# Velkommen!

•••

Johanna
johannph på mattermost
johannph@uio.no på mail!

# Kort: Praktisk informasjon

- Undervisningstilbud
  - https://www.uio.no/studier/emner/matnat/ifi/IN1010/v21/undervisningstilbud/
  - Jeg har konkrete spørsmål/problemer med min kode -> Labtime!
  - Jeg vil ha mer liveprogrammering -> Plenumstime!
  - Jeg vil jobbe med andre (og kanskje en kjapp recap av forelesning) -> Gruppetime!
  - Jeg vil ha en recap av de vanskeligste konseptene fra forelesning -> Repetisjonsgruppe!
- Skriv alt dere lurer på i chatten enten til everybody eller bare til meg 😊
- Grupper oblig 4 er ute, håper dere har fått en gruppe, eller laget en selv!

# Mentimeter neste uke: repetisjon

https://www.menti.com/wpzy45bsao

# Repetisjon forrige uke

# Forrige gang

Stabel

Kø

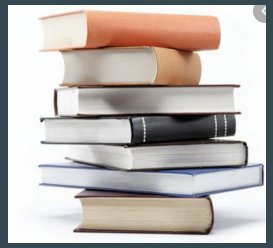
Litt mer om lenkelister

Comparable og compareTo

Katt som lenkeliste: Iterable og iterator

# Stabel/stack og kø/queue

Stabel: LIFO (Last In First Out)



Kø: FIFO (First In First Out)

# Interface

1) Vi bruker interface eller grensesnitt for det vi kan aksessere utenfor en klasse, altså de metodene som er public.
2) Vi bruker også ordet interface når vi vil sette noen krav til interfacet/Grensesnittet til en klasse.
    - F.eks. i obligen så satte vi noen krav til Lenkeliste når vi sa at Lenkeliste implements interfacet Liste, Lenkeliste måtte da ha metodene gitt i interfacet Liste.

# Interfacet Comparable og Metoden CompareTo

Hvis en klasse implements Comparable betyr det at den må ha metoden compareTo() og at vi kan gjøre sånne operasjoner:

objekt1.compareTo(objekt2)
objekt1.compareTo(objekt2)
objekt1.compareTo(objekt2)

f.eks. klassen Integer og String implementerer comparable:

“a”.compareTo(“b”)  “a”.compareTo(“b”)  “a”.compareTo(“b”)

# Interface Iterable og interfacet Iterator

Disse finnes i java biblioteket!! Vi skal ikke skrive dem!

```
public interface Iterator<T>{
  boolean hasNext();
  T next();
}
public interface Iterable{
  Iterator iterator();
}
```

Hvis vi ønsker å lage en beholder der man f.eks. kan bruke en for-each-loop må beholderen vår implements interface Iterable altså implementere metoden iterator(). Metoden skal returnere et objekt av en klasse som implements interfacet Iterator. Den klassen må vi også skrive!

# Repetisjon denne uken

# I dag

Casting

Exceptions

# Hva printes? Send meg direktemelding i chatten!

```java
class Dyr{
    protected String navn;

    public Dyr(String navn){
        this.navn = navn;
    }
    public String hentNavn(){
        return navn;
    }
    public String hentLyd(){
        return "hello";
    }
}
```

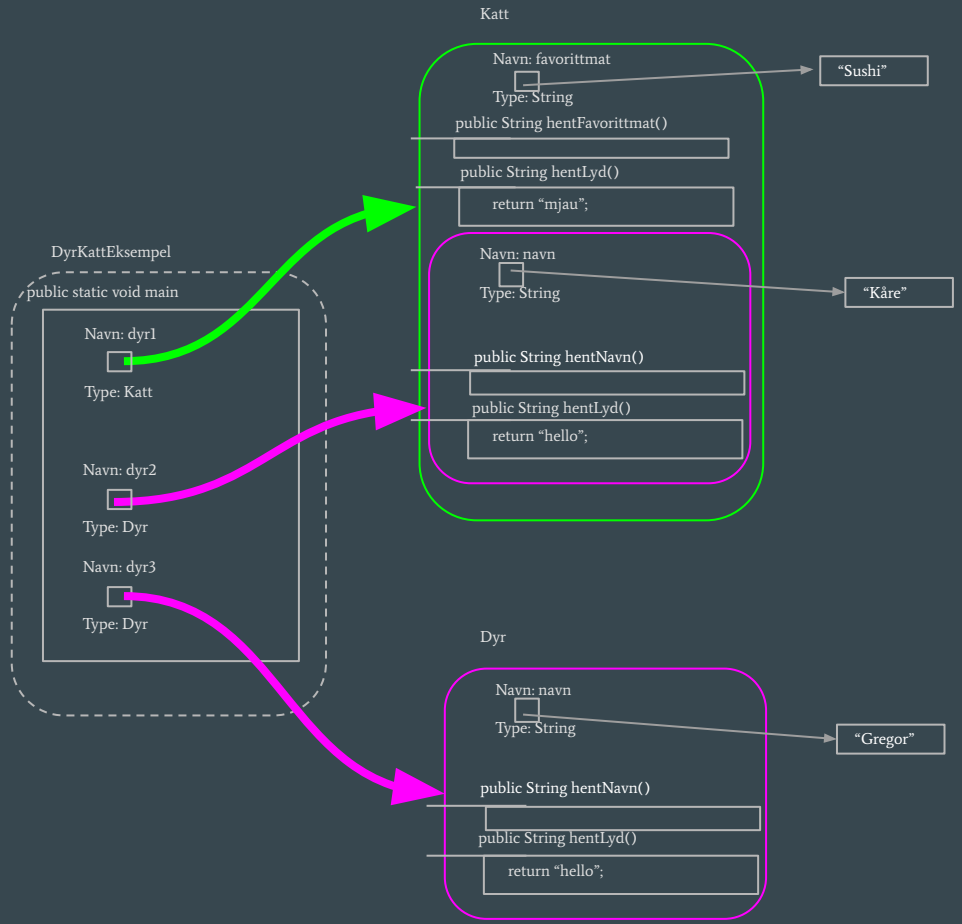
```java
class Katt extends Dyr{
    protected String favorittmat;

    public Katt(String navn, String favorittmat){
        super(navn);
        this.favorittmat = favorittmat;
    }
    public String hentFavorittMat(){
        return favorittmat;
    }
    public String hentLyd(){
        return "mjau";
    }
```

```java
class TestKatt{
    public static void main(String[] args){
        Katt dyr1 = new Katt("Kåre", "Sushi");
        Dyr dyr2 = (Dyr) dyr1;
        Dyr dyr3 = new Dyr("Gregor");
        System.out.println(dyr1.hentLyd());
        System.out.println(dyr2.hentLyd());
        System.out.println(dyr3.hentLyd());
        System.out.println(dyr1.hentNavn());
        System.out.println(dyr2.hentNavn());
        System.out.println(dyr3.hentNavn());
        System.out.println(dyr1.hentFavorittMat());
        System.out.println(dyr2.hentFavorittMat()); //ERROR
        System.out.println(dyr3.hentFavorittMat()); //ERROR
```

