



JavaScript 1
alert("Lektion 6");

Utbildare: Mahmud Al Hakim

NACKADEMIN

Lektionstillfällets mål

Mål med lektionen

- Lite mer om versionshantering
- Arbeta med funktioner
- Deklarera funktioner
- Anropa funktioner
- Parametrar
- Argument
- Anonyma funktioner
- IIFE

Arbetsmetod

- Teori och praktik varvas under lektionen

NACKADEMIN

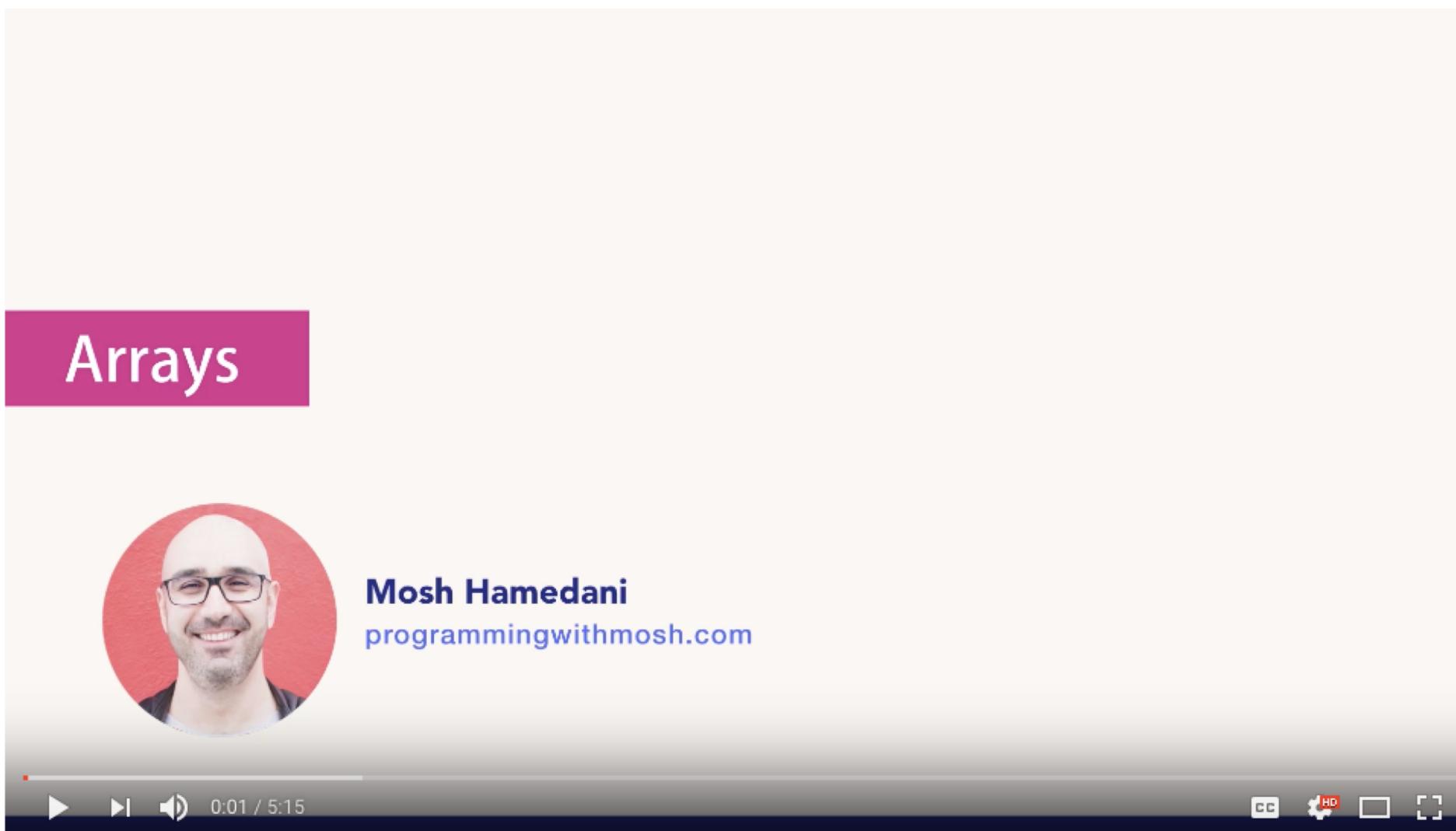
Kort summering av föregående lektion

Föregående lektion:

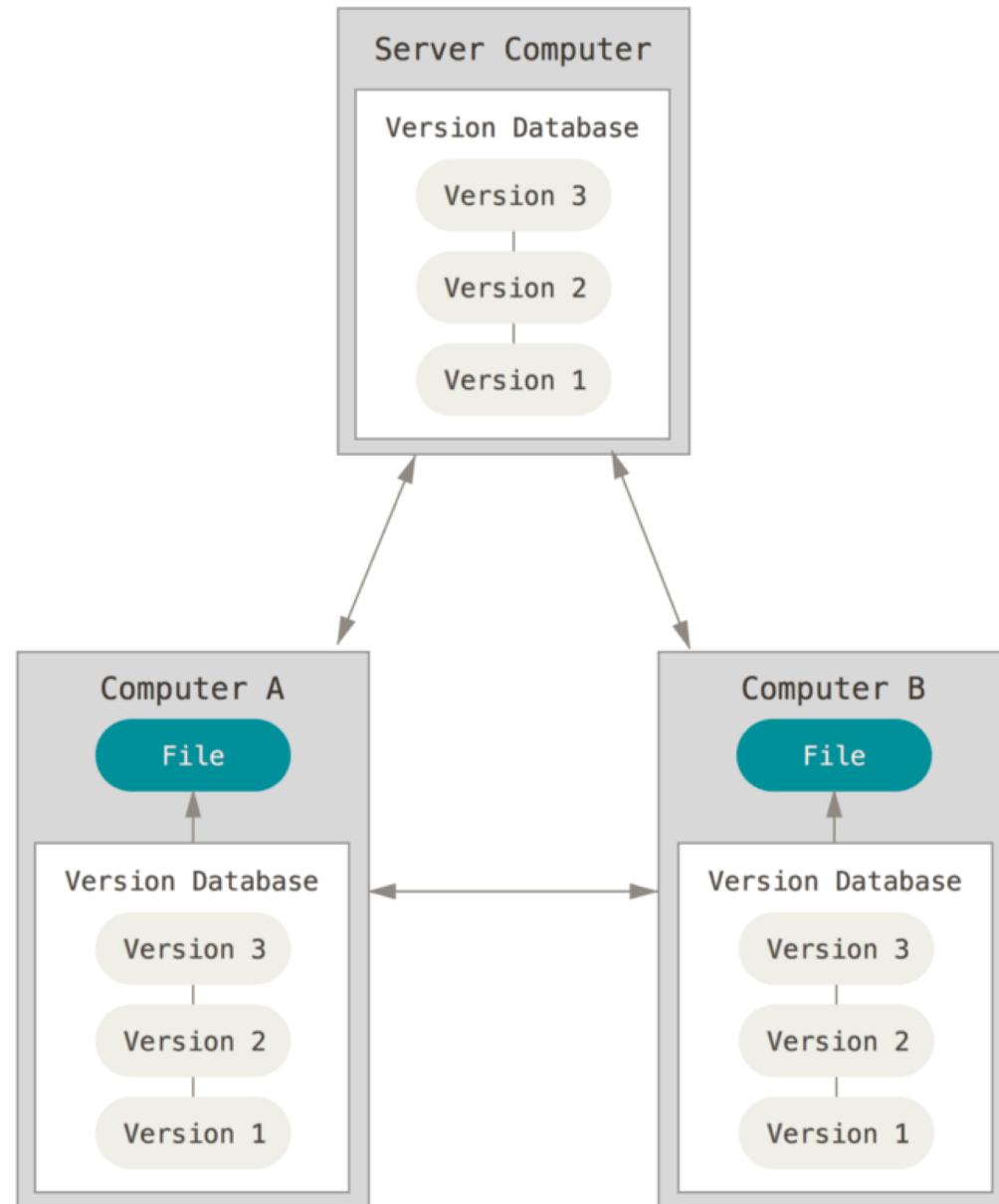
- Vi har jobbat med arrayer (fält),
uttryck och operatorer.

