Lecture 8 and 9

Software developing in organizations

Ways for achieving data persistence between different runs of your program

Serialization to Databases

Failure

- By some estimates over 55% of software projects fail, often this is found to be because of human factors
- Failure means not satisfying requirements, delivering on schedule, or not meeting quality standards (e.g. too many bugs)
- There are many reasons that are suggested for why this is the case (besides competence/incompetence): organization culture, management, poor planning, etc
- Organizational maturity is a model for characteristics of firms/organizations developing software and their processes: those with higher maturity have more effective processes and fail less often

Capability Maturity Model Integrated

Initial

- Heroics

Repeatable

- Project management

Defined

- Engineering process

Managed

- Product and process quality

Optimizing

- Continuous improvement

Enterprise Software Developing

- Processes…
- Scaled Agile Framework (SAFe)
- Most large software organizations use some [illegible]milar derivates of SAFe
- **The goal was to integrate a range of different innovations in management and processes:** agile, lean, DevOps, test driven development, continuous delivery and release on demand, vision, prioritization, aligning strategy with operations, continuous learning culture.
- Approach for managing large scale software development in organizations (100s + of people): large applications, networks, cyber-physical solutions

What is DevOps?

DevOps unifies people, process, and technology to bring better products to customers faster.

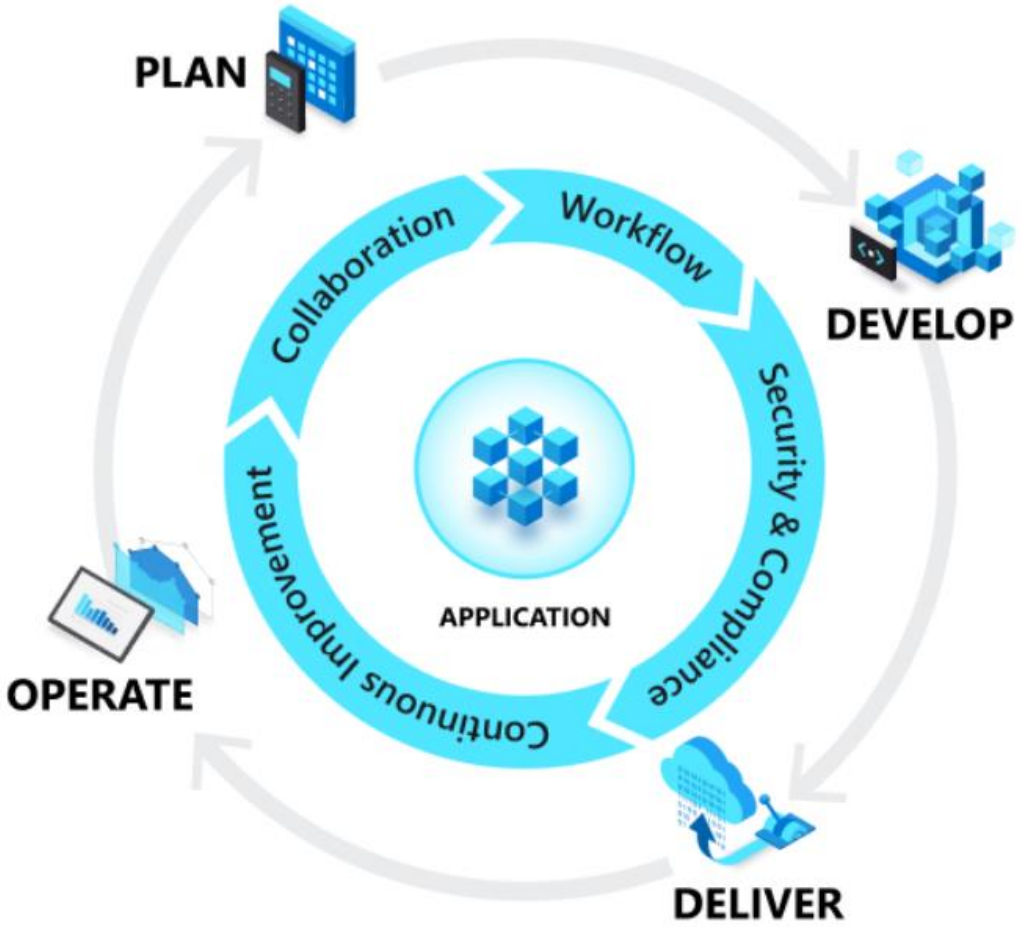
What does DevOps mean for teams? DevOps enables formerly siloed roles—development, IT operations, quality engineering, and security—to coordinate and collaborate to produce better, more reliable products. By adopting a DevOps culture along with DevOps practices and tools, teams gain the ability to better respond to customer needs, increase confidence in the applications they build, and achieve business goals faster.

- Build Faster
- Adapt to the market
- Maintain reliability
- Improve the time to recover from issues

DevOps Application Life Cycle

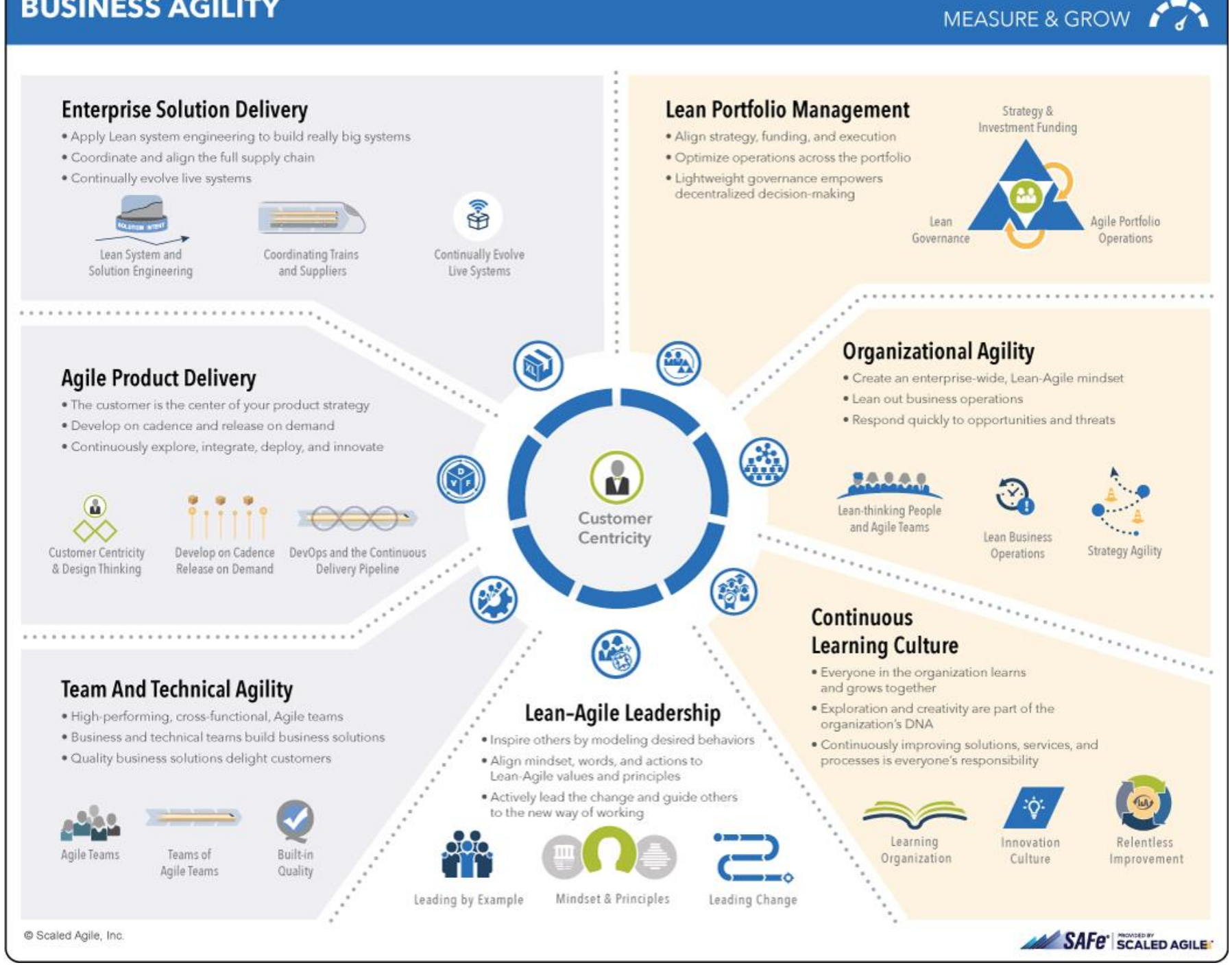
Each phase relies on the others, and the phases are not role-specific. In a true DevOps culture, each role is involved in each phase to some extent.

- **Plan**: In the plan phase, DevOps teams ideate, define, and describe features and capabilities of the applications and systems they are building.
- **Develop:** The develop phase includes all aspects of coding—writing, testing, reviewing, and the integration of code by team members
- **Deliver**: Delivery is the process of deploying applications into production environments in a consistent and reliable way. The deliver phase also includes deploying and configuring the fully governed foundational infrastructure that makes up those environments.
- **Operate**: The operate phase involves maintaining, monitoring, and troubleshooting applications in production environments.