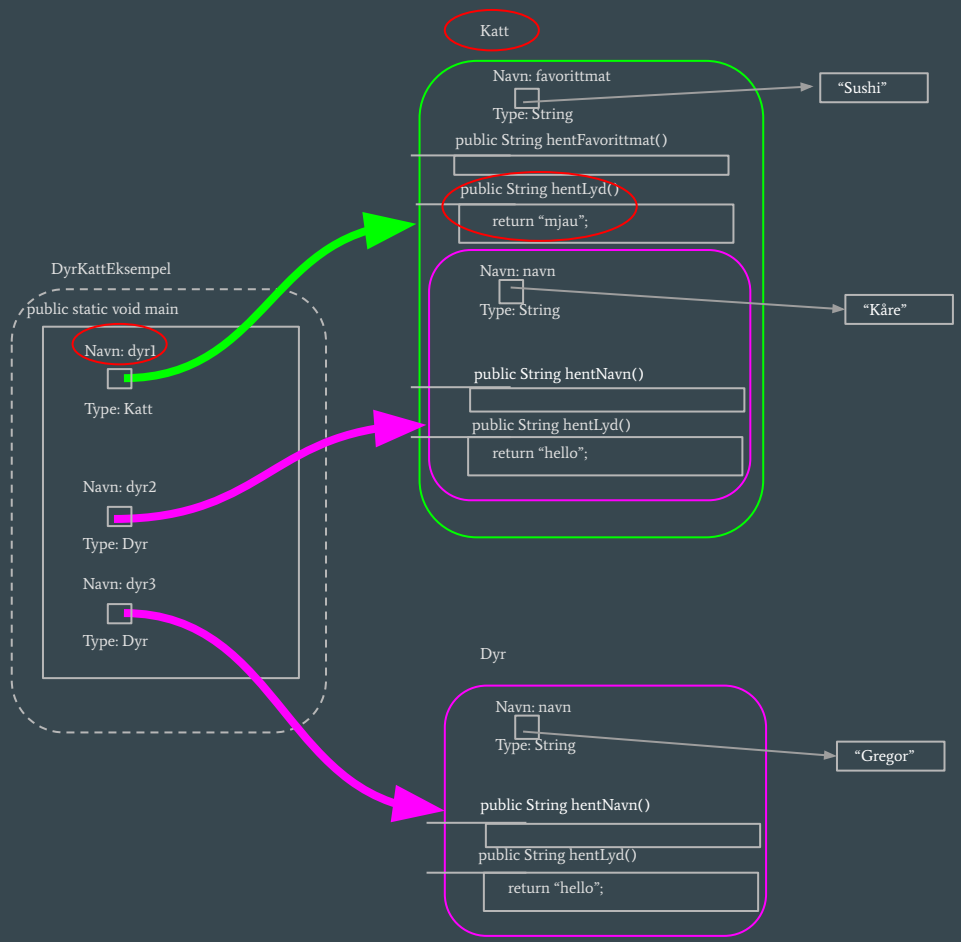
# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```



# Casting

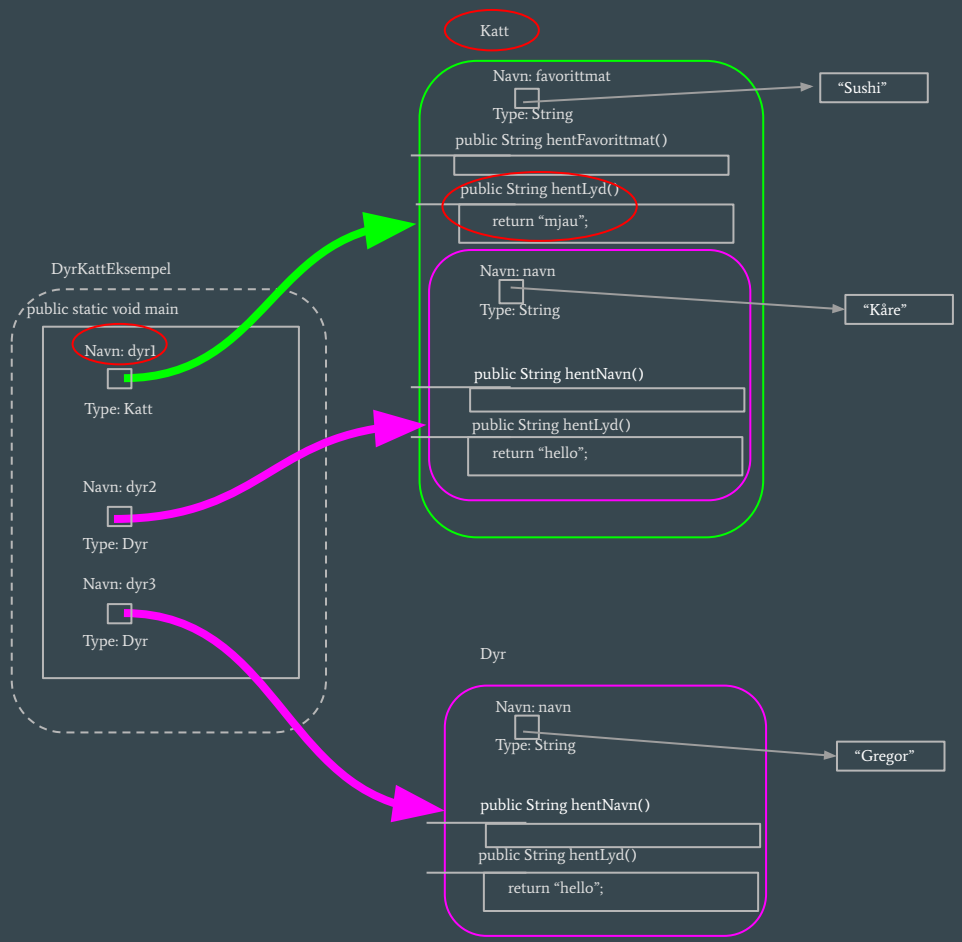
```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```




Vi må se på typen til objektet for å avgjøre hvilken hentLyd metode som kjører.

# Casting

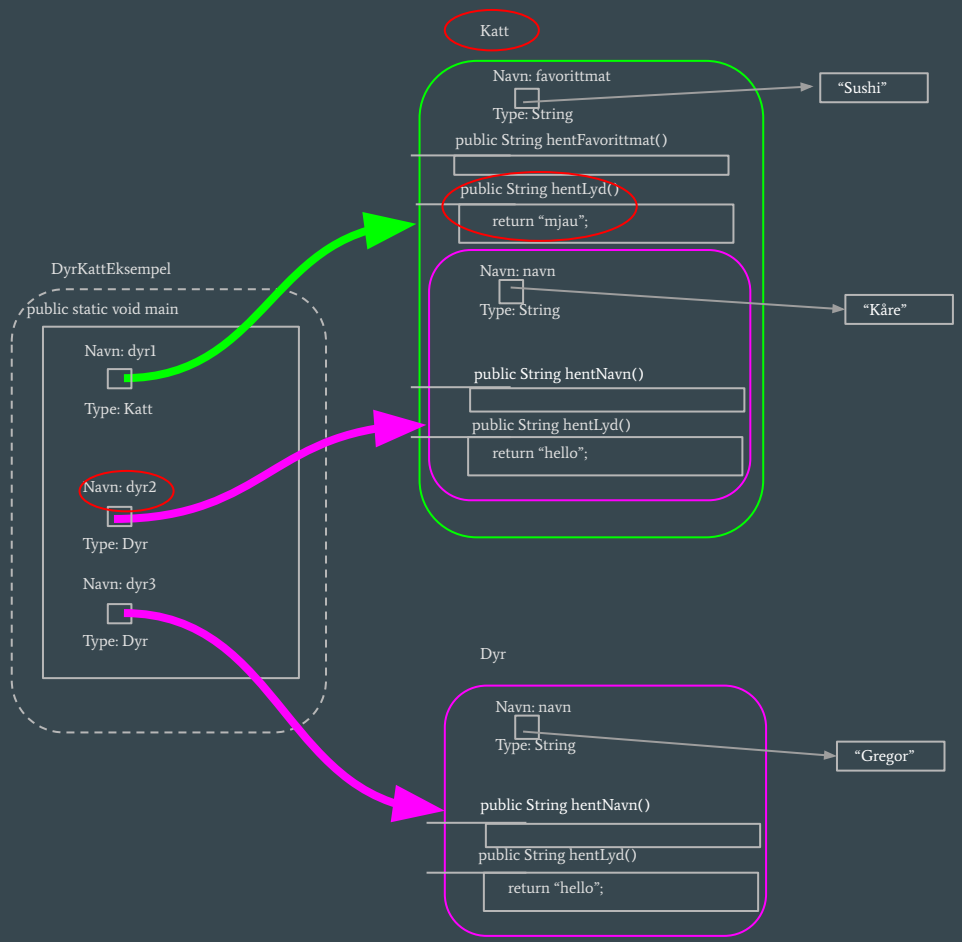
```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());   mjau
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```