Repetition: JavaScript Arrays av Mosh Hamedani

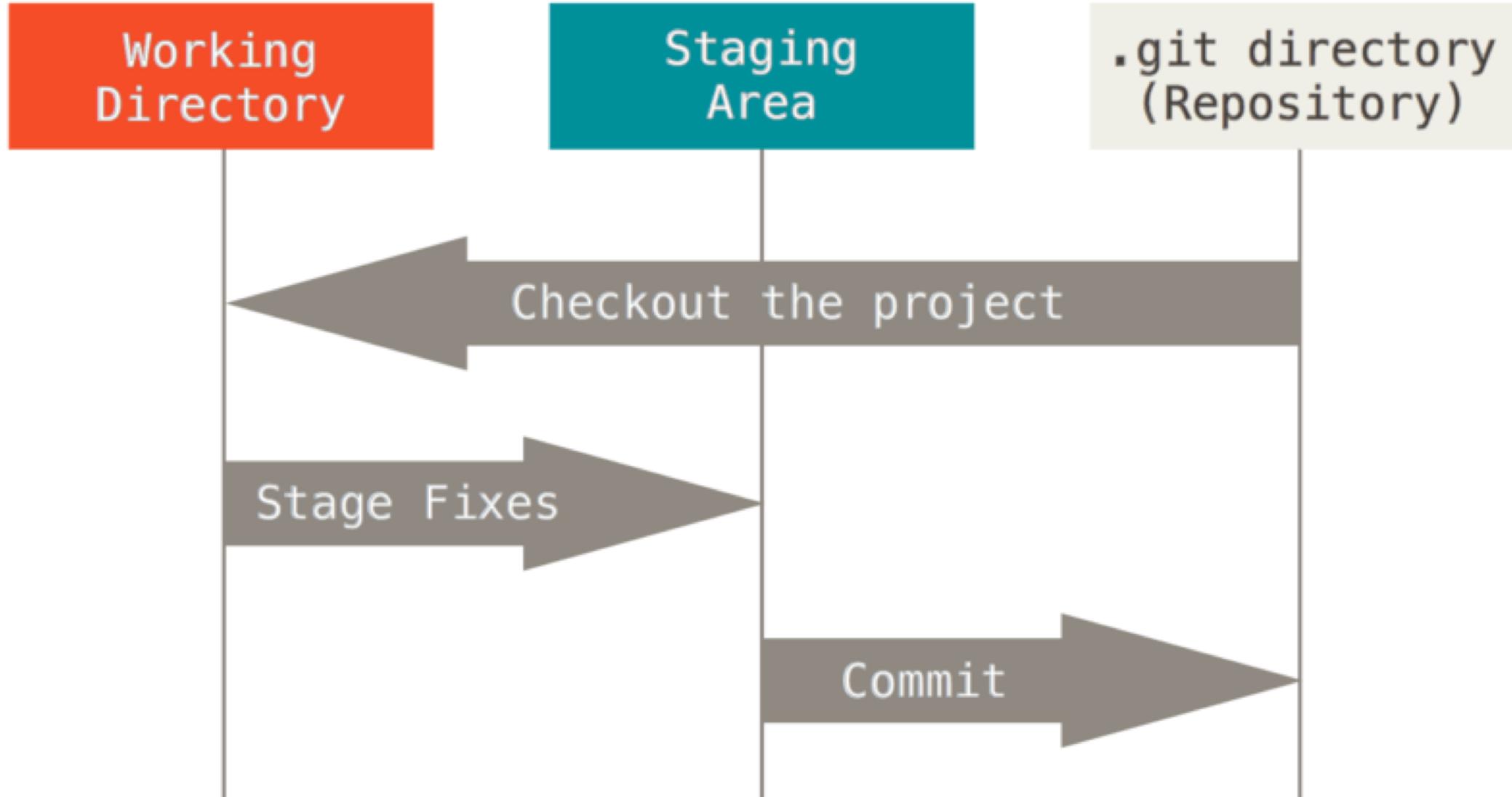
<https://youtu.be/oigfaZ5ApsM>

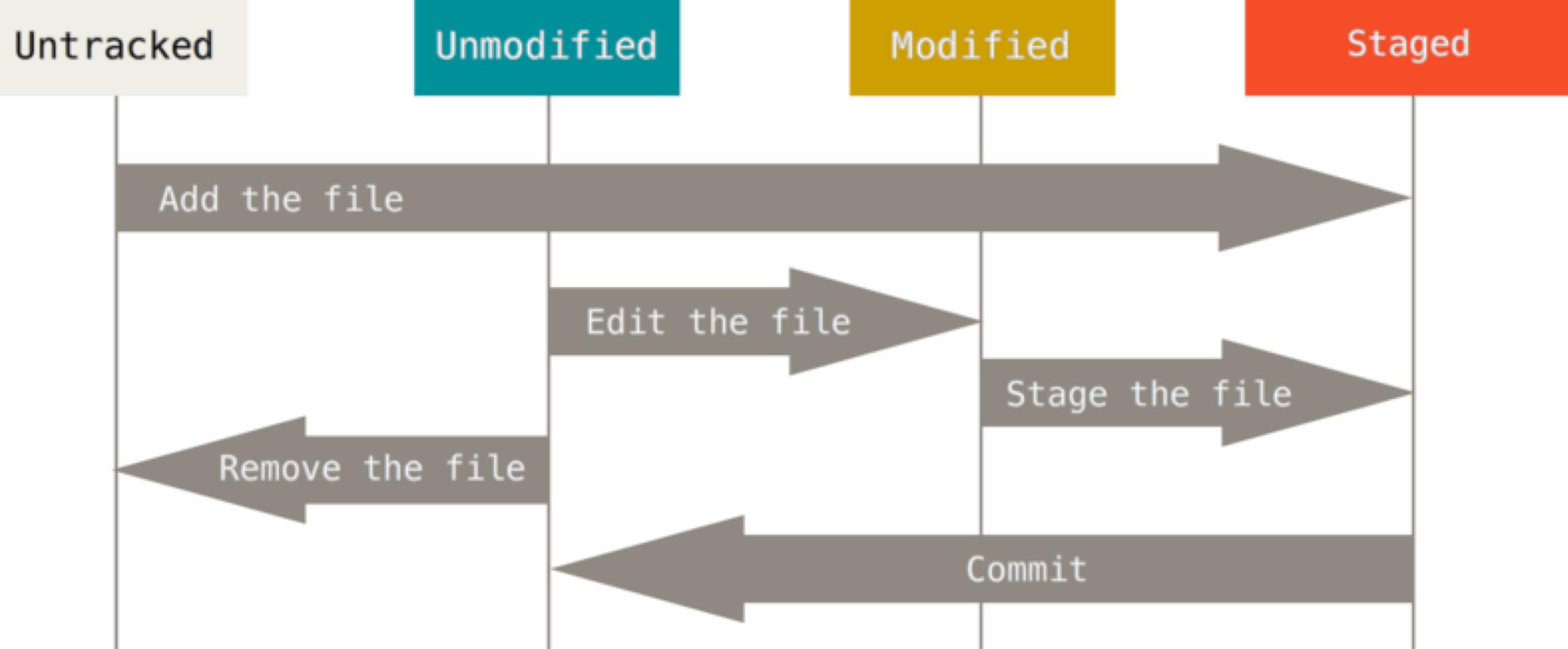


Vad är en DVCS? “Distributed Version Control System”



Alla bilder är hämtade från boken **Pro Git**
<https://git-scm.com/book>





.gitignore