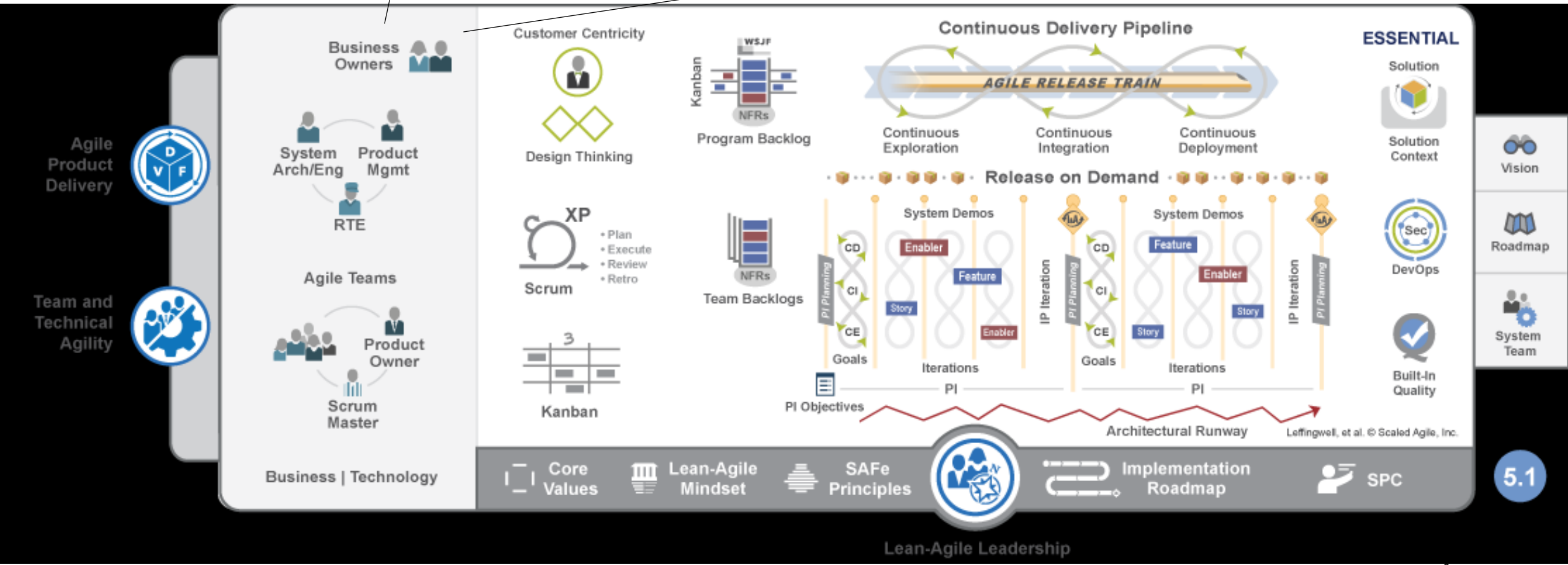


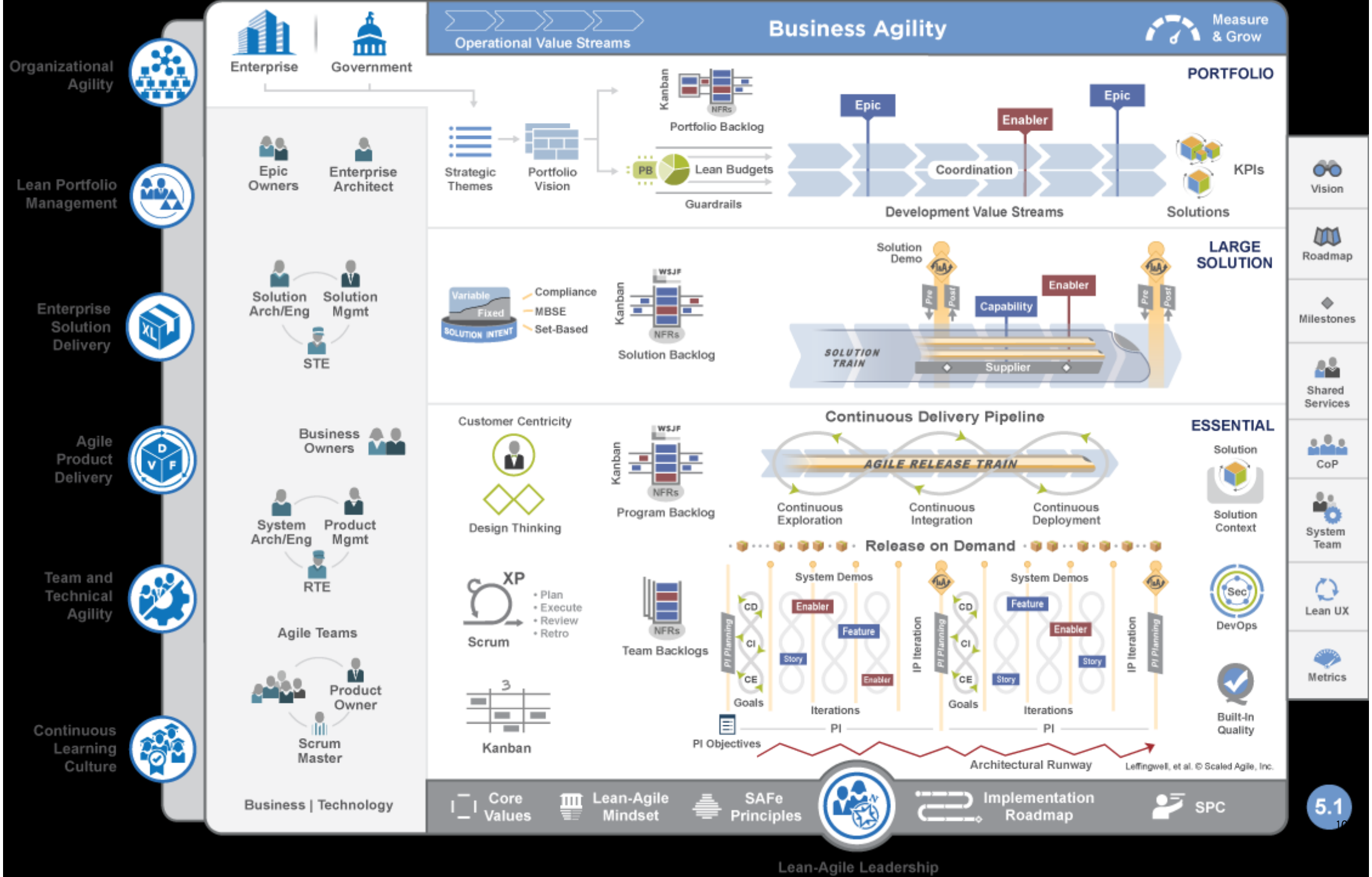
Roles…

Scaled Agile Framework



Business Owners: It is hard to imagine a more stupid or more dangerous way of making decisions than by putting those decisions in the hands of people who pay no price for being wrong.



