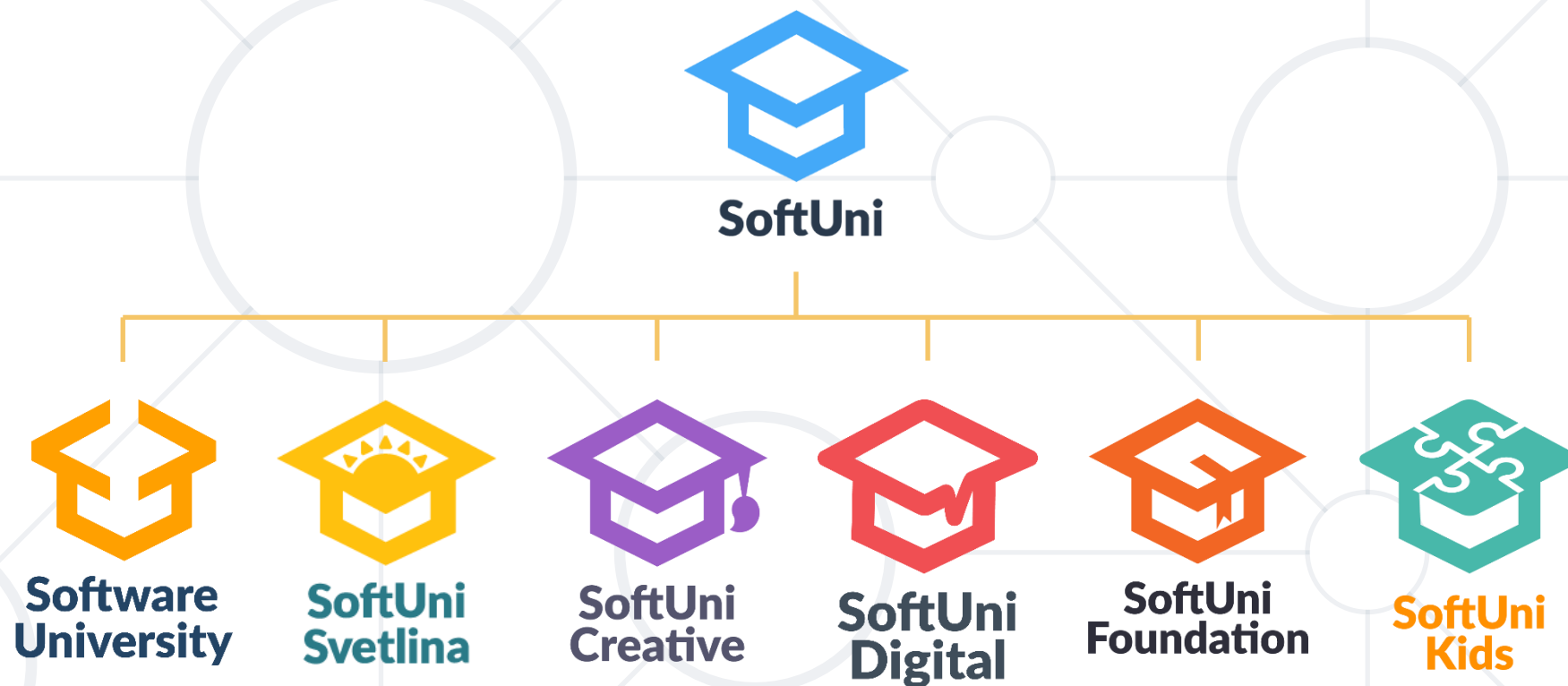


Live Exercises

- Classes define templates for object
 - **Fields**
 - **Constructors**
 - **Properties**
 - **Methods**
- Objects
 - holds a set of **named values**
 - **Instance** of a class



Questions?



SoftUni Diamond Partners



XSsoftware



SBTech
we know sports



telenor



SoftwareGroup
doing it right

NETPEAK



SmartIT



Postbank

Решения за твоето утре

**SUPER
HOSTING
.BG**

INDEAVR

Serving the high achievers



INFRAGISTICS®

LIEBHERR



aeternity



SoftUni Organizational Partners



OneBit
SOFTWARE



WORLD
OF
MYTHS

Trainings @ Software University (SoftUni)

- Software University – High-Quality Education and Employment Opportunities
 - softuni.bg
- Software University Foundation
 - <http://softuni.foundation/>
- Software University @ Facebook
 - facebook.com/SoftwareUniversity
- Software University Forums
 - forum.softuni.bg



- This course (slides, examples, demos, videos, homework, etc.) is licensed under the "Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International" license

