ESOP - Information Hiding

Assoc. Prof. Dr. Mathias Lux
ITEC / AAU

Encapsulation

In big software projects the globally available names (classes, fields and methods) need to be structured and organized

- We distinguish between public and hidden identifiers.

Example

```
public class ShipExample {
    // actual position of the ship
    private int positionX, positionY;
    // maximum number for x and y
    private int maxX = 320, maxY = 640;

    public ShipExample() {
        this.positionX = maxX/2;
        this.positionY = maxY/2;
    }

    public void moveShip(int offSetX, int offsetY) {
        positionX += offSetX;
        positionY += offsetY;
         // check for violation of maximum
        if (positionX > maxX)
            positionX = maxX;
        if (positionY > maxY)
            positionY = maxY;
    }
}
```

Encapsulation

- Clients can only access specified classes, fields and methods.
- A critical part cannot be accessed or overwritten from external sources.

Encapsulation

- Identifier from the specification of an abstract data type should be public.
- Identifier, that are only needed fro implementation purposes shoüuld be hidden.

All in all …

- Never put more out into the public than you actually need there.

Example: Too Public

```
Stack myStack = new Stack();
myStack.push(1);
myStack.push(2);
myStack.push(3);
myStack.top = 0; // 2 and 3 are „deleted“
int drei = myStack.pop();
```