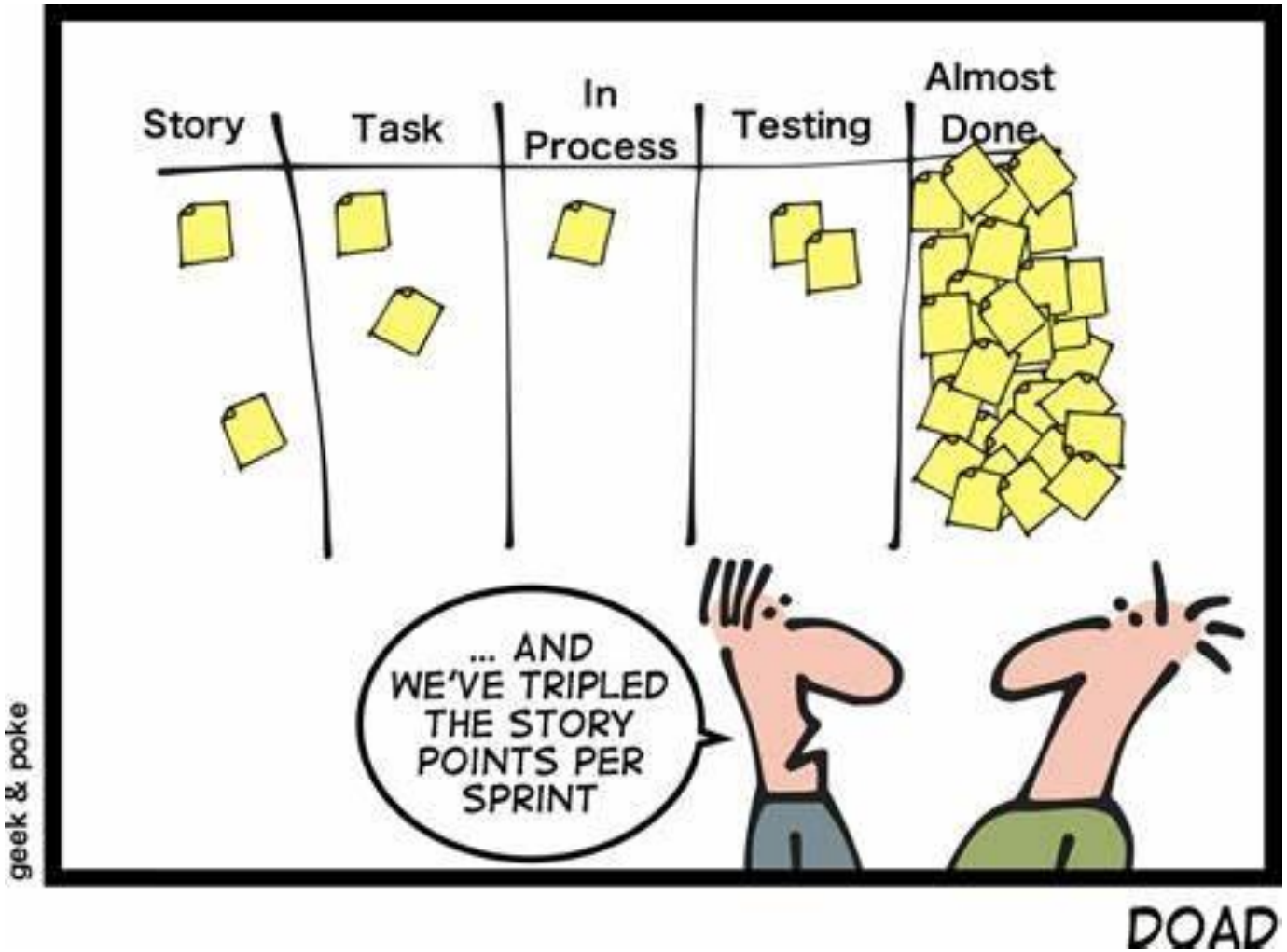

Organizational Agility
Lean Portfolio Management
Enterprise Solution Delivery
Agile Product Delivery
Team and Technical Agility
Continuous Learning Culture
Enterprise
Government
Operational Value Streams
Business Agility
Measure & Grow
PORTFOLIO
Epic Owners
Enterprise Architect
Strategic Themes
Portfolio Vision
Kanban
NFRs
Portfolio Backlog
PB
Lean Budgets
Guardrails
Epic
Enabler
Epic
Coordination
Development Value Streams
KPIs
Solutions
Solution Arch/Eng
Solution Mgmt
STE
Variable
Fixed
SOLUTION INTENT
Compliance
MBSE
Set-Based
WSJF
Kanban
NFRs
Solution Backlog
LARGE SOLUTION
Solution Demo
Pre
Post
Capability
Enabler
SOLUTION TRAIN
Supplier
Business Owners
Customer Centricity
Design Thinking
Kanban
WSJF
NFRs
Program Backlog
Continuous Delivery Pipeline
AGILE RELEASE TRAIN
Continuous Exploration
Continuous Integration
Continuous Deployment
ESSENTIAL
Solution
Solution Context
System Arch/Eng
Product Mgmt
RTE
XP
Plan
Execute
Review
Retro
Scrum
Agile Teams
NFRs
Team Backlogs
Release on Demand
System Demos
CD
CI
CE
Enabler
Feature
Story
PI Planning
IP Iteration
Goals
Iterations
PI
PI Objectives
Architectural Runway
Sec
DevOps
Built-In Quality
Product Owner
Scrum Master
3
Kanban
Business | Technology
Core Values
Lean-Agile Mindset
SAFe Principles
Implementation Roadmap
SPC
Lean-Agile Leadership
Vision
Roadmap
Milestones
Shared Services
CoP
System Team
Lean UX
Metrics
Leffingwell, et al. © Scaled Agile, Inc.
5.1

Benefits they claim:

- 20 – 50% increase in productivity
- 25 – 75% improvements in quality
- 30 – 75% faster time-to-market
- 10 – 50% increase in employee engagement and job satisfaction

https://www.scaledagileframework.com/about/


YOUR COMPETITORS ARE FASTER BECAUSE THEY HAVE MEETINGS WHERE EVERYONE HAS TO STAND UP.
WE'LL TOP THAT BY HAVING MEETINGS WHERE EVERYONE DOES JUMPING JACKS WHILE I PELT THEM WITH OFFICE SUPPLIES.
IT'S WORKING!
Dilbert.com DilbertCartoonist@gmail.com
3-11-11 ©2011 Scott Adams, Inc./Dist. by UFS, Inc.


Story
Task
In Process
Testing
Almost Done
... AND WE'VE TRIPLED THE STORY POINTS PER SPRINT
geek & poke
DOAD

Story points

- Velocity is a metric that measures the amount of work completed per time unit by each worker (team member / software developer)
- Story points used to measure velocity (story points / day or week etc) and are units of measure for expressing an estimate of the overall effort required to fully implement a product backlog item or any other piece of work. Teams assign story points relative to work complexity, the amount of work, and risk or uncertainty
- Why story points and not time or dates?
- Because as the team gets more experience they complete more story points
- Story points reward team members for solving problems based on difficulty, not time spent. This keeps team members focused on shipping value, not spending time

JavaFX (Cont.)

Multimedia

Media

You can also play audio and video in JavaFX. It's not very different from images. With images you have an Image object, and the ImageView that shows it on screen. With audio and video, you have a Media object, you have a MediaView, and between the two you have a MediaPlayer object with controls allowing you to start, pause, stop, rewind and so forth.

Very much like Images

- Media
- MediaPlayer  Controls
- MediaView

```
import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.layout.*;
import javafx.geometry.Pos;
import javafx.util.Duration;

import javafx.scene.media.Media;
import javafx.scene.media.MediaPlayer;
import javafx.scene.media.MediaView;
```

```
14  public class MediaDemo extends Application {
15
16  private final String MEDIA_URL = this.getClass()
17                                       .getClassLoader()
18                                       .getResource("TestVid.mp4")
19                                       .toString();
20
21
22  private final String MEDIA_URL = "http://edu.konagora.com/video/TestVid.mp4";
```

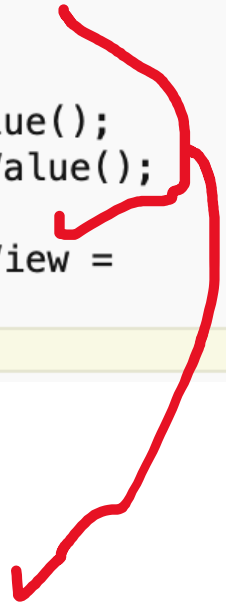
OR

URL, Path and String

- URL: prefix://path
- PATH: path

A Media (like an Image) can take a URL as argument of a constructor. An URL (like an URI, basically the same thing) is a prefix + a path. If the prefix is "file:" it means that the resource (... name given to anything you can load) is accessible through the file system of your computer (it's not necessarily local, it can be a network disk). It can also be something else such as "http:" to mean that the resource is accessed from a web server through HTTP requests. You usually need to apply toString() to them.

```
21
22  private final String MEDIA_URL = "http://edu.konagora.com/video/TestVid.mp4";
23
24
25  @Override
26  public void start(Stage primaryStage) {
27      Media media = new Media(MEDIA_URL);
28      int width = media.widthProperty().intValue();
29      int height = media.heightProperty().intValue();
30      MediaPlayer mediaPlayer =
31      new MediaPlayer(media); MediaView mediaView =
32      new MediaView(mediaPlayer);
33      Button playButton = new Button(">");
```



Once the media is loaded, you associate it with a MediaPlayer, and the MediaPlayer with a MediaView. Controls will execute MediaPlayer methods.

```
new MediaView(mediaPlayer);
Button playButton = new Button(">");

playButton.setOnAction(e -> {
if (playButton.getText().equals(">")) {
        mediaPlayer.play();
        playButton.setText("||"); }else{
        mediaPlayer.pause();
        playButton.setText(">"); }
        });

Button rewindButton = new Button("<<");
rewindButton.setOnAction(e->
        mediaPlayer.seek(Duration.ZERO));

Slider slVolume = new Slider();
```

The text on the button tells us what is the current state, and whether we should play or pause.
I'm also adding another button for rewinding, and a new widget (Slider) for setting the volume.

```
54
55      HBox hBox = new HBox(10);
56      hBox.setAlignment(Pos.CENTER); hBox.getChildren().addAll(playButton,
57                  rewindButton,
58      new Label("Volume"), slVolume);
59      BorderPane pane = new BorderPane();
60
61      pane.setCenter(mediaView);
62      pane.setBottom(hBox);
63      Scene scene = new Scene(pane, 750, 500);
64      primaryStage.setTitle("MediaDemo");
65      primaryStage.setScene(scene);
66      primaryStage.show();
67      }
68
69
70      public  void main(String[] args) {
71          launch(args);
72      }
73  }
74
75
```

Other than geometry (size) I give the range and initial value for the slider (0 to 100, initially 50) and "bind" it to the MediaPlayer. There is an implicit ChangeListener behind, to change the volume when the slider moves.

Controls are in a box, everything is added to a BorderPane (that controls placement as top/right/bottom/left and center) and we are ready to go.

This demo and the slides are by Stephane Faroult


MediaView
MediaDemo
MediaPlayer
The Media is added to a MediaPlayer that provides methods for controlling the media (playing it, pausing it and so forth). The MediaPlayer is in turn added to a MediaView which is what JavaFx can display.
Media
I have in my demo application added to a "border Pane" the MediaView and a Hbox (Horizontal box) that contains widgets that call the MediaPlayer methods.
Hbox for controls

Style Sheets

Skins

- Last JavaFx subject, superficial in all meanings of the word but important (looks sell): how to change the appearance of a JavaFx application. I have already briefly talked about it, the best way is to do it through an external style sheet (.css file). People will be able to change, often in a very impressive way, the looks of your application by changing this file and without any need to access the code (in fact, they just need the .css and the .class to run the "modified" application).
- If you want to see how far you go with "styling", you can visit http://csszengarden.com and click on designs on the right hand- side. The same page will look completely different.

