

# Programming in C++ - Primer

## Lesson 6 - Advanced Objects

Jakub 'Eremiell' Marek  
<marekj14@fel.cvut.cz>

Silicon Hill C++ Academy

2013/11/25

## C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

**1** Revision

**2** Encapsulation

**3** Inheritance

**4** Operators

**5** Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

# Containers and Structs

C++ Primer

Jakub Marek

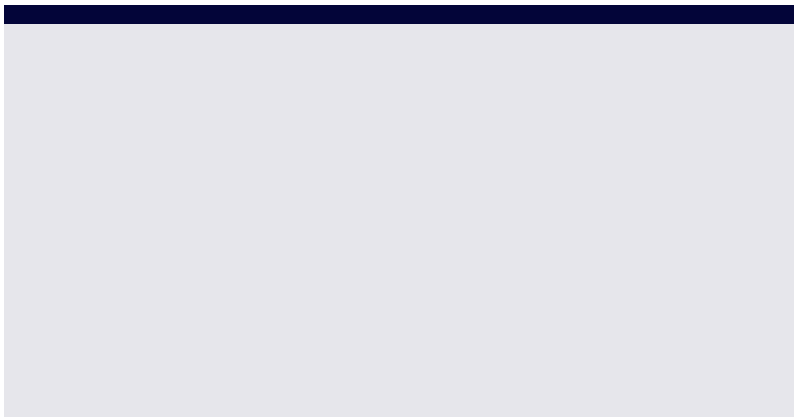
Revision

Encapsulation

Inheritance

Operators

Friendship



# Containers and Structs

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

### ■ Strings

# Containers and Structs

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

- Strings
- Vectors

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

- Strings
- Vectors
- and many others. . .



C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

- Strings
- Vectors
- and many others. . .

## Structs

# Containers and Structs

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

- Strings
- Vectors
- and many others...

## Structs

- aggregate data types

# Containers and Structs

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

- Strings
- Vectors
- and many others. . .

## Structs

- aggregate data types
- struct

# Containers and Structs

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Containers

- Strings
- Vectors
- and many others...