Vi må se på typen til objektet for å avgjøre hvilken hentLyd metode som kjører.

# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());   mjau
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

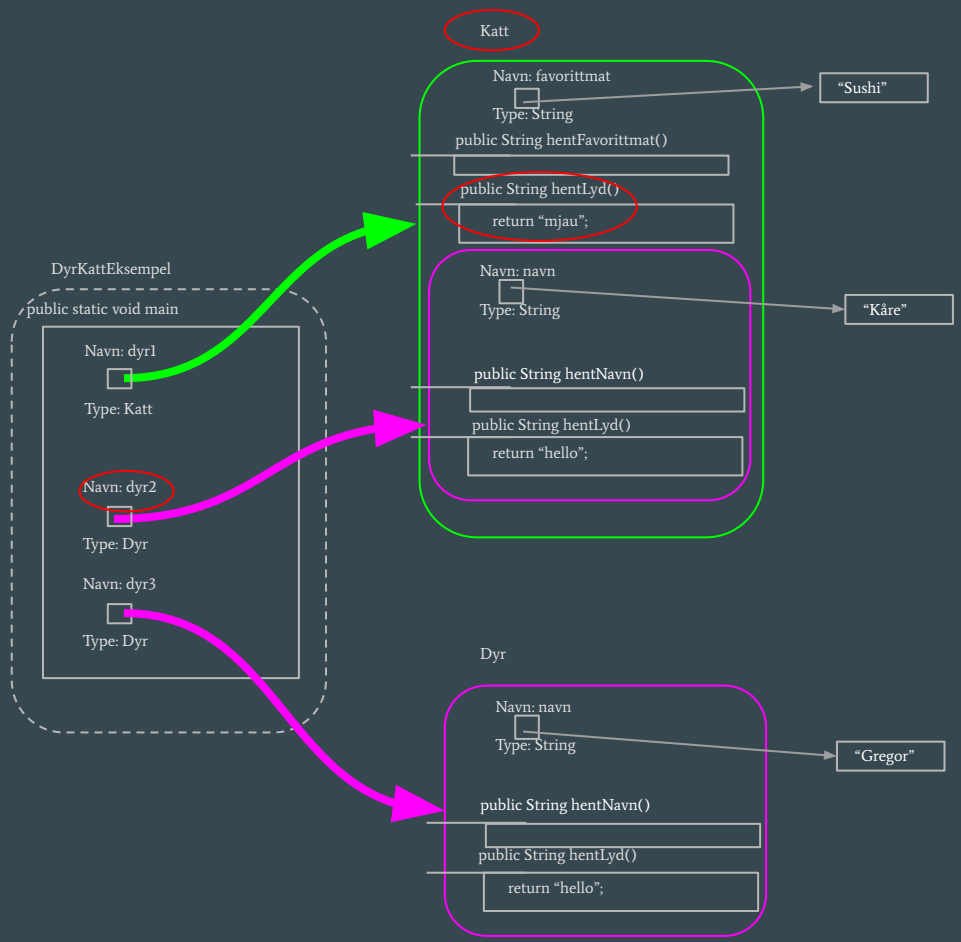



Vi må se på typen til objektet for å avgjøre hvilken hentLyd metode som kjører.

# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

```
mjau
mjau
```

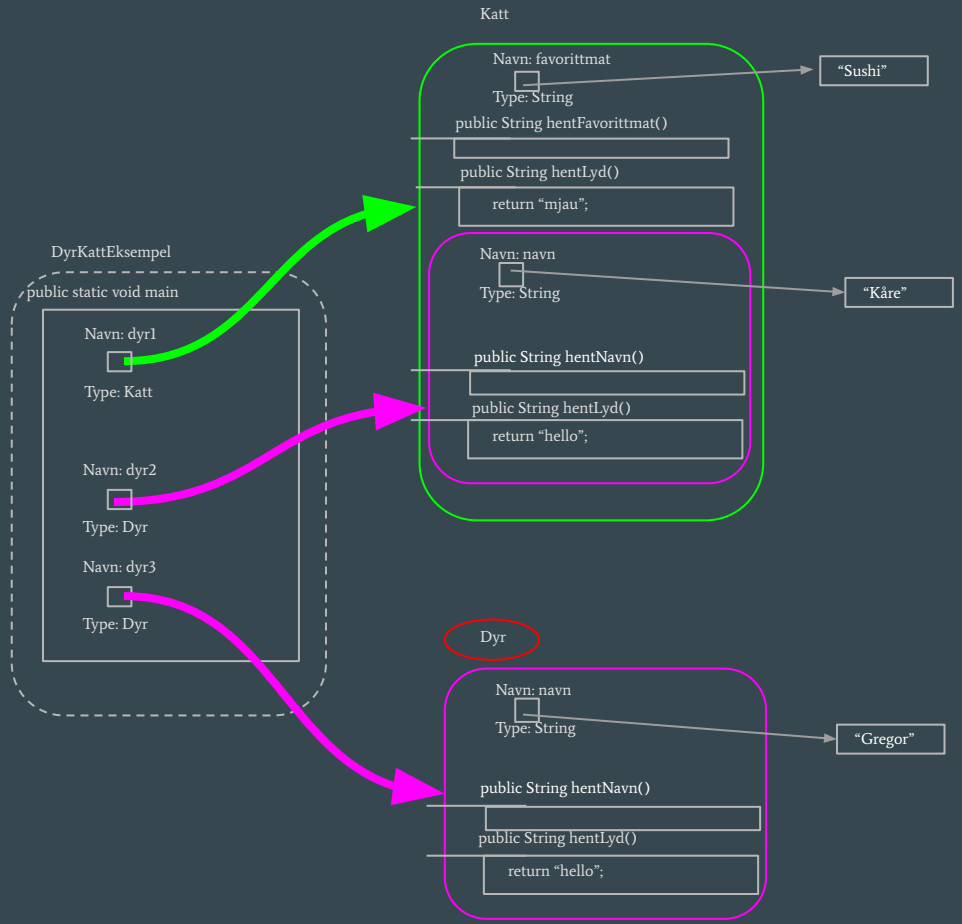



Vi må se på typen til objektet for å avgjøre hvilken hentLyd metode som kjører.

# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

```
mjau
mjau
```

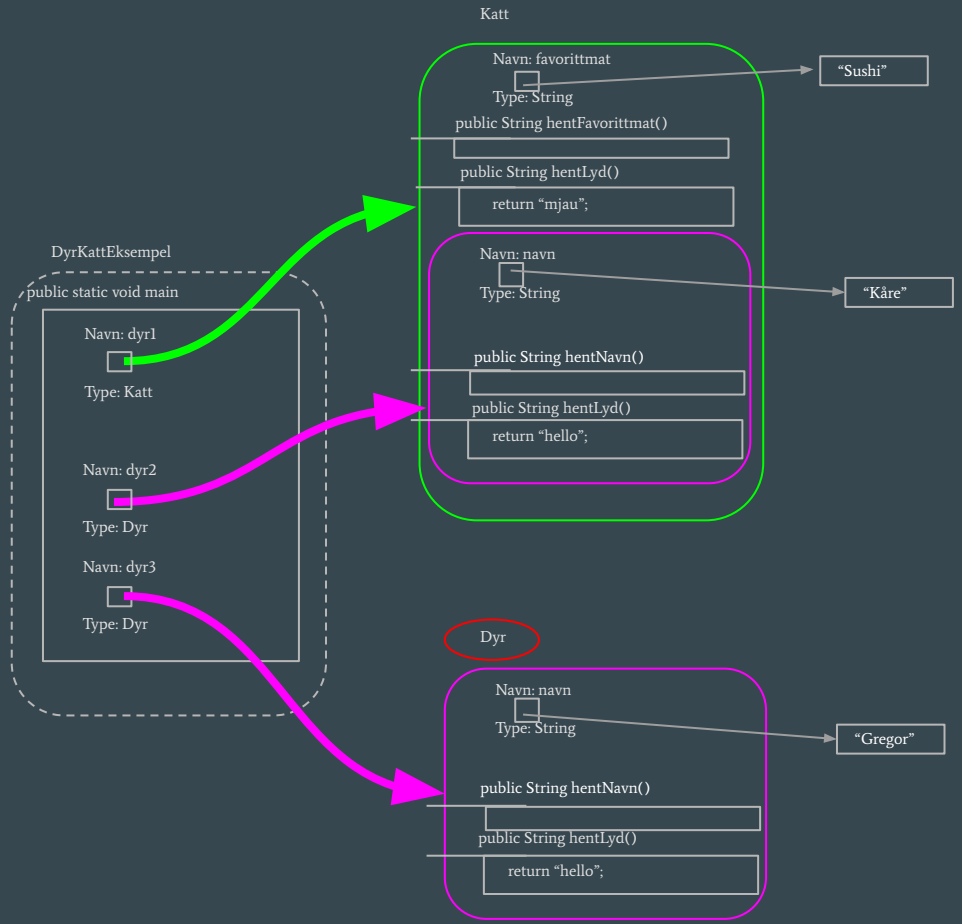



Vi må se på typen til objektet for å avgjøre hvilken hentLyd metode som kjører.

# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

```
mjau
mjau
hello
```

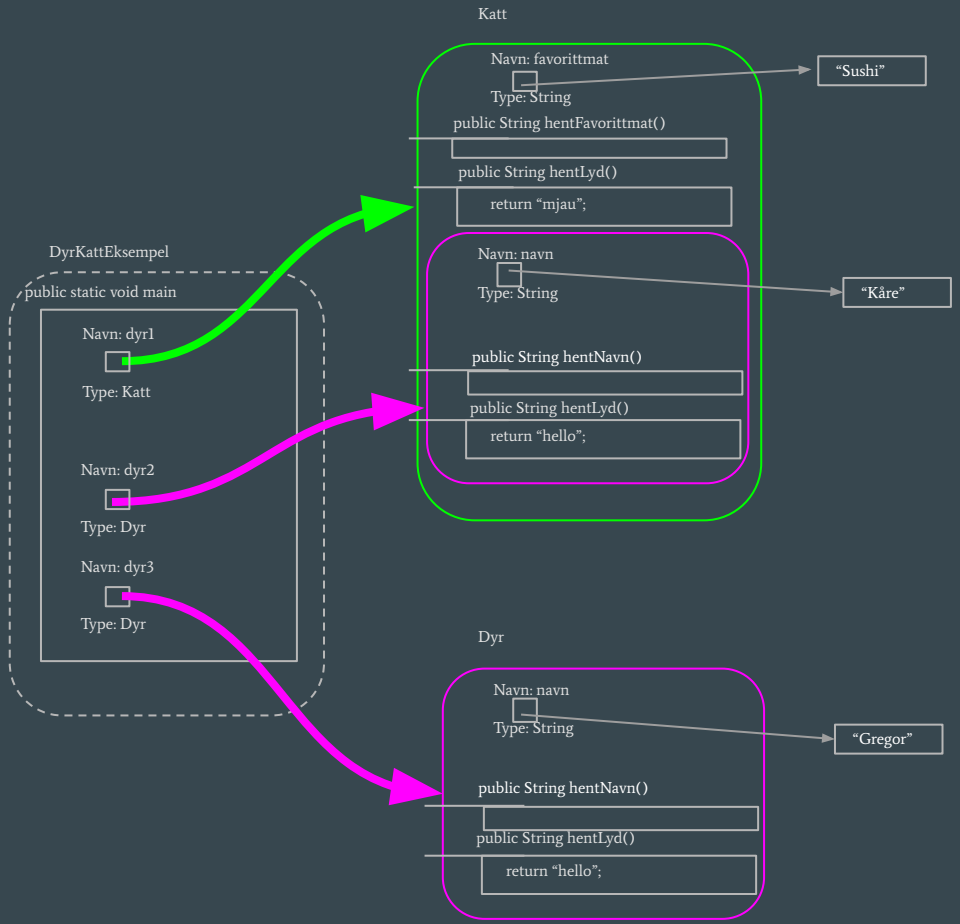



Vi må se på typen til objektet for å avgjøre hvilken hentLyd metode som kjører.

# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

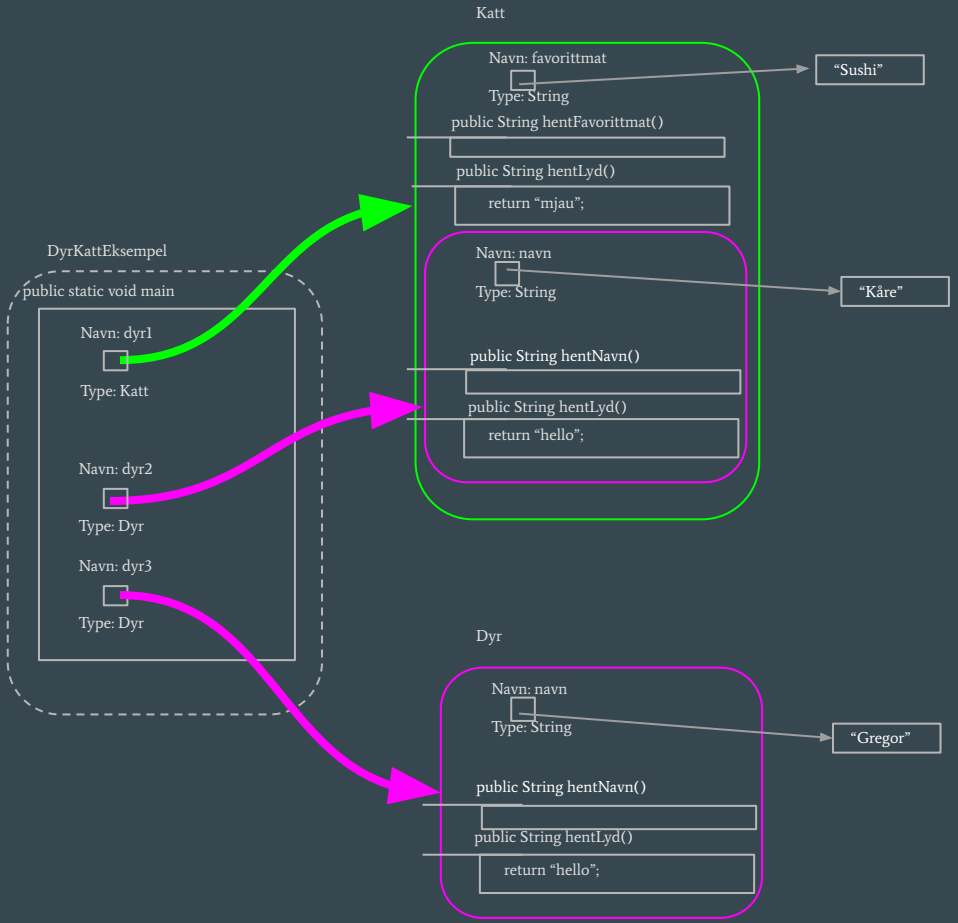
```
mjau
mjau
hello
Kåre
```



# Casting

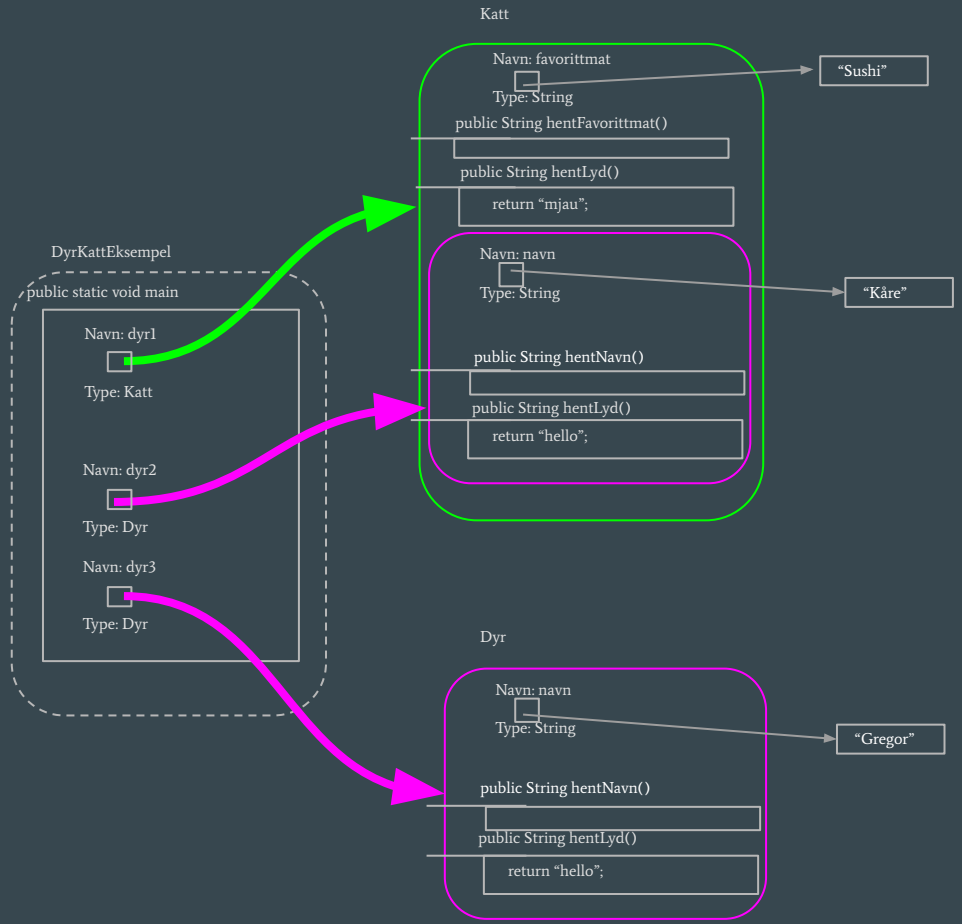
```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());
    System.out.println(dyr2.hentLyd());
    System.out.println(dyr3.hentLyd());
    System.out.println(dyr1.hentNavn());
    System.out.println(dyr2.hentNavn());
    System.out.println(dyr3.hentNavn());
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