CSS ZEN GARDEN

The Beauty of CSS Design

Select a Design:

Mid Century Modern by Andrew Lohman

Garments by Dan Mall

Steel by Steffen Knoeller

Apothecary by Trent Walton

Screen Filler by Elliot Jay Stocks

Fountain Kiss by Jeremy Carlson

A Robot Named Jimmy by meltmedia

Cascading Style Sheet (CSS)

CSS stands for Cascading Style Sheet and Cascade is French for waterfall.
The name emphasizes that styles "drip down”, and are inherited from previous style sheets

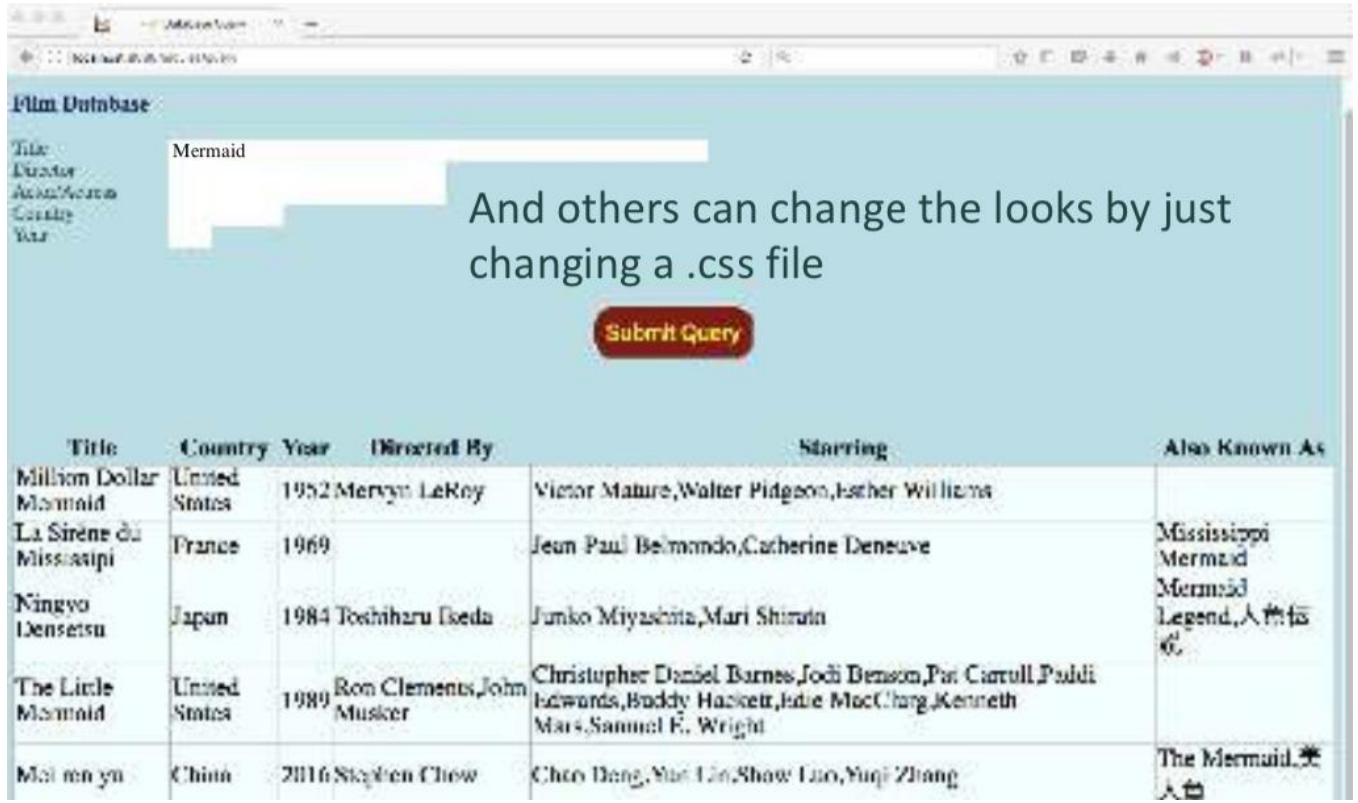
It all starts with HTML…

The idea is stolen from web applications. Web applications just send HTML pages to browsers that decode and display these pages. An HTML page is organized by sections between pairs of tags (<tag> at the beginning, </tag> at the end) that structure the document and can contain in turn other pairs of tags, thus defining a kind of hierarchical structure (note that some tags, such as those for images, act both as opening and closing tags). Tags pretty often also contain attributes (such as the image file name for an image tag).

<tags>

In the very early days of the web, people were using tags to format their pages, for instance what they wanted in bold was between **<bold>** and **</bold>** (inspired by previous document generation systems that were sending special signals to printers), and you could change the fonts with **<font** *attributes specifying the font, size and everything***>** ... ****. As websites were growing in size and number of pages, and as increasingly pages were generated by programs instead of being created by hand, it became unmanageable, especially when the marketing department was deciding on new corporate colors. So the idea was to associate formatting to tags in one or several separate text files.

Some people could just focus on the "engine", such as here an application allowing to retrieve film information from a database ...



And others can change the looks by just changing a .css file

The way it works in web pages

- Before I discuss about CSS in JavaFx, I'm going to talk about CSS with HTML, because there is far more CSS written for HTML than for JavaFX.
- There are three main ways to specify how to display what is between tags.
- 1. You can specify for a given tag, associating a tagname with visual characteristics
  2. I have mentioned that tags can have attributes, one is "class" (unrelated to object-oriented programming) listing one or several categories. This allows for creating a subcategory, or to give some common visual characteristics across different tags that have the same class (for instance "inactive")
- 3. You can give another attribute "id" to a tag, an this allows you to make one particular tag look really special.

```
<tag>
```

```
tag {
    attribute: value;
    ...
}
```

I focus on simple tags in the next example.

```
<tag class="category">
```

You have here how the tag looks in HTML and how styling looks in CSS.

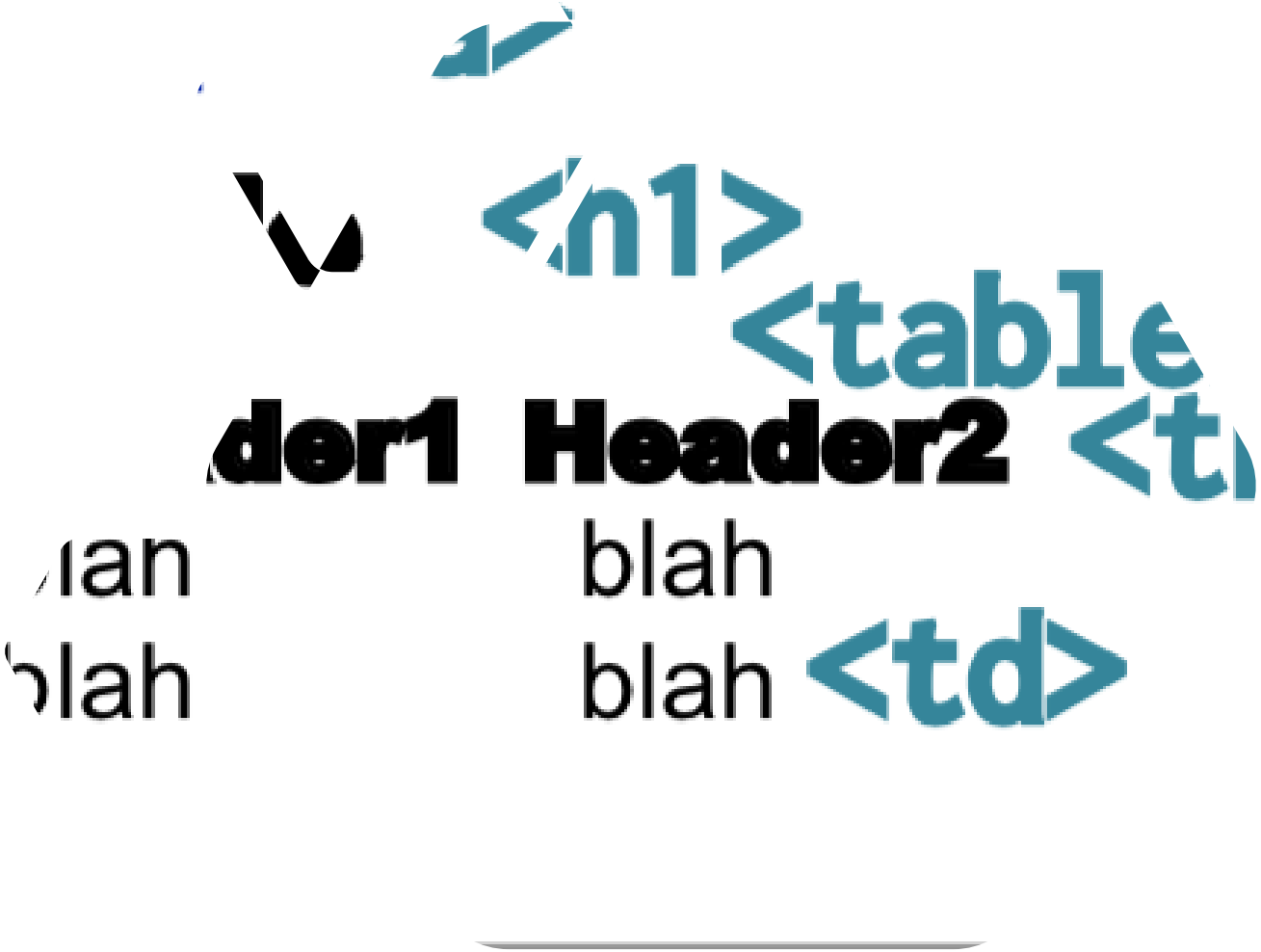
```
.category {
    attribute: value;
    ...
}
```

```
<tag id="name">
```

```
#name {
    attribute: value;
    ...
}
```