Specifies intentionally untracked files to ignore



Create useful .gitignore files for your project

Search Operating Systems, IDEs, or Programming Languages

Create

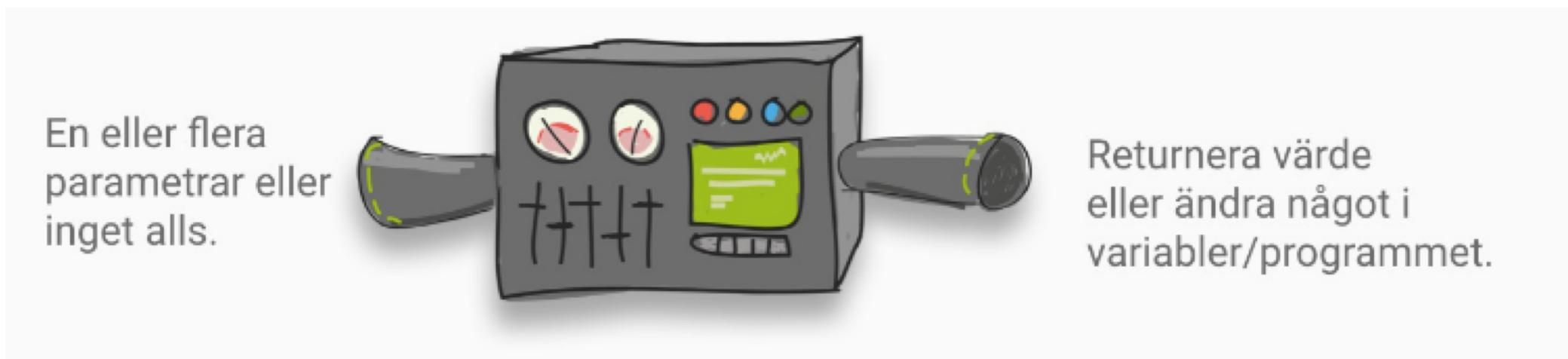
[Source Code](#)

[Command Line Docs](#)

[Watch Video Tutorial](#)

Vad är en funktion?

