

# Klassebiblioteker & Dokumentation

## (tilfældige tal, maps, sets)

**GRPRO: "Grundlæggende Programmering"**

# Sem. Rep.?

- Semester repræsentanter?



# A G E N D A

- Klassebiblioteker og dokumentation
- Tilfældige tal
- *Map* og `HashMap`
- *Set* og `HashSet`
- *List* -vs- *Set* -vs- *Map*

Studere dette med udgangspunkt i eksemplet:

- Study "**tech-support**" project (IVR System)

# IVR System

- **The Simpsons,**  
*"Bart of Darkness"*  
(Season 6)



[Bart calls the police when he sees Flanders going after Lisa with an axe.]

**Voice>** Hello, and welcome to the Springfield Police Department **Resc-u-Fone**.  
If you know the name of the felony being committed, press "**one**".  
To choose from a list of felonies, press "**two**".  
If you are being murdered or calling from a rotary phone,  
please stay on the line.

**Bart>** [growls, punches some *random* numbers]

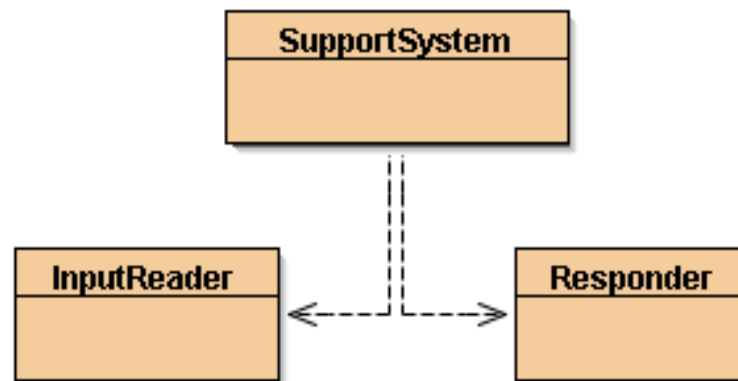
**Voice>** You have selected "**regicide**".  
If you know the name of the King or Queen being murdered, press "**one**", ...

# ArrayList<MyClass> -VS- MyClass[]

```
MyClass m1 = new MyClass("aaa", 1);  
MyClass m2 = new MyClass("bbb", 2);  
MyClass m3 = new MyClass("ccc", 3);
```

# Gennemgående eksempel: "Automatisk teknisk support"

- Projekt "tech-support" [Eliza]  
(med meget primitiv "kunstig intelligens")



- Illustrerer Java klassebiblioteker:
  - **Random** (tilfældige tal)
  - **HashSet** (uordnede mængder af objekter)
  - **HashMap** (afbildninger, tabeller)

# Metode start() i klassen SupportSystem

- Standard dialogløkke: Velkommen. Gentag replikskifte (indtil farvel). Farvel. Afslut.

```
public void start() {  
    boolean finished = false;  
    printWelcome();  
    while (!finished) {  
        String input = reader.getInput();  
        if (input.startsWith("bye")) {  
            finished = true;  
        } else {  
            String response = responder.generateResponse();  
            System.out.println(response);  
        }  
    }  
    printGoodbye();  
}
```

Fra klasse  
String

Hvordan ved vi hvad  
startsWith(..) gør?!

# Java's klassebiblioteker: Pakker, klasser, metoder

- Pakke `java.lang`:

- Klasse `String`:

- `int length()`
    - `String trim()`
    - ...

61 andre metoder i  
`String` klassen

- Klasse `Math`:

- `double sqrt(double a)`
    - `double cos(double a)`
    - ...

63 andre metoder i  
`Math` klassen

- ...

38 andre klasser i  
`java.lang` pakken

- Pakke `java.util`:

- Klasse `ArrayList<E>`:

- `E get(int index)`
    - `boolean add(E item)`
    - ...

- Klasse `Random`:

- `int nextInt()`
    - `int nextInt(int n)`
    - ...