<body>

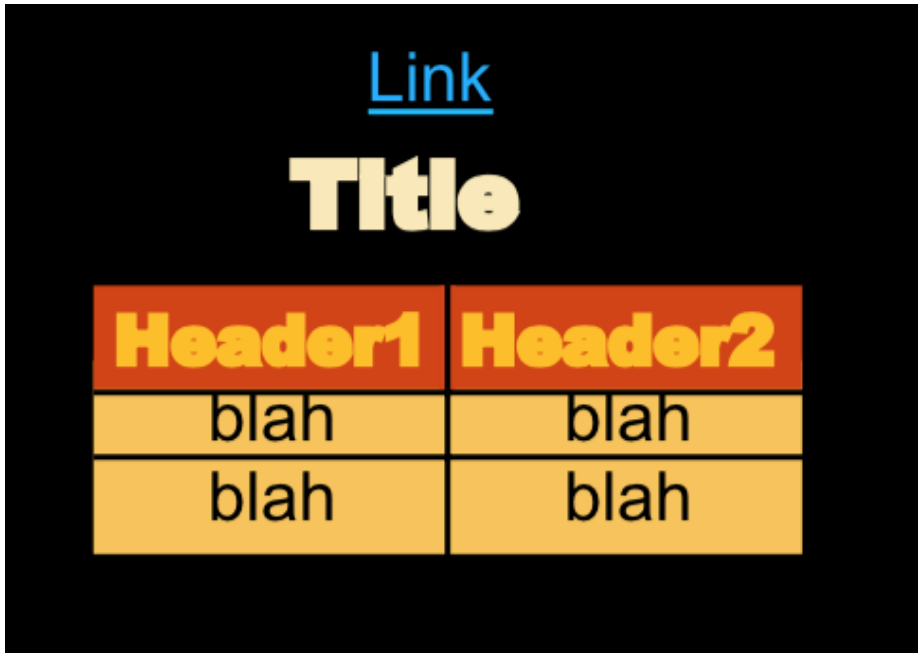
- In a HTML page, everything that is displayed is between <body> and </body> tags, so <body> really defines the global looks of the page.
- In my example, I'll use tags <a> associated with a link, <h1> with a (first level) header, <table>, <th> (table header) and <td> (table data). I have omitted <tr> (table row) which usually surrounds the columns of a same row. The tag will be there on the page, but I'm not associating any special style with it in my example.



Notice the special a:visited (link you have already clicked on) and a:hover (when the cursor moves over the link)

```
body {text-align: center;
        background-color: black;}
a {color: lightskyblue;}
a:visited {color: lightseagreen;}
a:hover {color: salmon;}
h1 {color: cornsilk;}
th {background-color: sienna;
    color: orange;}
td {background-color: burlywood;}
table {width: 80%;
    margin-left: auto;
    margin-right: auto;
    text-align: center;}
form {width: 50%;
        margin-left: auto;
        margin-right: auto;
        padding: 20px;
        background-color: silver;}
```

styles.css



<link rel="stylesheet" href="styles.css"/>

Styling is written to a file that is included in the HTML page by a special tag in the <header>...</header> section that precedes the "body". Then you can change it when needed.

The way it works in JavaFX

Nodes are (almost) equivalent to tags

- .root plays the same role as body
- Otherwise use class names prefixed with a dot
- .button

You have of course no tags in a JavaFx application, but you have the same kind of hierarchy through nodes, containers and widgets. The JavaFx class names are used with the same syntax as the HTML classes in CSS, that means that they are prefixed by a dot.

The way it works in JavaFX

Nodes are (almost) equivalent to tags

.root plays the same role as body

Otherwise use class names prefixed with a dot

.button

Attribute names are prefixed with –fx-

-fx-font-size: 150%;

CSS attributes also have a special name with JavaFx. The idea is to be able to have a single CSS files shared by a Web and a JavaFx application without having any conflict.

The way it works in JavaFX

Nodes are (almost) equivalent to tags

.root plays the same role as body

Otherwise use class names prefixed with a dot

.button

Attribute names are prefixed with –fx-

-fx-font-size: 150%;

Node.setId("name")

Node.getStyleClass().add("css class")

Finally, node methods allow you to associate with a node the same kind of attributes as with a HTML tag. You can only have a single Id, but you can have several classes, and therefore getSyleClass() returns a list.

The way it works in JavaFX

Nodes are (almost) equivalent to tags

.root plays the same role as body

Otherwise use class names prefixed with a dot

.button

Attribute names are prefixed with –fx-

-fx-font-size: 150%;

Node.setId("name")

Node.getStyleClass().add("css class")

Node.setStyle("-fx-attribute: value")

```
Scene scene = new Scene(new Group(), 500, 400);
scene.getStylesheets().add("path/styles.css");
```

(then load into the application)

Not Everything cannot be styled with CSS in JavaFX

CSS styling allows you to go rather far in JavaFx, but not as far as you could go in HTML. Some elements may prove hard to style with CSS. You may sometimes have to code some styling in the Java application. However, if you still want this styling to be "externalized", don't forget that properties files also provide a way to read attributes at run-time. It's of course better to have all styling at one place, but it's better to use a properties file than to hard-code.

A Properties file might sometimes be a workaround

Persistence and Memory

Lecture 8 – Part 1

Persistence

New Topic :-)

Java persistence refers to programs which have elements that can “persist”

The term is also sometimes used to refer specifically to the practice of storing and retrieving java program elements (e.g. objects and methods) using the Jakarta Persistence API (was renamed from the Java Persistence API to the current name last year)

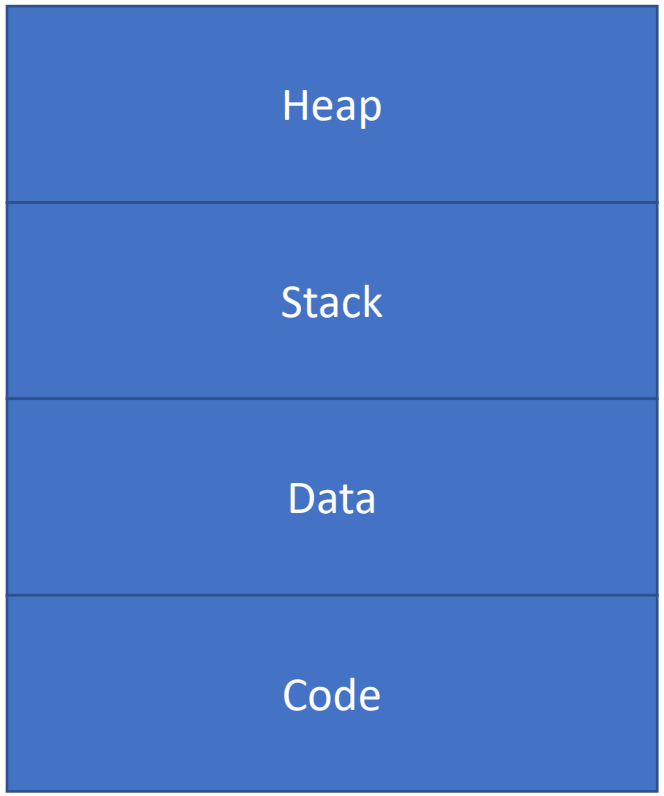
Data Persistence (数据坚持 shù jù jiān chí)