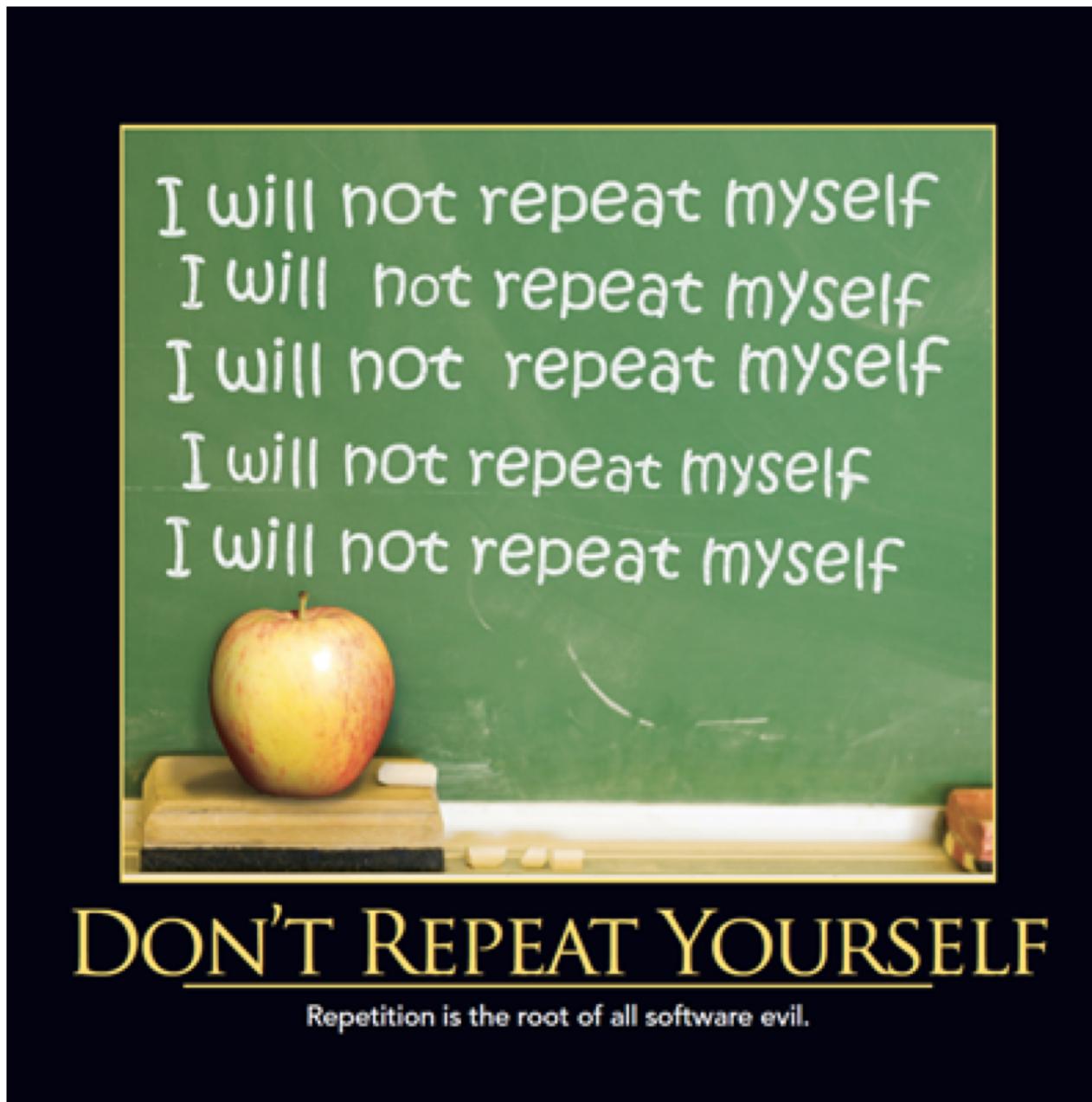
- En funktion är en del av ett program som kan anropas för att utföra en viss uppgift oberoende av resten av koden.
- Funktioner hjälper oss att återanvända kod.