## Structs

- aggregate data types
- struct
- union

# Objects

C++ Primer

Jakub Marek

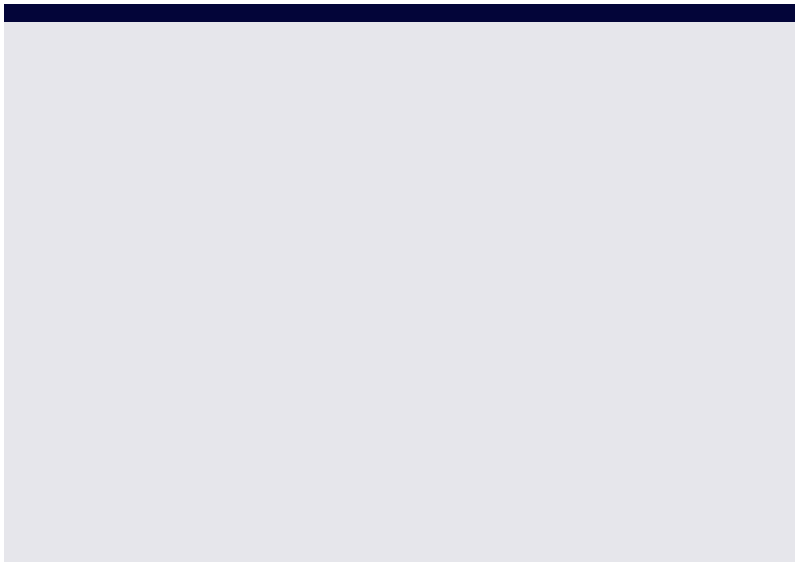
Revision

Encapsulation

Inheritance

Operators

Friendship



C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

- header file

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

- header file
- included wherever we use it



C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

- source file

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

- source file
- includes the header file

# Objects

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

- source file
- includes the header file
- is usually the only include

# Objects

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

- source file
- includes the header file
- is usually the only include

## Access rights

# Objects

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

- source file
- includes the header file
- is usually the only include

## Access rights

- public

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

- source file
- includes the header file
- is usually the only include

## Access rights

- public
- protected



# Objects

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Declaration

- header file
- included wherever we use it
- includes whatever is needed

## Definition

- source file
- includes the header file
- is usually the only include

## Access rights

- public
- protected
- private

# Encapsulation

C++ Primer

Jakub Marek

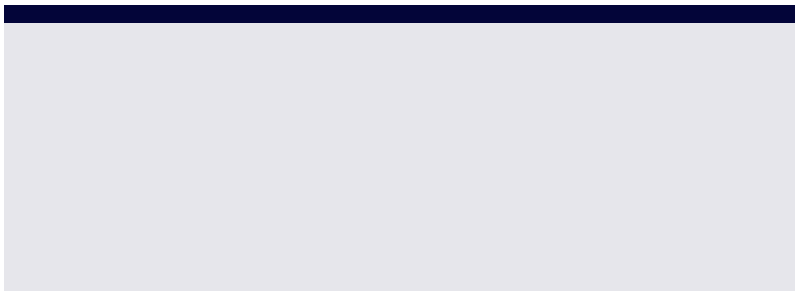
Revision

Encapsulation

Inheritance

Operators

Friendship



# Encapsulation

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

black boxes

# Encapsulation

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

black boxes

noone should care, how it works inside

# Encapsulation

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

black boxes  
noone should care, how it works inside  
providing interface

# Encapsulation

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

black boxes

no one should care, how it works inside

providing interface

changing interface might break other code

# Encapsulation

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

black boxes

noone should care, how it works inside

providing interface

changing interface might break other code

changing internals shouldn't

# Encapsulation

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

black boxes

noone should care, how it works inside

providing interface

changing interface might break other code

changing internals shouldn't

getters/setters guard the invariant



# Encapsulation

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

black boxes

no one should care, how it works inside

providing interface

changing interface might break other code

changing internals shouldn't

getters/setters guard the invariant

user of your code only needs the contract, headers & object files

# Questions?

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

# Inheritance

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

# Inheritance

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

objects can have special cases

# Inheritance

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

objects can have special cases  
two ways of inheritance:

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

objects can have special cases  
two ways of inheritance:

- specialization

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

objects can have special cases  
two ways of inheritance:

- specialization
- generalization

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

objects can have special cases  
two ways of inheritance:

- specialization
- generalization

think use wise and data wise



C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

objects can have special cases  
two ways of inheritance:

- specialization
- generalization

think use wise and data wise

C++ knows multiple inheritance! (unlike Java)

# Polymorphism

C++ Primer

Jakub Marek

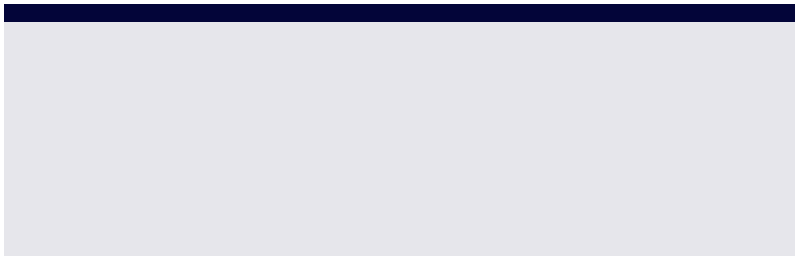
Revision

Encapsulation

**Inheritance**

Operators

Friendship



# Polymorphism

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

the parent pointer can contain the derived object

# Polymorphism

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

the parent pointer can contain the derived object  
because it's special case

# Polymorphism

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

the parent pointer can contain the derived object  
because it's special case  
you can overshadow parent's methods

# Polymorphism

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

the parent pointer can contain the derived object  
because it's special case  
you can overshadow parent's methods  
can still get the proper ones

# Polymorphism

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

the parent pointer can contain the derived object  
because it's special case  
you can overshadow parent's methods  
can still get the proper ones  
but you need virtual keyword

# Polymorphism

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

the parent pointer can contain the derived object  
because it's special case  
you can overshadow parent's methods  
can still get the proper ones  
but you need virtual keyword  
early/late binding