- ...

- ...

Samt en fantasilion  
andre pakker



# Dokumentation til Javas klassebiblioteker

- **BlueJ:** Help -> Java Class Libraries

...or Google:  
"Java API"

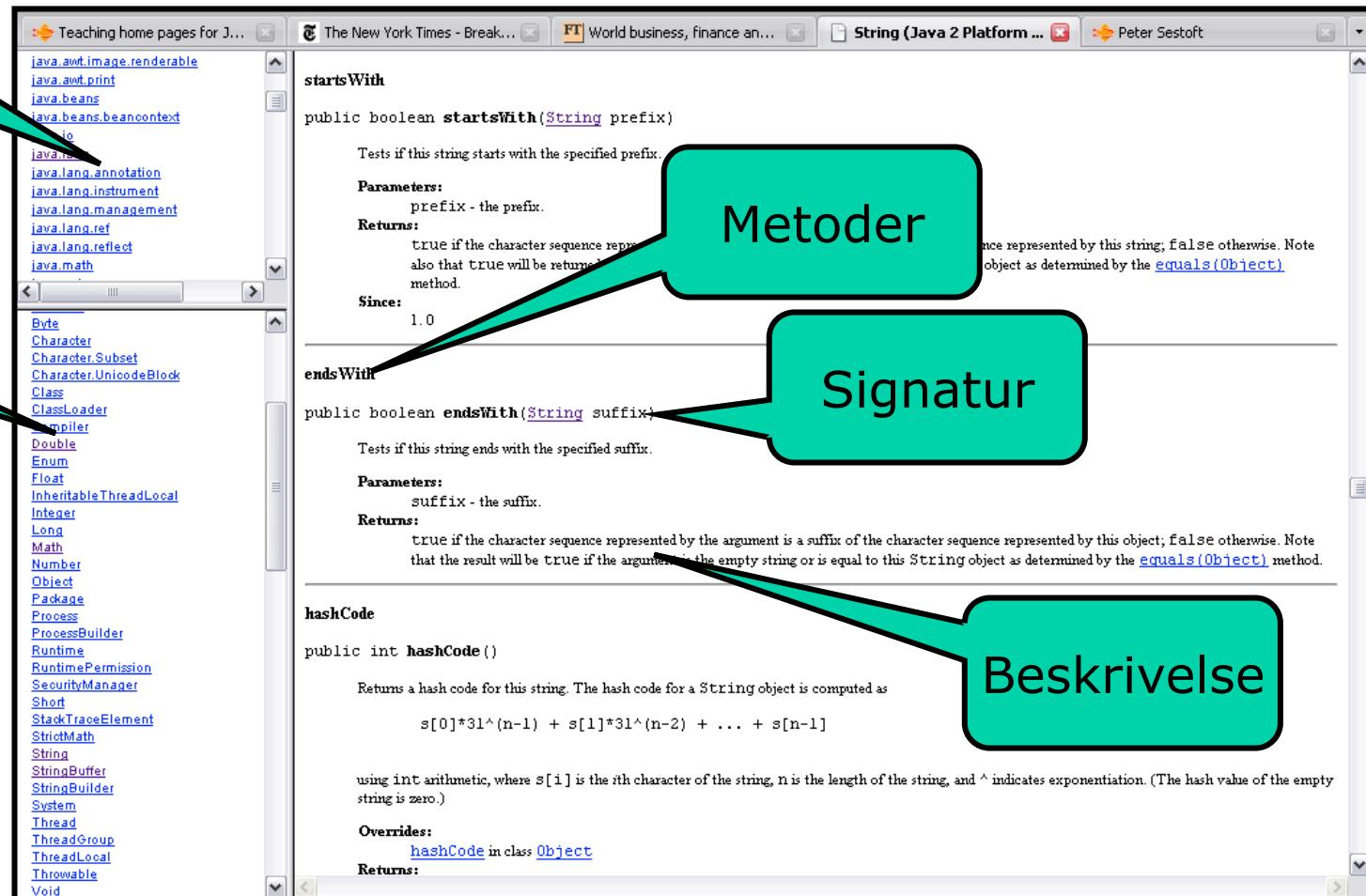
Pakker

Klasser

Metoder

Signatur

Beskrivelse

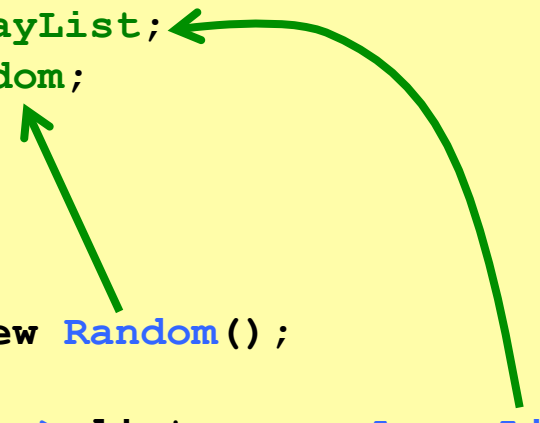


# Pakker og import

- Man skal importere de klasser man bruger:

```
import java.util.ArrayList;
import java.util.Random;

class MyClass {
    void myMethod() {
        ...
        Random rnd = new Random();
        ...
        ArrayList<String> list = new ArrayList<String>();
        ...
    }
}
```

A diagram with two green arrows. One arrow starts from the `ArrayList` in the `ArrayList<String>` line and points to the `import java.util.ArrayList;` line. The other arrow starts from the `Random` in the `Random rnd = new Random();` line and points to the `import java.util.Random;` line.

- Alternativt importerer *alle* pakkens klasser:

```
import java.util.*;

class MyClass { ... }
```

**NB: HOWEVER, ...**

# Pakker og import

- Men det kan være forvirrende:

```
import dk.itu.cool.util.*;
import min.gigantpakke.*;
import en.anden.pakke.*;
import oles.bizarre.klasser.*;
import some.other.classes.*;
...
import yet.another.set.of.classes.*;

class MyClass {
    void myMethod() {
        ...
        ObscureClass c = new ObscureClass();
        ...
    }
}
```

Hvilken **pakke** mon  
**ObscureClass**  
kommer fra **?!?**

# Metode generateResponse () i klassen Responder

- **Dumt 1:** Giver altid samme svar:

```
public String generateResponse() {  