Per - Sistere

Per means thoroughly
Sistere means to stand still

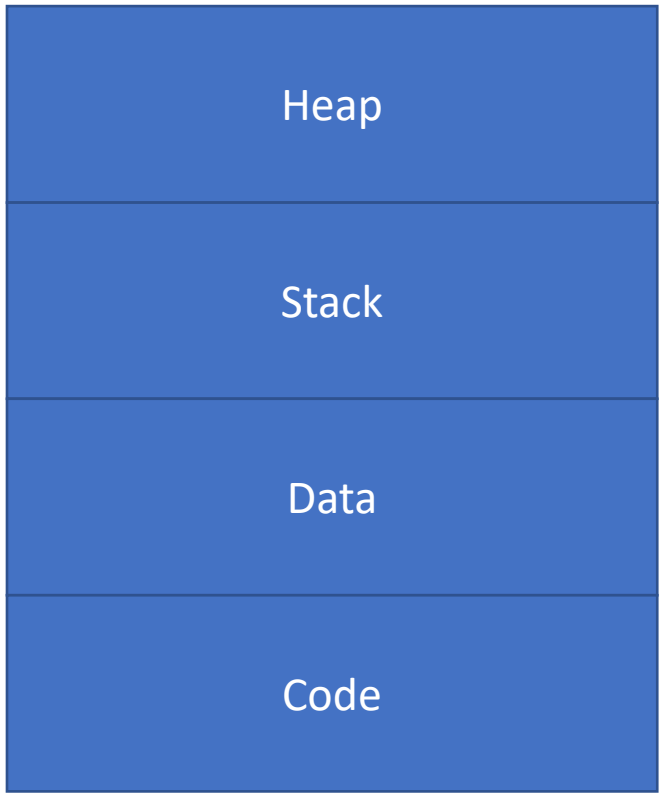
The word persistence comes from Latin

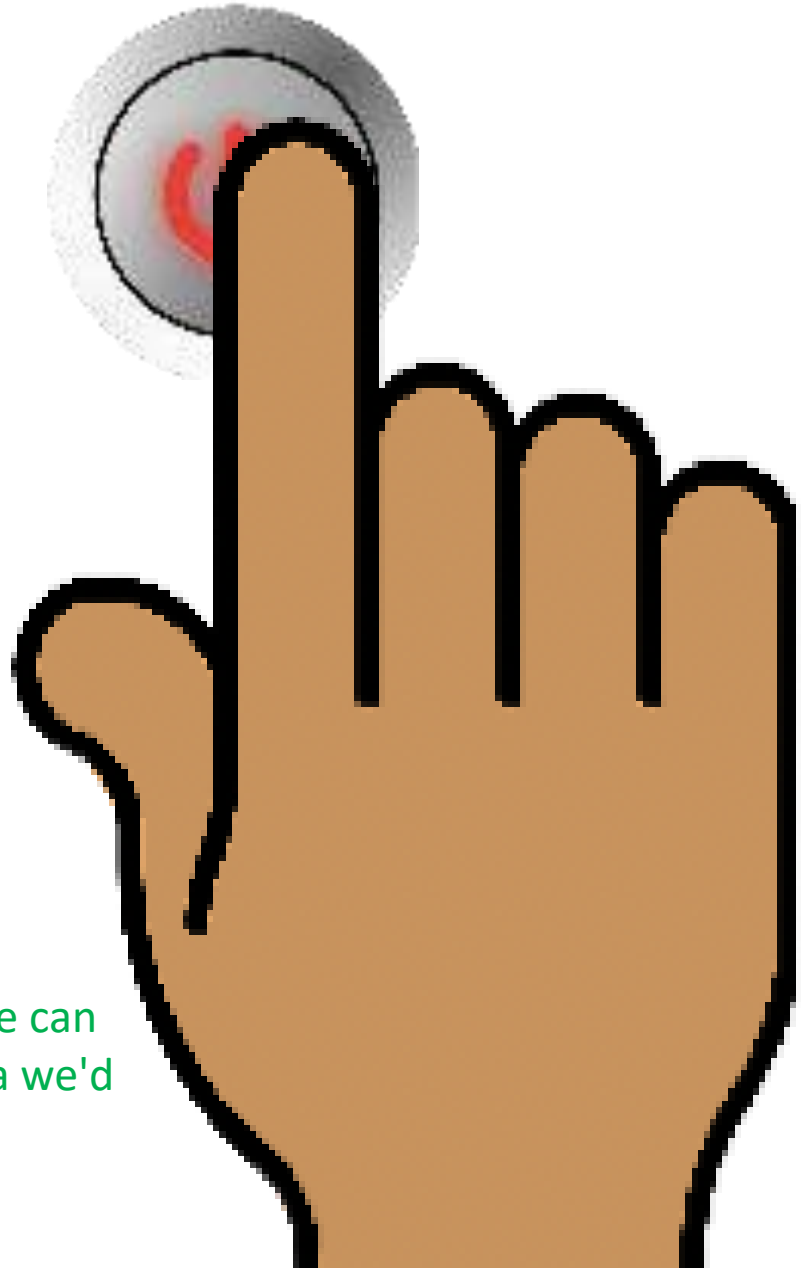
坚持 (jiān chí) insist, adhere, stick, persist, persistence, uphold
毅力 (yì lì) perseverance, willpower, persistence, stamina, fortitude, grit
持久 (chí jiǔ) lasting, persistent, durable, enduring, persistence, sustained



The von Neumann Architecture

In the von Neuman architecture we have everything in memory.



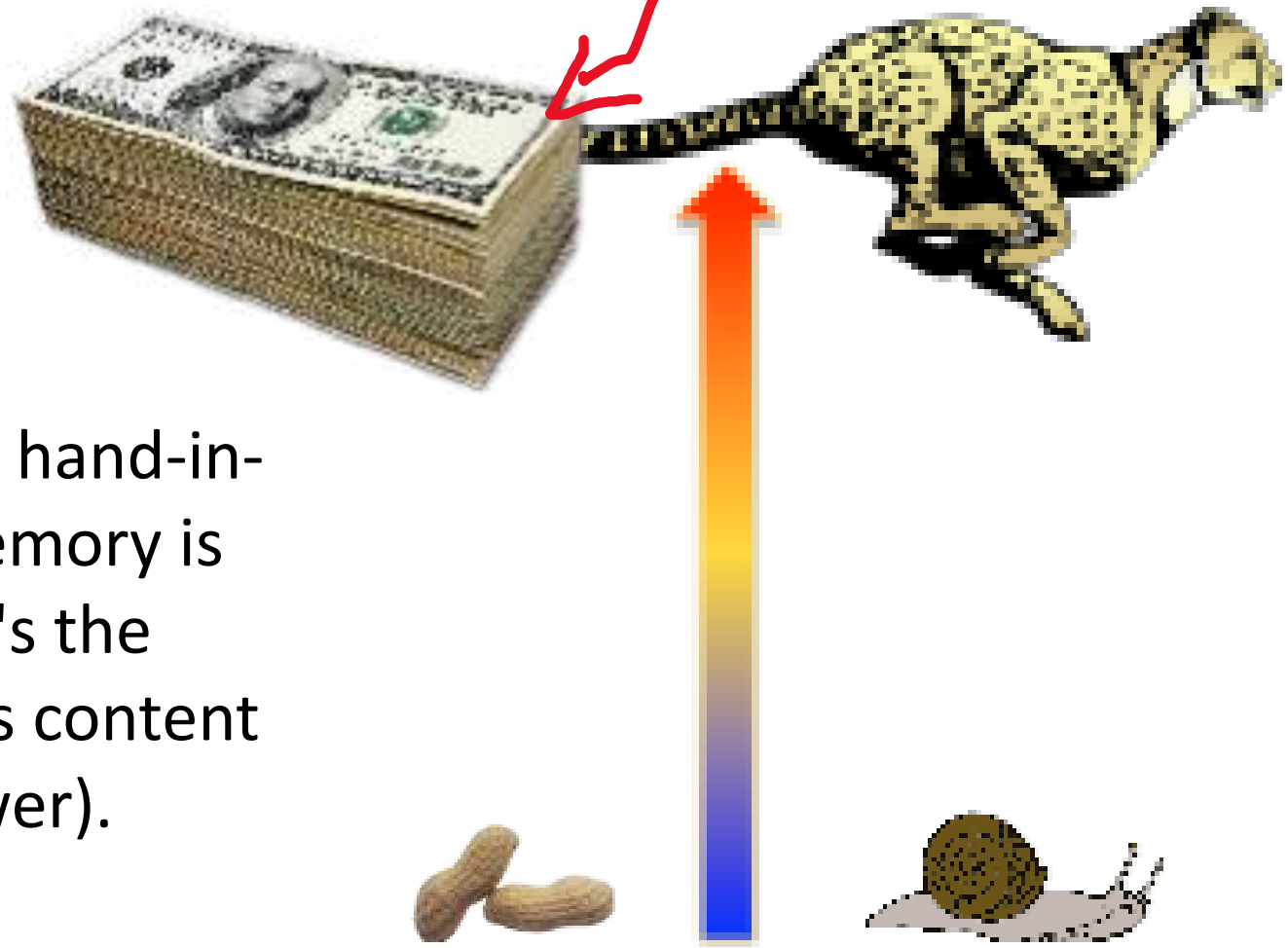


But what happens when the power is cut? Poof, gone. We can reload programs when we restart, but if they modify data we'd rather save the work done somewhere.

Need to persist program results somewhere

That's all the idea of persistence. "Somewhere" may take multiple shapes (including a remote computer)

Memory



Memory is like cars: fast goes hand-in-hand with expensive. Fast memory is "volatile", which means (that's the meaning of the word) that it's content can "fly away" (if you cut power). Slow memory stays, but the problem is that when you access it your program works at its slow speed.


MCL F1

So we have to work as much as we can in fast memory (RAM), and only in memory, for speed ...

Memory

Mostly work in memory for speed.

... while we still need a safety net.

But write to files for safety.

Non-Volatile Memory Technology

Magnetic

There are several non-volatile technologies available. The oldest one, still alive and kicking, is the magnetic disk, which has done tremendous progress over the years in data density and transfer rates (not so much directly accessing data somewhere on the disk, called "random access")

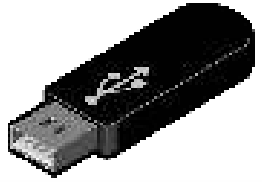
Non-Volatile Memory Technology

Optical

Optical technologies are no longer so hot as they used to be, but they are very good for archiving and, because readers/writers are mass produced and cheap. Data updates? Better to look elsewhere.

