Delegate / Lambda
Linq

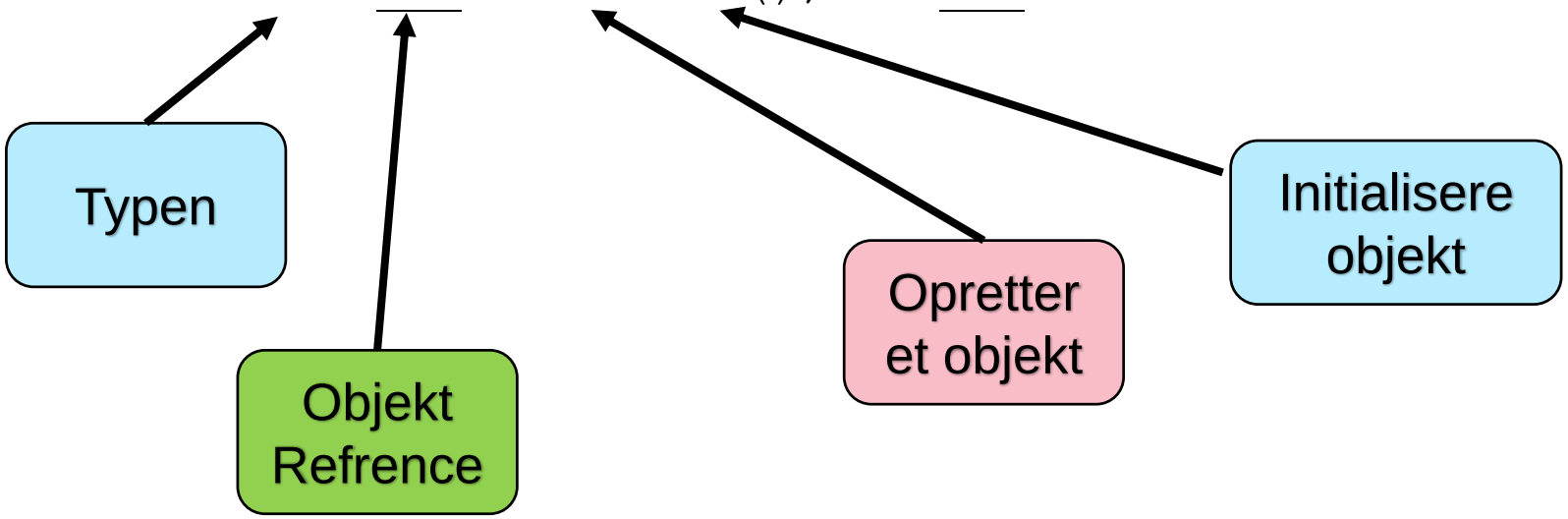
Peter Levinsky, IT Roskilde

25.02.2021

Lambda / Delegates

Kender Object referencer

```
Car car = new Car(); -- car er en reference til et objekt
```



Delegates

Referencer til metoder

```
delegate int AddType(int x, int y);
```

-- type erklæring af en reference til en metode
-- Her en metode der tager to int parametre og
-- returnere en int

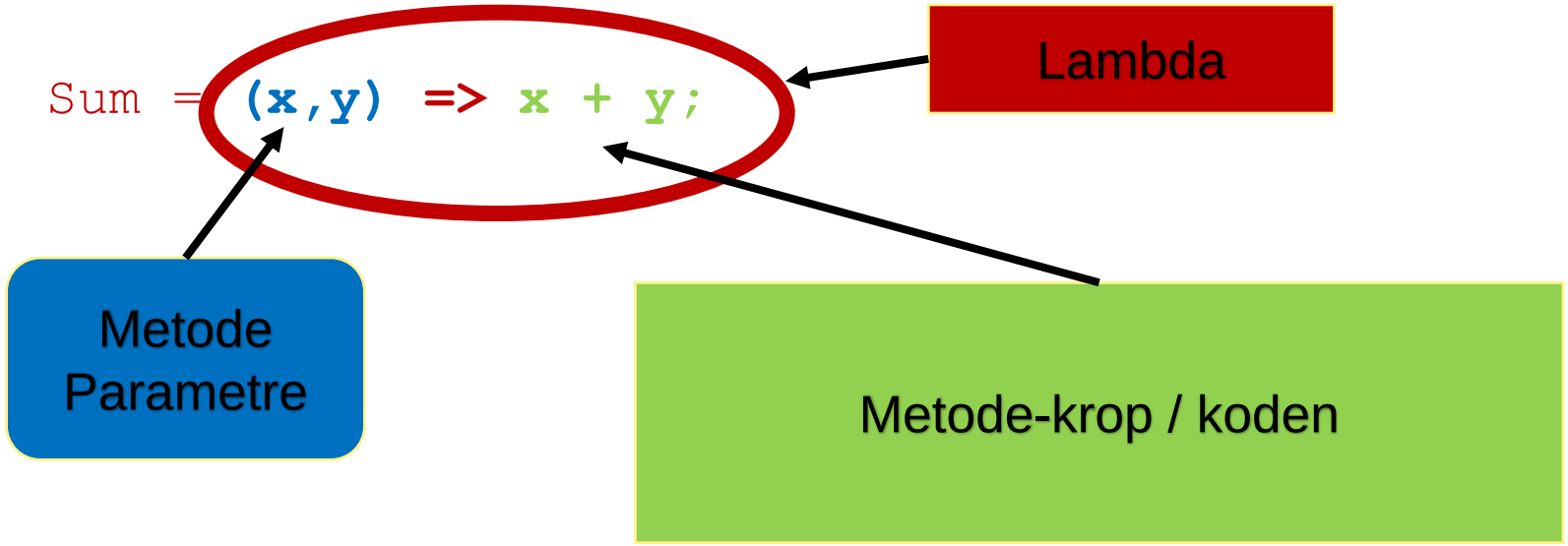
```
private AddType Sum = null;
```
-- Type variablenavn = værdi

```
Sum = (x,y) => x + y;
```
-- variablenavn for ny værdi (metode)

```
int sum = Sum(5,8);
```
-- anvender metode-referencen til at
-- kalde metoden med to tal

Lambda

Anonyme metoder (som regel små metoder er defineres løbende)



Andre lambda udtryk

```
    ()=> Console.WriteLine(”Peter was Here”);          // ingen parametre
   (i) => {                                            // krop flere linjer
            i=i*i;
            Console.WriteLine(”i opløftet til 2 = ” + i);
           }
```

Demo

Opgaver PARA.4 .

LINQ – A query language – like SQL

To eksempler

(Der er en liste: List<Bil> **biler** = new List<Bil>(MockDataFactory.GetBiler);)

Query Language

```
var res = from b in biler
          where b.Farve == "rød"
          select b.RegistreringsNr
```

Metode

```
Var res = biler.Where(b => b.Farve == "rød").
          Select(b => b.RegistreringsNr);
```

LINQ – A query language – like SQL

Forudsætning

1. En datastruktur der understøtter

```
public interface IEnumerable<out T>
{
    IEnumerator<T> GetEnumerator();
}
```

2. Hvor **IEnumerator** indeholde tre metoder

```
bool MoveNext();
void Reset();
T Current { get; }
```

LINQ – A query language – like SQL

Mapper de tre metoder til foreach-loop

```
Foreach (var t in 'liste')  // Reset – starter på ny
{

      t // pege på Current


}  // MoveNext -> sand hvis der findes nogen næste ellers falsk
```

Demo

Opgaver Pro 3.4, Pro 3.4a, Pro 3.4b .