    return "That sounds interesting. Tell me more...";  
}
```

- I stedet vælg et ***tilfældigt*** svar hver gang:
- Vælg et tal **index** blandt 0, 1, ..., n-1  
(brug det som indeks i en arraylist af svar):

```
public String generateResponse() {  
    int index = randomGenerator.nextInt(responses.size());  
    return responses.get(index);  
}
```

# Pseudo-tilfældige tal

Klassen `java.util.Random`:

- Pseudo-tilfældige tal (ikke rigtig tilfældige!)
  - `nextInt()` giver en tilfældig int
  - `nextInt(n)` giver et tilfældigt heltal mellem 0 og  $n-1$
- **Anvendelser:**
  - Simulering  
(trafik, plantevækst, dyrs adfærd, indkøbsmønstre, ...)
  - Statistiske analyser
  - Kryptering (primtalstest)
  - Computerspil
  - Objekt genkendelse
- Dokumentation: Help -> Java Class Libraries

# Hele klassen Responder

- Klasse **Responder** har to felter:
  - **Random** randomGenerator (til at lave tilfældige tal)
  - **ArrayList<String>** responses (er en liste af (u)mulige svar)

```
import java.util.Random;
import java.util.ArrayList;

public class Responder {
    private Random randomGenerator;          // init'd in constructor (not shown)
    private ArrayList<String> responses;     // init'd in constructor (not shown)

    private void fillResponses() {
        responses.add("That sounds odd. Could you ... ");
        responses.add("No other customer has ever ... ");
        responses.add("That sounds really interesting.");
        ...
    }

    public String generateResponse() {
        int index = randomGenerator.nextInt(responses.size());
        return responses.get(index);
    }
}
```