# Inheritance & Polymorphism

C++ Primer

Jakub Marek

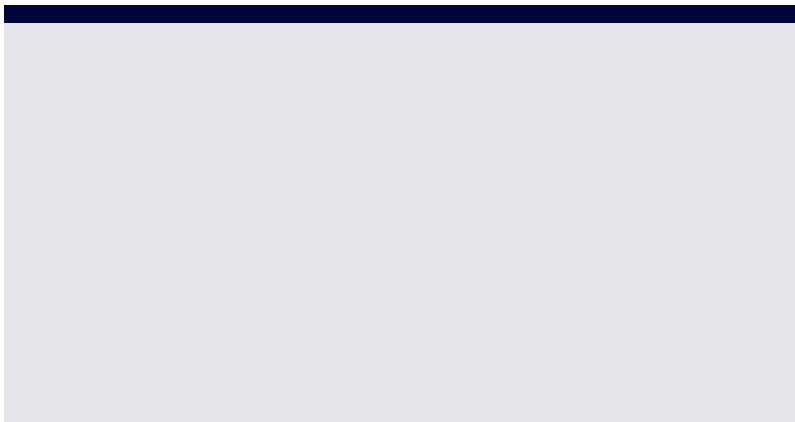
Revision

Encapsulation

**Inheritance**

Operators

Friendship



# Inheritance & Polymorphism

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

```
class Square {
public:
    Square(int a = 1) : a(a) {};
    int getA() {return a;};
    int setA(int a) {this->a = a; return this->a;};
    virtual int area() {return a * a;};
protected:
    int a;
}

class Rectangle : public Square {
public:
    Rectangle(int a = 1, int b = 1) : Square(a), b(b) {};
    int getB() {return b;};
    int setB(int b) {this->b = b; return this->b;};
    virtual int area() {return a * b;};
protected:
    int b;
}
```

# Questions?

C++ Primer

Jakub Marek

Revision

Encapsulation

**Inheritance**

Operators

Friendship

# Operators

C++ Primer

Jakub Marek

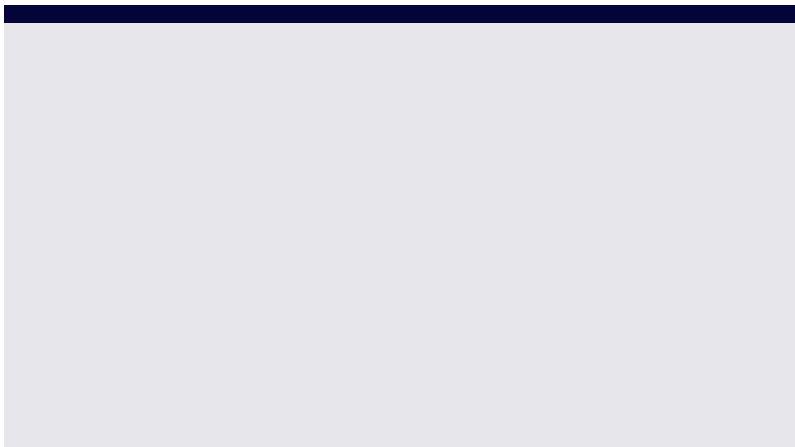
Revision

Encapsulation

Inheritance

**Operators**

Friendship



# Operators

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

operators are functions like any other

# Operators

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

operators are functions like any other  
so we can overshadow and overload them!

# Operators

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

operators are functions like any other  
so we can overshadow and overload them!

```
class Complex {  
    public:  
        Complex(int r = 0, int i = 0) : r(r), i(i) {};  
        Complex operator+(const Complex &c)  
            {return Complex(this->r + c.r, this->i + c.i);};  
        Complex& operator=(const Complex &c)  
            {this->r = c.r; this->i = c.i; return *this;};  
        Complex& operator+=(const Complex &c)  
            {*this = *this + c; return *this;};  
    protected:  
        int r, i;  
};
```

# Operators

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

operators are functions like any other  
so we can overshadow and overload them!

```
class Complex {  
    public:  
        Complex(int r = 0, int i = 0) : r(r), i(i) {};  
        Complex operator+(const Complex &c)  
            {return Complex(this->r + c.r, this->i + c.i);};  
        Complex& operator=(const Complex &c)  
            {this->r = c.r; this->i = c.i; return *this;};  
        Complex& operator+=(const Complex &c)  
            {*this = *this + c; return *this;};  
    protected:  
        int r, i;  
};
```

what to give and get from operators is a bit of alchemy



# Copying

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

# Copying

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

by copying an object, we get a shallow copy

# Copying

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

by copying an object, we get a shallow copy  
it only copies the pointers, we have

# Copying

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

by copying an object, we get a shallow copy  
it only copies the pointers, we have  
sometimes, we need a deep copy

# Copying

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

by copying an object, we get a shallow copy  
it only copies the pointers, we have  
sometimes, we need a deep copy  
we need to implement that!