```
mjau
mjau
hello
Kåre
Kåre
```



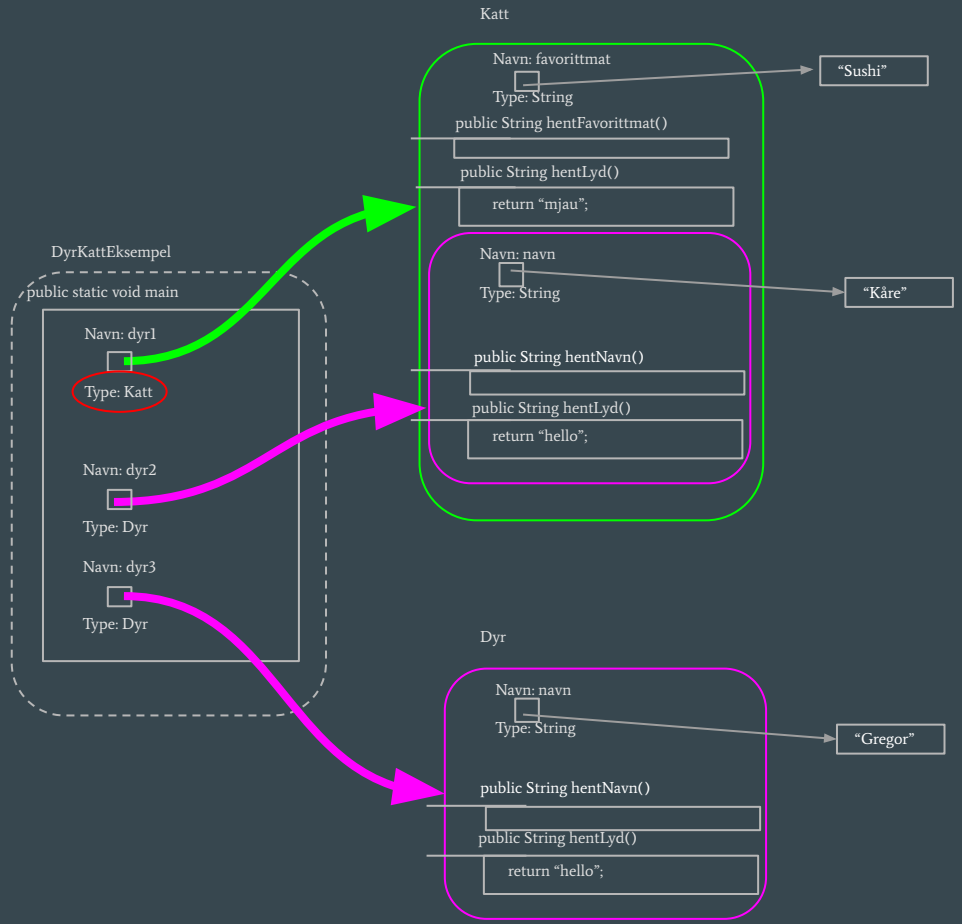
# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());          mjau
    System.out.println(dyr2.hentLyd());          mjau
    System.out.println(dyr3.hentLyd());          hello
    System.out.println(dyr1.hentNavn());         Kåre
    System.out.println(dyr2.hentNavn());         Kåre
    System.out.println(dyr3.hentNavn());         Gregor
    System.out.println(dyr1.hentFavorittMat());
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```



# Casting

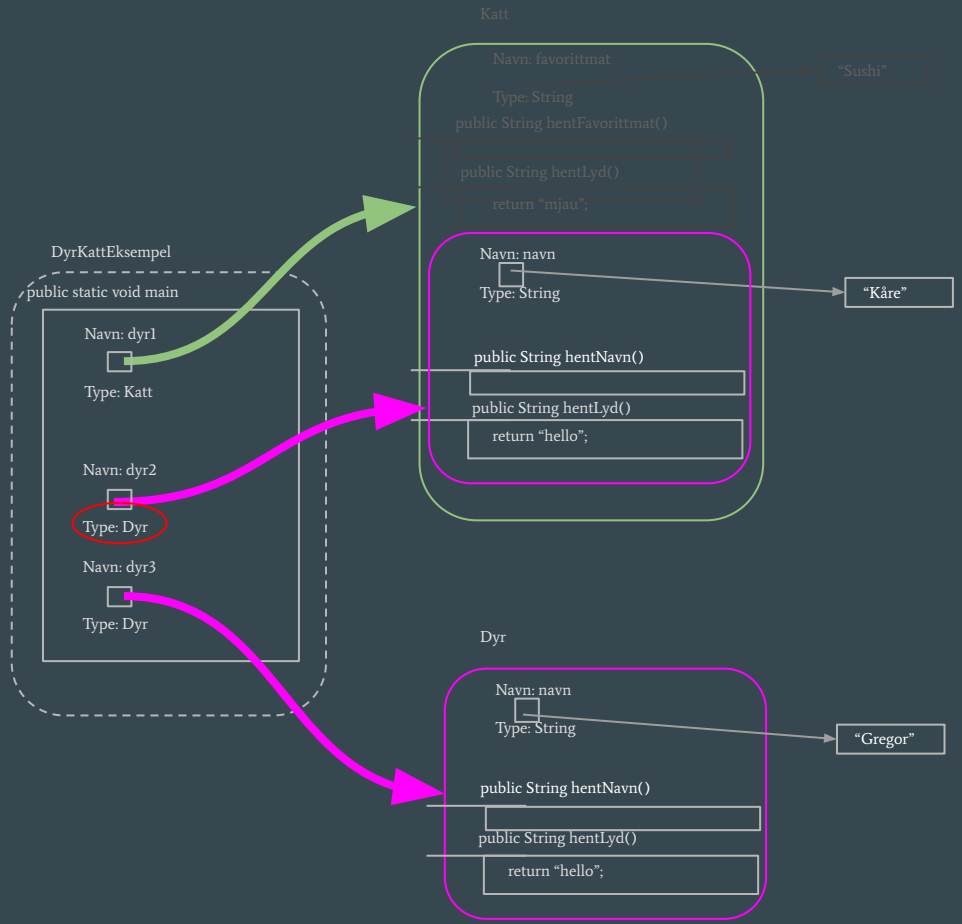
```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());          mjau
    System.out.println(dyr2.hentLyd());          mjau
    System.out.println(dyr3.hentLyd());          hello
    System.out.println(dyr1.hentNavn());         Kåre
    System.out.println(dyr2.hentNavn());         Kåre
    System.out.println(dyr3.hentNavn());         Gregor
    System.out.println(dyr1.hentFavorittMat());  Sushi
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```




Vi må se på typen til referansen for å avgjøre hva vi kan aksessere.

# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());          mjau
    System.out.println(dyr2.hentLyd());          mjau
    System.out.println(dyr3.hentLyd());          hello
    System.out.println(dyr1.hentNavn());         Kåre
    System.out.println(dyr2.hentNavn());         Kåre
    System.out.println(dyr3.hentNavn());         Gregor
    System.out.println(dyr1.hentFavorittMat());  Sushi
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```