# Tjekopgaver: "RandomTester"

## [B&K] 5.14:

- *"Write some Java code to test random number generation..."*

## [B&K] 5.16:

- *"Implement, `throwDice()`, returning a random number 1-6."*

## [B&K] 5.17:

- *"Implement method that randomly returns 'yes', 'no', 'maybe'."*

## [B&K] 5.18:

- *"Extend this to arbitrary number of responses (from ArrayList) and randomly return one of them."*

# Endnu bedre "tech-support"

- **Dumt 2:**  
Ingen sammenhæng mlm spørgsmål/svar:
- **Idé:** Lad bestemte **nøgleord** styre hvilket svar der vælges:
  - Fx vil ordet "**s1ow**" medføre et svar om at:  
*det nok er hardwaren det er galt med*
  - Fx vil ordet "**bug**" medføre et svar om at:  
*det ikke er en bug, men en feature*
  - ...
- Dette kan implementeres med en **HashMap**



# Map: `HashMap<String, String>`

- Map = afbildning (tabel) nøgler  $\rightarrow$  værdier
- En `HashMap<K, V>` har nøgler af type `K` og værdier af type `V`
- Kaldet `put(k, v)` tilføjer nøgle `k` med værdi `v`
- Kaldet `get(k)` finder værdien hørende til nøgle `k` hvis der er en (ellers `null`)

Nøgle $\rightarrow$	Værdi
"slow"	"I think this has to do with your hardware..."
"bug"	"Well, you know, all software has some bugs..."
"expensive"	"The cost of our product is quite competitive..."
...	...

# Forbedret generateResponse() metode i klasse Responder

Gå gennem input-ord, et efter et

en mængde af input-ord

```
public String generateResponse(HashSet<String> words) {  
    for (String word : words) {  
        String response = responseMap.get(word);  
        if (response != null) {  
            return response;  
        }  
    }  
    return pickDefaultResponse();  
}
```

slå input-ordet  
op i HashMap

hvis input-ordet findes,  
returner tilhørende svar

hvis ingen af input-ordene findes i  
hashmap, giv et tilfældigt svar

# Debugger



- Afvikling (execution) en linie ad gangen:
  - Debuggeren blokerer når Java-programmet venter på input
- "Step":  
Udfør næste linje (inkl. hele metodekald)
- "Step Into":  
Udfør næste linje (gå ind i metodekald)  
kaldet fra `start()` til `generateResponse()`

# Fælles tjekopgaver

## [B&K] 5.25 (& 5.31):

- *How do you check how many entries are contained in a map?*

## [B&K] 5.26:

- *Implement `MapTester`; a "phone book" with two methods:*

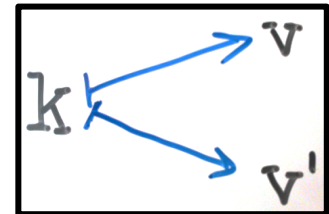
- `void enterNumber(String name, String number)`

- `String lookupNumber(String name)`

[use `HashMap`]

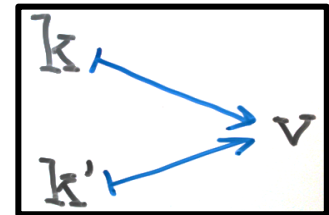
## [B&K] 5.27:

- *What happens when you add the same key for two diff vals?*



## [B&K] 5.28:

- *What happens when you add the same val with two diff keys?*

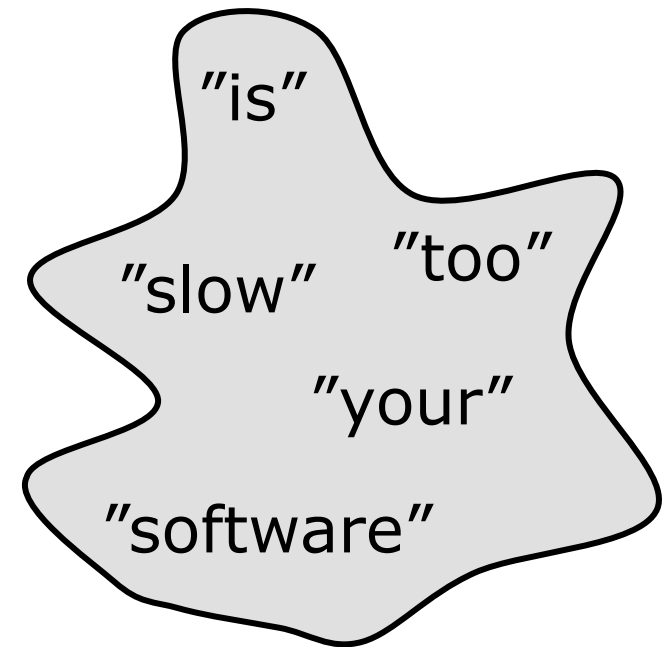


## [B&K] 5.29:

- *How do you check whether a given key is in the map?*

# SET: `HashSet<String>`

- Set = *mængde* (af værdier)
  - **NB:** Værdierne er uordnede!
  - **NB':** Ingen gentagelser!
- Et `HashSet<V>` har værdier af type `V`
- Kaldet `add(v)` tilføjer værdi `v` hvis den ikke allerede er der
- Kaldet `contains(v)` giver `true` hvis `v` er i mængden, ellers `false`



# Find mængden af ord i input

Næste inputlinje,  
med små bogstaver

... splittet op i input-ord,  
lagret i et array

Brug JavaDoc

```
public HashSet<String> getInput() {  
    String inputLine = reader.nextLine().trim().toLowerCase();  
    String[] wordArray = inputLine.split(" ");  
    HashSet<String> words = new HashSet<String>();  
    for (String word : wordArray) {  
        words.add(word);  
    }  
    return words;  
}
```

Opret tom  
mængde

Fyld input-ordene i  
mængden et ad gangen

Returner mængden  
af input-ord

# Opdeling af tegnstreng

- Vi starter med en:

- String

"your software is too slow"
-----------------------------

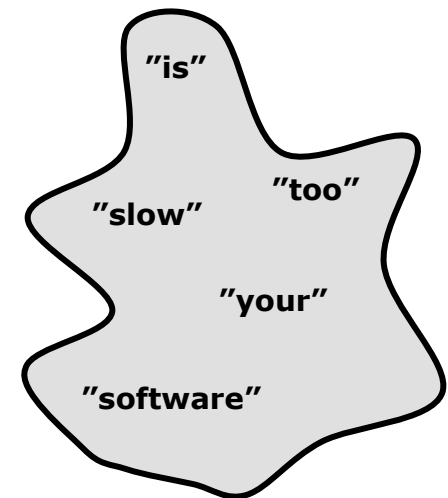
- `split("your software is too slow")`

- String[]

"your"	"software"	"is"	"too"	"slow"
--------	------------	------	-------	--------

- Foreach: insert words into:

- HashSet<String>



# Forbedret metode start() i klassen SupportSystem

```
public void start() {  
    boolean finished = false;  
    printWelcome();  
    while (!finished) {  
        HashSet<String> input = reader.getInput();  
        if (input.contains("bye")) {  
            finished = true;  
        } else {  
            String response = responder.generateResponse(input);  
            System.out.println(response);  
        }  
    }  
    printGoodbye();  
}
```

Mængden af input-ord  
gives til svar-funktionen

- Nu bliver ordene fra input brugt til at vælge svar (men stadig ret primitivt)