En eller flera parametrar eller inget alls.

Returnera värde eller ändra något i variabler/programmet.

DRY



Bildkälla: <https://deviq.com/don-t-repeat-yourself/>

Deklarera en funktion – Grundläggande syntax

```
function name() {  
    // En eller flera satser...  
}
```

En funktion deklareras med nyckelordet **function**. Efter namnet kommer alltid parenteser () och direkt efter ett **kodblock** som skrivs inom klammerparenteser { } (måsvingar).

En parameterlös funktion

```
function hej(){  
    alert("Hej från en funktion");  
}
```

Anropa en funktion (call/invoke)

- En funktion kan anropas med ()-operator (parenteser)

```
hej();
```

- **hej** är namnet på en funktion som finns i programmet.

OBS!

Du får anropa en funktion som finns längre ner i koden!

```
hej();
```

```
function hej() {  
    alert("Hej från en funktion");  
}
```

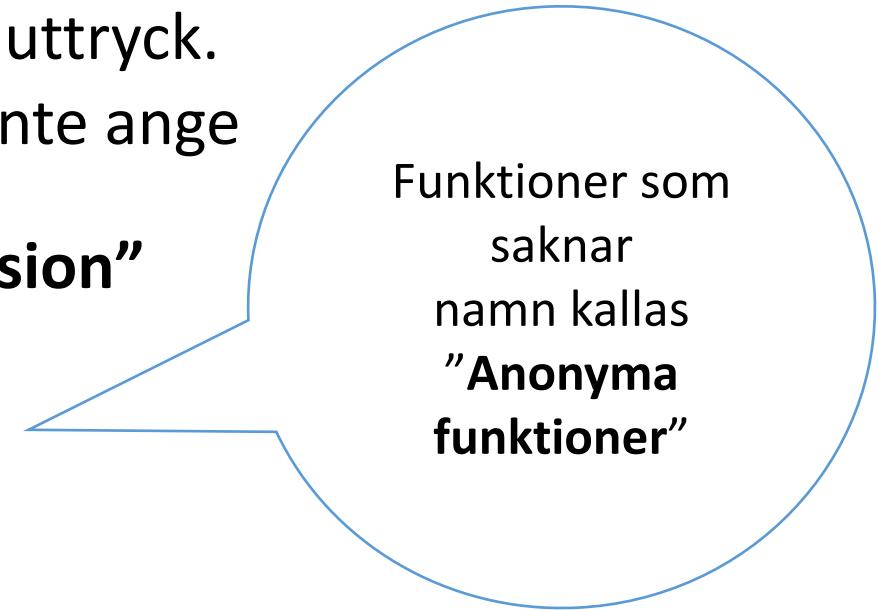
```
hej();
```

Anonyma funktioner

- Nyckelordet **function** används för att skapa ett uttryck.
- Funktionen lagras i en variabel och då behöver vi inte ange ett namn på själva funktionen.
- Funktionen blir alltså ett uttryck **“Function expression”**

```
let hej = function(msg){  
    console.log(msg);  
}  
hej('Hej från en anonym funktion');
```

- OBS! Du får inte göra ett anrop före uttrycket.
(Jämför med föregående exempel).



Övningar

- Skapa en funktion som heter **meddelande**
- Funktionen visar meddelandet "Jag har anropats!" i konsolfönstret.
- Anropa funktionen 3 gånger.
- Flera övningar

https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_functions1

https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_functions2

Parametrar

- Funktioner behöver ibland data för att utföra ett visst uppdrag.
- Data skickas till en funktion via ett godtyckligt antal parametrar.

```
function name(parameter1, parameter2, ...) {  
    // En eller flera satser...  
}
```

En funktion med en parameter

```
function hej(msg){  
    console.log(msg);  
}
```

Argument

- Data skickas till en funktion vid anropet som argument (inom parenteser)

```
hej("Argument – första anropet");
```

```
hej("Argument – andra anropet");
```

Förvald (default) parameter

- Vad får du om du anropar funktionen utan några argument?

```
hej(); // undefined
```

- Lösningen till detta problem är att ange en förvald parameter

```
function hej(msg = "Welcome") {  
    console.log(msg);  
}  
hej(); // Welcome
```

OBS! Skapa aldrig flera funktioner med samma namn!
Testa nedanstående. Vilken funktion anropas?

```
function hej(msg){  
    console.log(msg);  
}  
  
function hej(){  
    console.log("En parameterlös funktion");  
}  
  
hej("Argument – första anropet");  
hej("Argument – andra anropet");
```