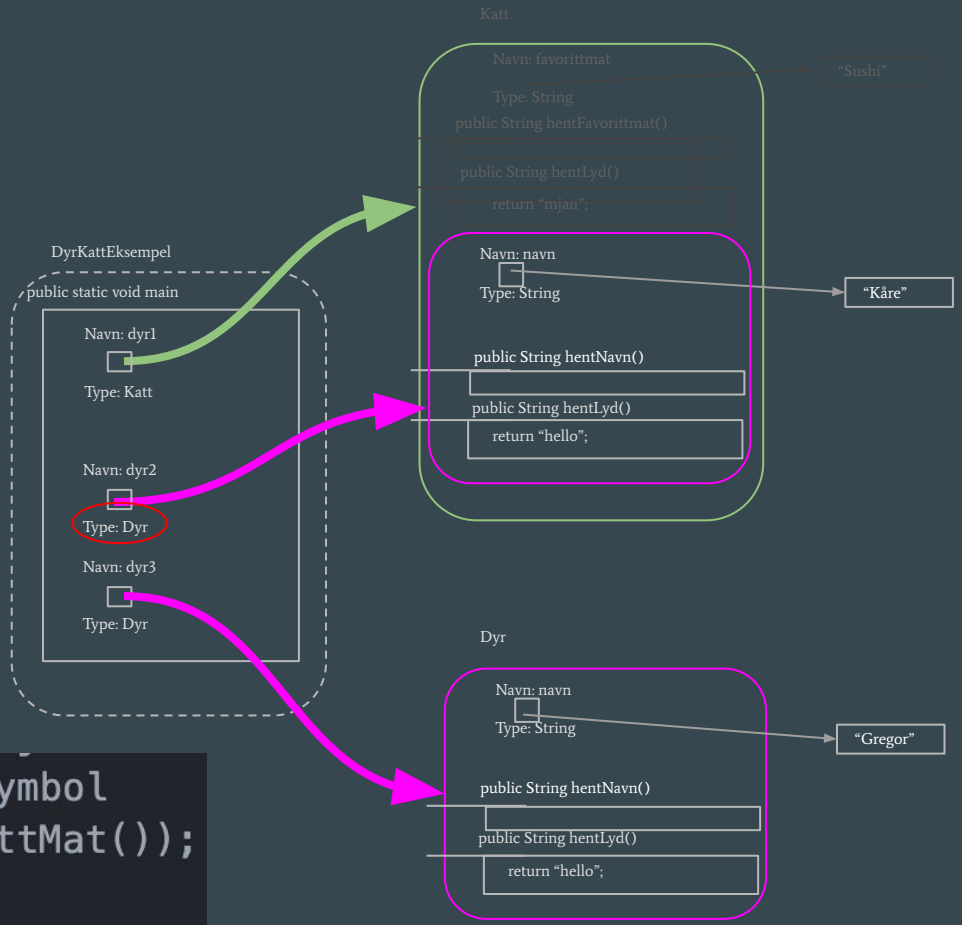

Vi må se på typen til variabelen for å avgjøre hva vi kan aksessere.

Vi tar på oss dyr-brillene

# Casting

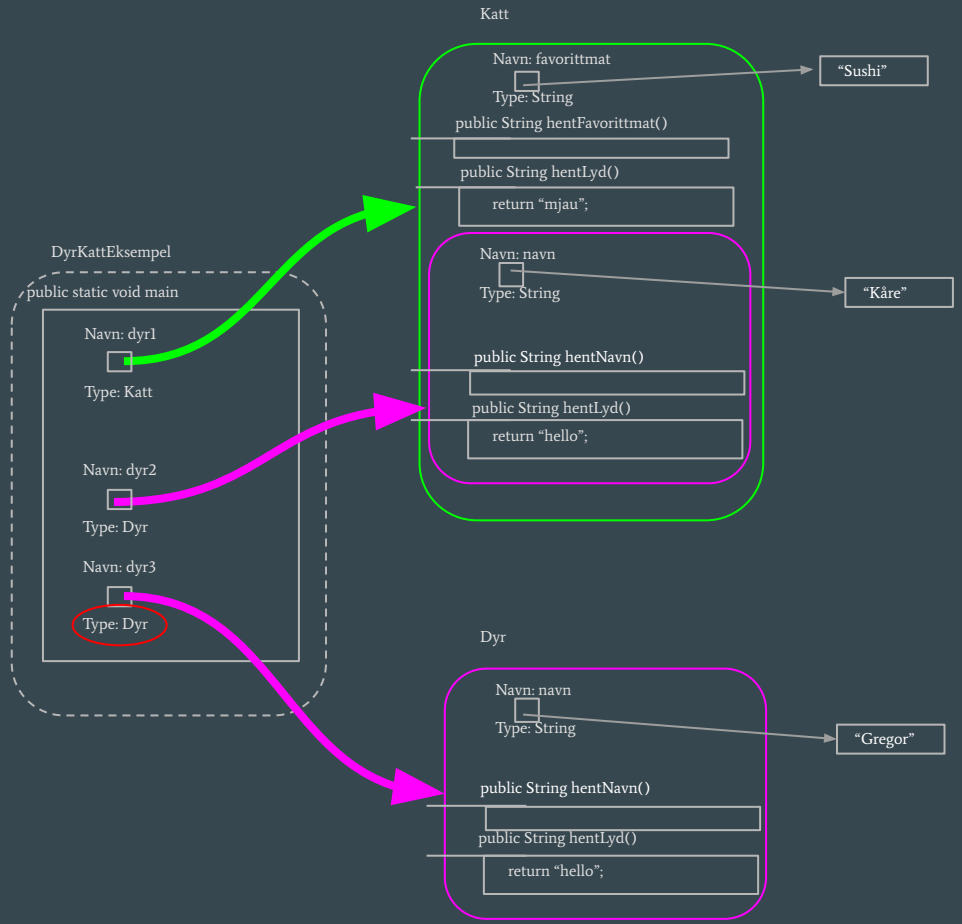
```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());          mjau
    System.out.println(dyr2.hentLyd());          mjau
    System.out.println(dyr3.hentLyd());          hello
    System.out.println(dyr1.hentNavn());         Kåre
    System.out.println(dyr2.hentNavn());         Kåre
    System.out.println(dyr3.hentNavn());         Gregor
    System.out.println(dyr1.hentFavorittMat());  Sushi
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

```
TestKatt.java:13: error: cannot find symbol
    System.out.println(dyr2.hentFavorittMat());
                       ^
  symbol:   method hentFavorittMat()
  location: variable dyr2 of type Dyr
```



# Casting

```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());          mjau
    System.out.println(dyr2.hentLyd());          mjau
    System.out.println(dyr3.hentLyd());          hello
    System.out.println(dyr1.hentNavn());         Kåre
    System.out.println(dyr2.hentNavn());         Kåre
    System.out.println(dyr3.hentNavn());         Gregor
    System.out.println(dyr1.hentFavorittMat());  Sushi
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