Non-Volatile Memory Technology

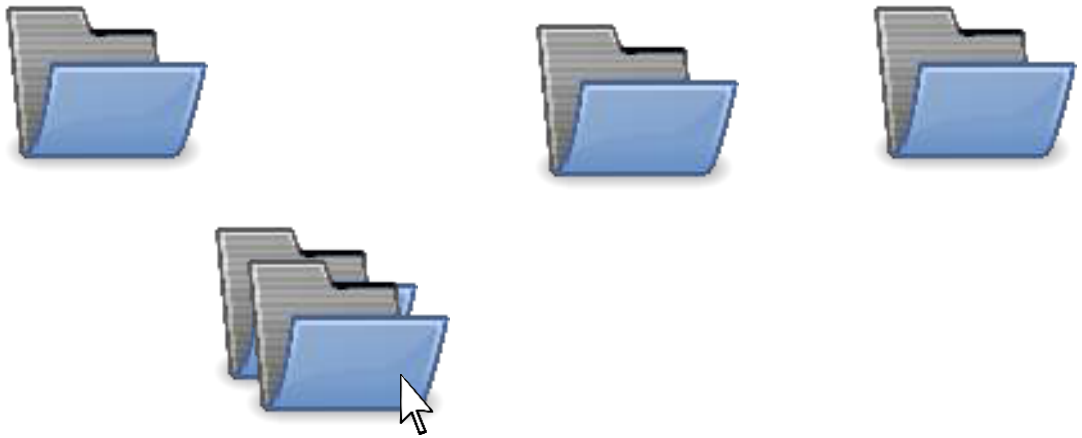
Solid State

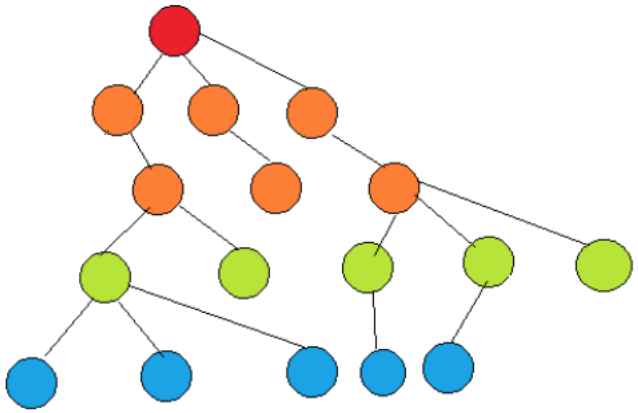


Solid-state technologies are very good for reading, much less so for writing.

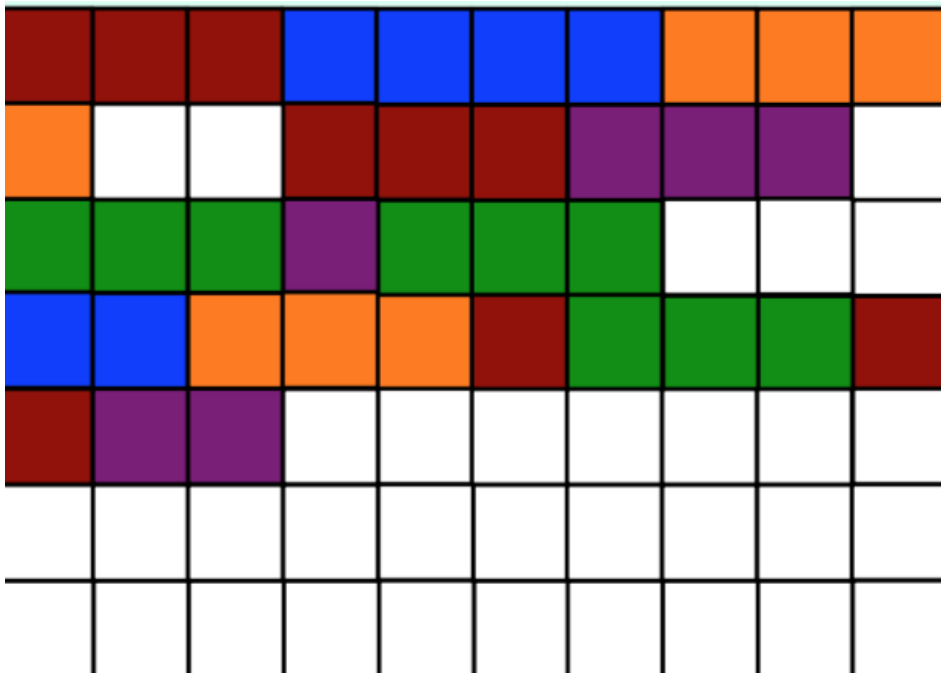
File System

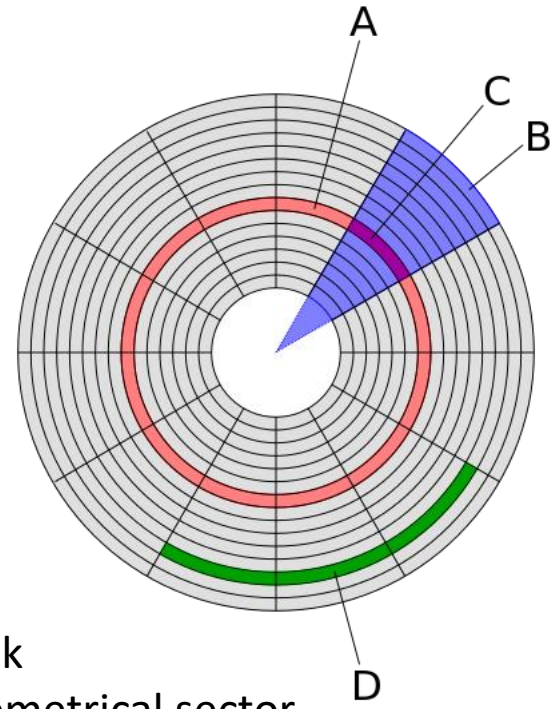
Directories/folders





Your operating system will create on a disk a "file system" giving you a hierarchical view of directories and files.





A – track
B – Geometrical sector
C – Track sector
D – Cluster

But in reality the system will reserve blocks, all multiples of a same unit, that it will associate with one file. Files will grow, deletions will create gaps soon filled by blocks from from other files, and the system will try to do all this as efficiently as possible. There is a strong disconnect between the view we have and physical reality.

Next: using file systems to store data

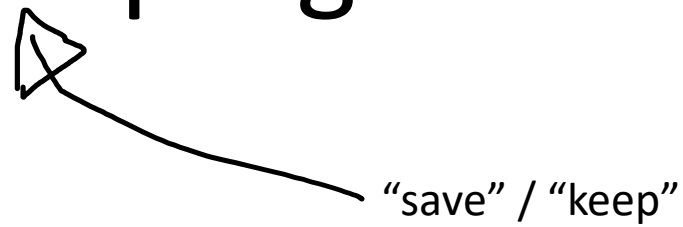
2 – Files and their Use

Lecture 8 Part 2

Memory

- Working in main memory is nice but everything goes when the computer is switched off (or crashes)

- Need to persist program results somewhere



- That is the idea of persistence: “somewhere” can be many different places (e.g. file, remote computer, etc, etc)

So we have to work as much as we can in memory, and only in memory, for speed ...

Memory

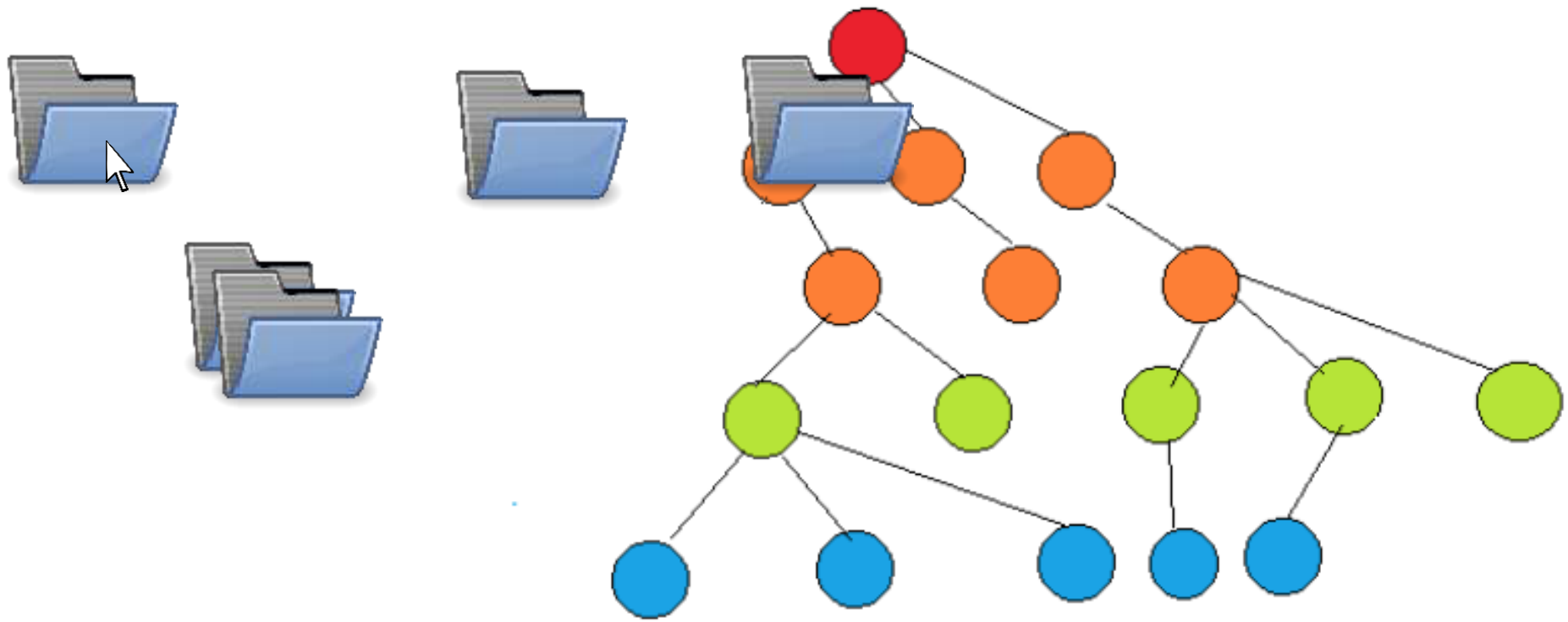
Mostly work in memory for speed.