Funktioner med två parametrar

```
function hej(firstName, lastName){  
  console.log("Hej " + firstName + " " + lastName );  
}
```

```
hej("Mahmud", "Al Hakim");
```

```
hej("Kalle", "Anka");
```

Funktioner med tre parametrar

```
function hej(firstName, lastName , age){  
    console.log(firstName + " är " + age + " år gammal");  
}  
hej("Mahmud", "Al Hakim", 45);  
hej("Kalle", "Anka", 10);  
  
// Vad händer om du inte anger rätt antal argument t.ex.  
hej("Mahmud");
```

Funktioner med flera parametrar

```
function manyThings(...things) {  
    console.log(things);  
}  
manyThings(1);  
manyThings(1,2);  
manyThings(1,2,3);
```

... är en operator
(tre punkter)
Här skapas en array av alla
argument som skickas till
funktionen

Return

- Funktioner kan returnera ett värde.
- Detta kan vi göra med hjälp av nyckelordet **return**

```
function fullName(firstName, lastName){  
    return firstName + " " + lastName;  
}
```

```
let name = fullName("Mahmud", "Al Hakim");  
console.log(name);
```

function keyword

name

parameter(s)

```
function addTwo(parameter){
```

return keyword

```
return parameter + 2;
```

action to be performed

function body (grayed
out, between curly
braces)

```
}
```

function invocation

```
addTwo(4)
```

arguments

Function will output 6

Immediately-invoked function expression IIFE (Självanropande funktioner)

- Kallas även "self-invoked anonymous function"
- eller " self-executing anonymous function".

```
(function () {  
    console.log("Jag är en IIFE");  
    // I will invoke myself  
})();
```

- Tips: <https://developer.mozilla.org/en-US/docs/Glossary/IIFE>

IIFE returnerar resultatet och inte själva funktionen

```
// Här kommer en IIFE
var result = (function () {
    return "Mahmud";
})();
console.log(result); // Mahmud

// Vad händer utan IIFE
var result2 = function () {
    return "Mahmud";
}
console.log(result2); // Hela funktionen returneras!
```

Övning – Förklara nedanstående kodsnutt

```
function getSize(width, height, depth) {  
    var area = width * height;  
    var volume = width * height * depth;  
    var sizes = [area, volume];  
    return sizes;  
}  
var areaOne = getSize(3, 2, 3)[0];  
var volumeOne = getSize(3, 2, 3)[1];
```

Övning 1

- a) Skapa en funktion som beräknar medelvärdet av två tal.
- b) Skapa en funktion som beräknar medelvärdet av tre tal.

Övning 2

- Skapa ett program som läser in ett **namn** via en prompt.
- Skicka namnet till en funktion som returnerar en hälsningsfras, t.ex.
"Hej **namn**. Välkommen till vår webshop".
- Visa meddelandet på webbsidan. (ej alert eller console).



Övning 3

- Skapa ett program som läser in antalet minuter man i genomsnitt ringer per månad samt kostnaden per minut.
- Som resultat skall programmet visa en dialogruta (alert) där det anges hur stor den beräknade kostnaden per månad blir.



Övning 4

- Skriv ett program som läser in en varas pris, inkl. moms.
- Programmet skall också läsa in momssatsen som skall vara ett helt antal procent.
- Programmet skall beräkna dels varans pris exkl. moms och dels momsen.
- De två resultaten skall visas i en dialogruta (alert).
- Tips! Använd `\n` för att lägga till en radbrytning.