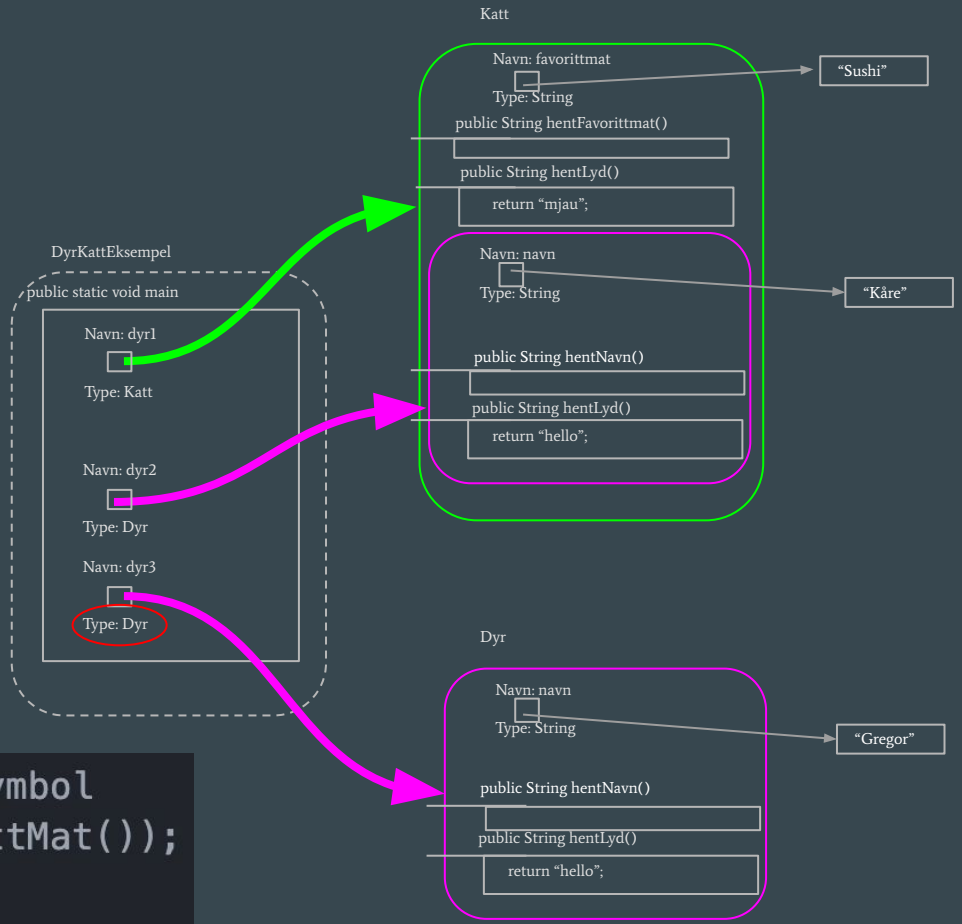



Vi må se på typen til referansen for å avgjøre hva vi kan aksessere.

# Casting

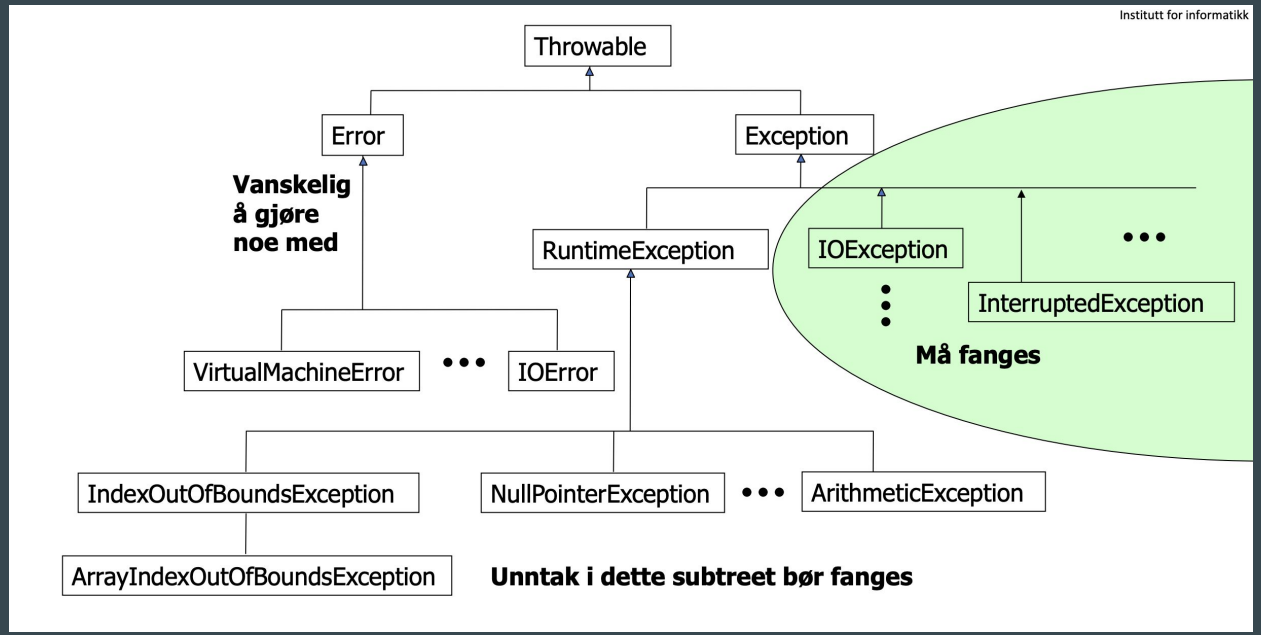
```
class TestKatt{
  public static void main(String[] args){
    Katt dyr1 = new Katt("Kåre", "Sushi");
    Dyr dyr2 = (Dyr) dyr1;
    Dyr dyr3 = new Dyr("Gregor");
    System.out.println(dyr1.hentLyd());          mjau
    System.out.println(dyr2.hentLyd());          mjau
    System.out.println(dyr3.hentLyd());          hello
    System.out.println(dyr1.hentNavn());         Kåre
    System.out.println(dyr2.hentNavn());         Kåre
    System.out.println(dyr3.hentNavn());         Gregor
    System.out.println(dyr1.hentFavorittMat());  Sushi
    System.out.println(dyr2.hentFavorittMat()); //ERROR
    System.out.println(dyr3.hentFavorittMat()); //ERROR
```

```
TestKatt.java:14: error: cannot find symbol
    System.out.println(dyr3.hentFavorittMat());
                       ^
  symbol:   method hentFavorittMat()
  location: variable dyr3 of type Dyr
```



# Feilmeldinger og feilhåndtering

Vi kan lage våre egne feilmeldinger eller bruke de som allerede finnes:



# Håndtere feil

1. Oppdag feil så tidlige som mulig, der du har mest mulig informasjon om hva feilen er.
2. Håndter feil der du har mest mulig informasjon om hva som er beste måte å håndtere feilen på.

# Eksempel

Vil gjøre sånn at du ikke kan gi et dyr en alder mindre enn null.

Et dyr kan du ikke ha alder -1 for eksempel.

# Lage ny feilmeldingsklasse

Lager en ny klasse for feilmeldingen min

```
1  class UlovligAlderException extends Exception{
2      public UlovligAlderException(int alder){
3          super("Alderen: " + alder + " er en ulovlig alder");
4      }
```

# Kaste en feilmelding (throw)

```
class Dyr{
  protected String navn;
  protected int alder;

  public Dyr(String navn, int alder) throws UlovligAlderException{
    if (alder < 0){
      throw new UlovligAlderException(alder);
    }
    this.navn = navn;
    this.alder = alder;
  }
  public String hentNavn(){
    return navn;
  }
  public String hentLyd(){
    return "hello";
```

# Sende en feilmelding videre (throws)

```
class Dyr{
    protected String navn;
    protected int alder;

    public Dyr(String navn, int alder) throws UlovligAlderException{
        if (alder < 0){
            throw new UlovligAlderException(alder);
        }
        this.navn = navn;
        this.alder = alder;
    }
    public String hentNavn(){
        return navn;
    }
    public String hentLyd(){
        return "hello";
```

# Ta i mot en feilmelding (catch)

```
1  class TestUlovligAlderException{
2      public static void main(String[] args) {
3
4          try{
5              Dyr dyr1 = new Dyr("Kåre", -1);
6          }catch(UlovligAlderException feilmelding){
7              System.out.println(feilmelding.getMessage());
8              System.exit(1);
9          }
```

# Kjøre koden

```
jonbon@jons-macbook-pro uke8 % java TestUlovligAlderException
Alderen: -1 er en ulovlig alder
```

# Jobbe selv

Jobb med hva dere vil og rekk opp hånda hvis dere trenger hjelp med noe/har spørsmål så møtes vi i breakoutroom! 😊