# Copying

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

by copying an object, we get a shallow copy  
it only copies the pointers, we have  
sometimes, we need a deep copy  
we need to implement that!

- overloading operators

# Copying

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

by copying an object, we get a shallow copy  
it only copies the pointers, we have  
sometimes, we need a deep copy  
we need to implement that!

- overloading operators
- creating copy constructor

by copying an object, we get a shallow copy  
it only copies the pointers, we have  
sometimes, we need a deep copy  
we need to implement that!

- overloading operators
- creating copy constructor

```
class Complex {  
    public:  
        Complex(int r = 0, int i = 0) : r(r), i(i) {};  
        Complex(Complex c) : r(c.r), i(c.i) {};  
        Complex& operator=(const Complex &c)  
            {this->r = c.r; this->i = c.i; return *this;};  
    protected:  
        int r, i;  
}
```



by copying an object, we get a shallow copy  
it only copies the pointers, we have  
sometimes, we need a deep copy  
we need to implement that!

- overloading operators
- creating copy constructor

```
class Complex {  
    public:  
        Complex(int r = 0, int i = 0) : r(r), i(i) {};  
        Complex(Complex c) : r(c.r), i(c.i) {};  
        Complex& operator=(const Complex &c)  
            {this->r = c.r; this->i = c.i; return *this;};  
    protected:  
        int r, i;  
}
```

when copying member objects, we need to do that manually!

# Questions?

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

**Operators**

Friendship

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

Friendship Is Magic!

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

Friendship Is Magic!  
(puns intended ;) )

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

Friendship Is Magic!  
(puns intended ;) )  
allows to propagate class internals

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

Friendship Is Magic!

(puns intended ;) )

allows to propagate class internals  
which breaks encapsulation!

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

Friendship Is Magic!  
(puns intended ;) )  
allows to propagate class internals  
which breaks encapsulation!  
but sometimes, it's needed



# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Friendship Is Magic!

(puns intended ;) )

allows to propagate class internals

which breaks encapsulation!

but sometimes, it's needed

- with certain operators

## Friendship Is Magic!

(puns intended ;) )

allows to propagate class internals

which breaks encapsulation!

but sometimes, it's needed

- with certain operators
- multiple classes as parts of same abstraction

## Friendship Is Magic!

(puns intended ;) )

allows to propagate class internals

which breaks encapsulation!

but sometimes, it's needed

- with certain operators
- multiple classes as parts of same abstraction

you can friend the whole class or just one function

## Friendship Is Magic!

(puns intended ;) )

allows to propagate class internals

which breaks encapsulation!

but sometimes, it's needed

- with certain operators
- multiple classes as parts of same abstraction

you can friend the whole class or just one function

a friend is not a member of given class!

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

## Friendship Is Magic!

(puns intended ;) )

allows to propagate class internals

which breaks encapsulation!

but sometimes, it's needed

- with certain operators
- multiple classes as parts of same abstraction

you can friend the whole class or just one function

a friend is not a member of given class!

friendship isn't propagated through inheritance

## Friendship Is Magic!

(puns intended ;) )

allows to propagate class internals

which breaks encapsulation!

but sometimes, it's needed

- with certain operators
- multiple classes as parts of same abstraction

you can friend the whole class or just one function

a friend is not a member of given class!

friendship isn't propagated though inheritance

(parent's friend isn't child's friend)

# Friendship

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship

```
class a {  
    public:  
        a(int i) : i(i) {};  
        friend int showl(a instance);  
        int getl();  
    private:  
        int i;  
};  
  
int a::getl() {  
    return this->i;  
}  
  
int showl(a instance) {  
    return instance.i;  
}
```

# Questions?

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship



# Break!

C++ Primer

Jakub Marek

Revision

Encapsulation

Inheritance

Operators

Friendship