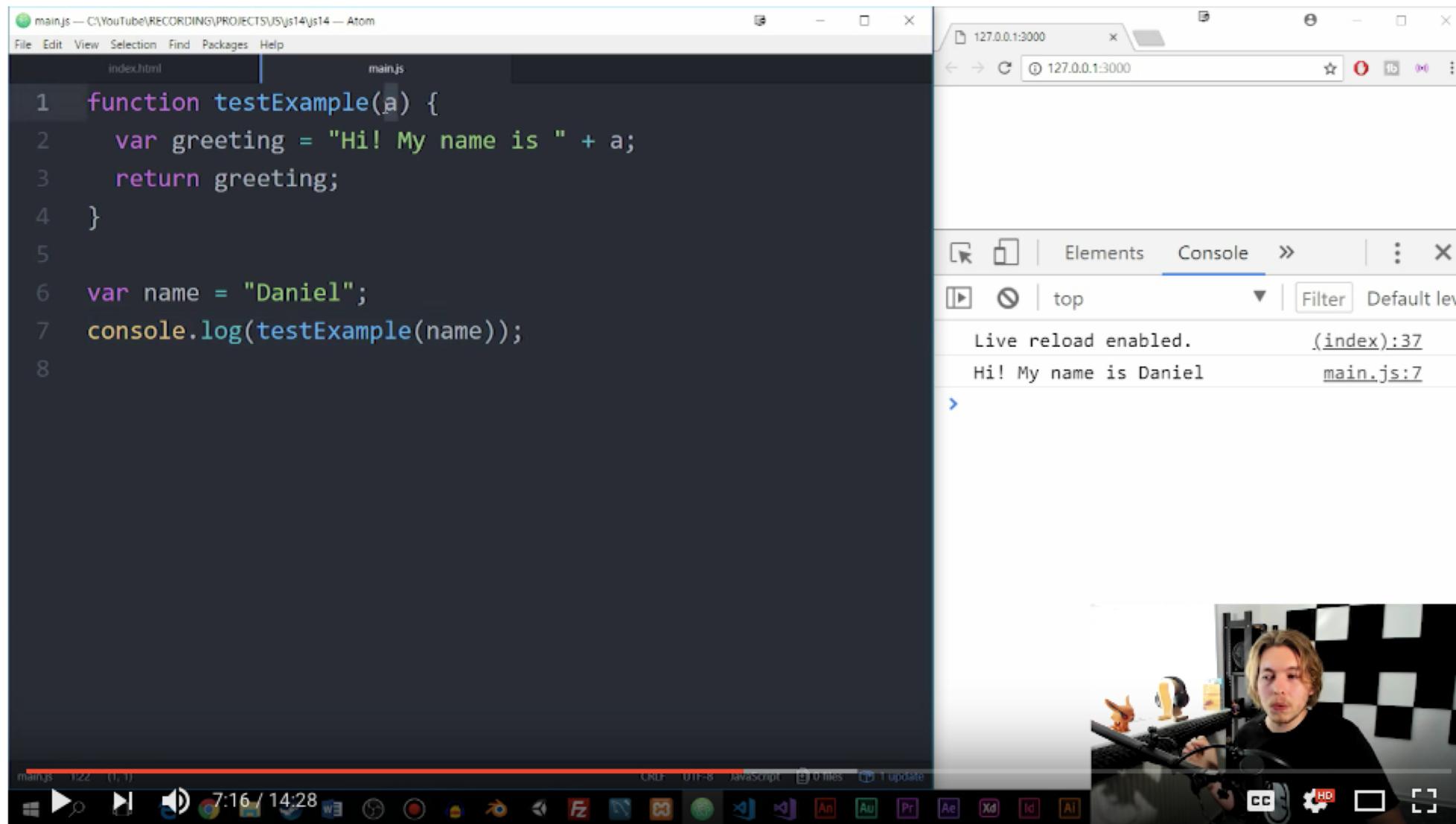
Extra övningar

https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_functions3

https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_functions4

Tips: How to Create JavaScript Functions

<https://youtu.be/Tvu1FqhOUrQ>



The screenshot shows a developer's setup. On the left, a monitor displays an Atom code editor with a dark theme. The file 'main.js' is open, containing the following code:

```
1 function testExample(a) {  
2     var greeting = "Hi! My name is " + a;  
3     return greeting;  
4 }  
  
5  
6 var name = "Daniel";  
7 console.log(testExample(name));  
8
```

On the right, a browser window shows the output of the code execution. The URL is '127.0.0.1:3000'. The browser's developer tools are open, specifically the 'Console' tab, which shows the following logs:

```
Live reload enabled. (index):37  
Hi! My name is Daniel main.js:7
```

At the bottom of the screen, a taskbar is visible with various application icons.

Summering av dagens lektion

- **Vi har idag jobbat med funktioner, parametrar och argument.**
- Reflektioner kring dagens lektion?
 - Vad tar du med dig från dagens lektion?
 - Finns det något som var extra svårt att förstå?
 - Finns det något som vi behöver repetera?
 - Hur upplevde du dagens arbetsmetoder?

Framåtblick inför nästa lektion

- Läs: Bok 2, sid. 85–97
- Under nästa lektion kommer vi att arbeta med **objekt**.