# Automatiske dialogsystemer

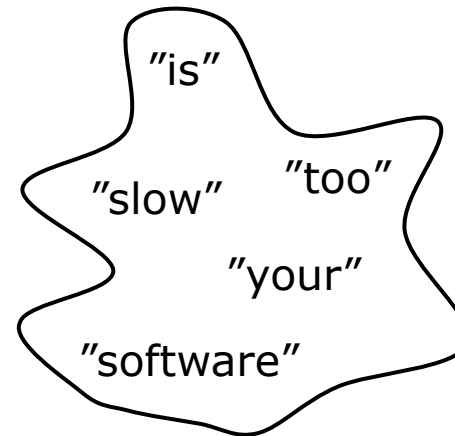
- IVR (Interactive Voice Response) Systems:
- Mest stemmestyrede systemer  
(Feriekonto, Generel Vejledning, ...)
- Ring 70 11 49 10 tast 5, så får man et  
automatisk dialogsystem man kan tale til
  - (Udviklet af Prolog Development Center i Brøndby)

# Forskellen på List, Set og Map

- **List:** Nummereret, dubletter tilladt

"your"	"software"	"is"	"too"	..
--------	------------	------	-------	----

- **Set:** Uordnet, ingen dubletter



- **Map:** Afbildning fra nøgle -> værdi

Nøgle	Værdi
"slow"	"I think this has to do with your hardware..."
"bug"	"Well, you know, all software has some bugs..."
...	...

# Forskellige slags lister, set og map - og gennemløbsordenen med foreach

<b>List</b>	<div>Indsættelsesorden</div> <b>ArrayList&lt;E&gt;</b>	
<b>Set</b>	<div>Uforudsigelig orden</div> <b>HashSet&lt;E&gt;</b>	<div>Voksende værdi orden</div> <b>TreeSet&lt;E&gt;</b>
<b>Map</b>	<div>Uforudsigelig orden</div> <b>HashMap&lt;K, V&gt;</b>	<div>Voksende nøgle orden</div> <b>TreeMap&lt;K, V&gt;</b>

# Gennemløbsordenen HashSet og TreeSet

- Lav mængde med 10 til 19 og udskriv

```
HashSet<Integer> hashSet = new HashSet<Integer>();  
for (int i=10; i<20; i++) {  
    hashSet.add(i);  
}  
for (int i : hashSet) {  
    System.out.print(i + " ");  
}
```

- Gennemløb af HashSet:

17 16 19 18 10 11 12 13 14 15

- Gennemløb af TreeSet:

10 11 12 13 14 15 16 17 18 19

# Kortere = bedre?

```
public HashSet<String> getInput() {  
    String inputLine = reader.nextLine().trim().toLowerCase();  
    String[] wordArray = inputLine.split(" ");  
    HashSet<String> words = new HashSet<String>();  
    for (String word : wordArray) {  
        words.add(word);  
    }  
    return words;  
}
```

Bedre?

```
public HashSet<String> getInput() {  
    String inputLine = reader.nextLine().trim().toLowerCase();  
    String[] wordArray = inputLine.split(" ");  
    HashSet<String> words  
        = new HashSet<String>(Arrays.asList(wordArray));  
    return words;  
}
```

Bedst?

```
public HashSet<String> getInput() {  
    return new HashSet<String>(Arrays.asList(  
        reader.nextLine().trim().toLowerCase().split(" ")));  
}
```

## En forbedring, side 155 (4. edition)

```
public String generateResponse(HashSet<String> words) {  
    Iterator<String> it = words.iterator();  
    while (it.hasNext()) {  
        String word = it.next();  
        String response = responseMap.get(word);  
        if (response != null) {  
            return response;  
        }  
    }  
    ...  
}
```

kan skrives pænere med foreach:

```
public String generateResponse(HashSet<String> words) {  
    for (String word : words) {  
        String response = responseMap.get(word);  
        if (response != null) {  
            return response;  
        }  
    }  
    ...  
}
```

**Tak**

Spørgsmål ?