... while we still need a safety net.

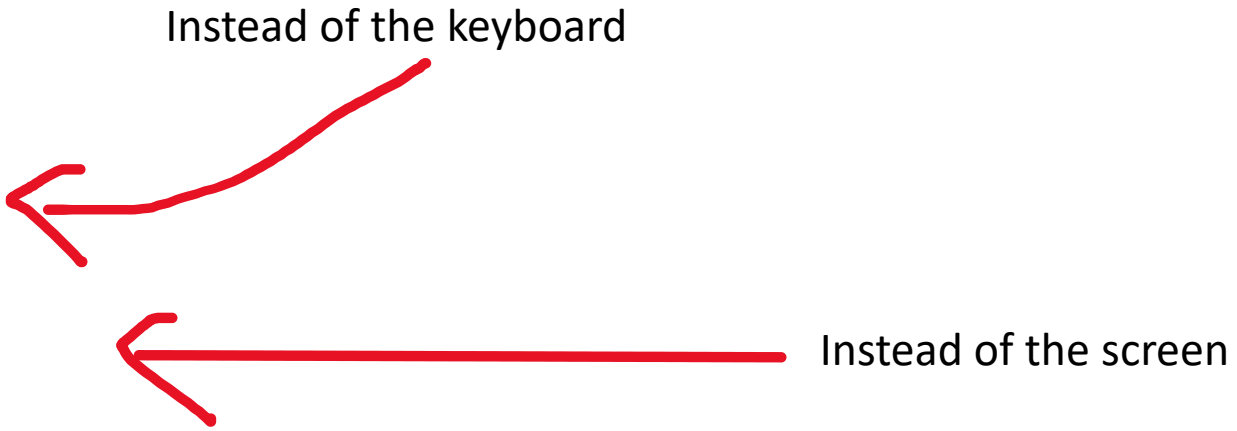
But write to files for safety.

File System

Directories/folders



Stream Redirection



One very easy way to save data and to restore it is, instead of doing it in Java, to let the system do it through what is known as "Stream redirection". Your program may read from what it thinks is the keyboard when in reality input comes from a file, and what is written to the screen can also be redirected to a file. System.out and System.err can be separated.

Types of Files

- Traditionally, there are mainly two types of files
  - Binary files

  - Text files

But if there are many different examples of file extensions and many types of files, **and** they all fall in one of two categories: text or binary - then the only problem is, whatever they are, they are all made of 0s and 1s. So really it's all a matter of interpretation.

Text files only contain printable characters

How to interpret the 0s and 1s

Interpretation is the big, hard question. You cannot guess the meaning of 1s and 0s just by looking at them. You must have an idea already. And even with text, there are many different ways to encode one single character (and don't believe that the problem doesn't exist even with basic Latin letters – there is another encoding system than ASCII called EBCDIC and the bits meaning 'a' in ASCII mean '/' in EBCDIC). If you haven't the key allowing you to decrypt the bits, you are lost.

Types of Files

- Traditionally, there are mainly two types of files
- Binary files

- Text files

So the true definition of a "text file" is that it only contains characters that you can print (including spaces and carriage returns) when you decrypt the file as a bunch of characters.

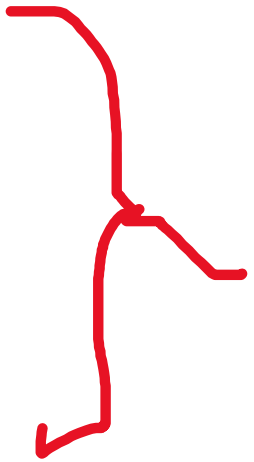
only printable characters WHEN DECODING AS CHARACTERS

Text Files

There are many types of text files: not only files with the extension .txt!

They can be opened with a ”text editor”
e.g. Notepad in windows, more in linux

Code - .c, .h, .py, .php, .java, .bat, .sh, …
Plain text - .txt, .ini
Text with readable tags - .html, .rtf, .xml
Data as text - .csv



All these only contain printable characters

Binary Files

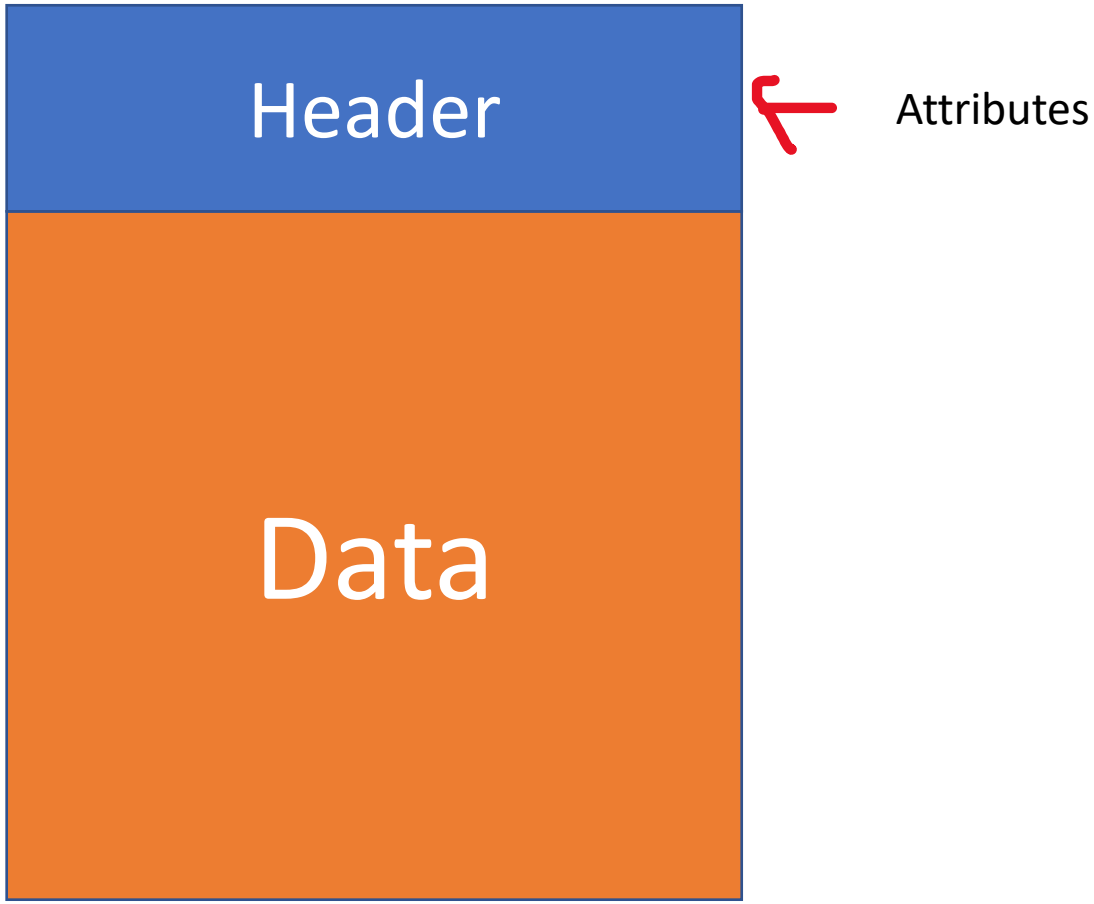
There are also many types of binary files… Including used by text applications…

They can only be opened with a **special** program!

Compiled program - .o, .exe. class
Archives, compressed files - .tar, .tgz, .zip
Encrypted files
Text with non readable formatting - .docx, .pdf
Multimedia - .gif, .mp4, .png, .flv, …

Binary Files

Most often a binary file contains a header part that describes the structure, followed by data proper.



Binary Files

Memory structures written ”as is”
More compact (no encodings etc)
No conversion during I/O
May be portability problems between computers (windows - mac etc)

One issue is the “small endian”/ ”big endian” issue which is a hardware spec.
The 4 bits that make up ½ a byte might be swapped

Stream Redirection

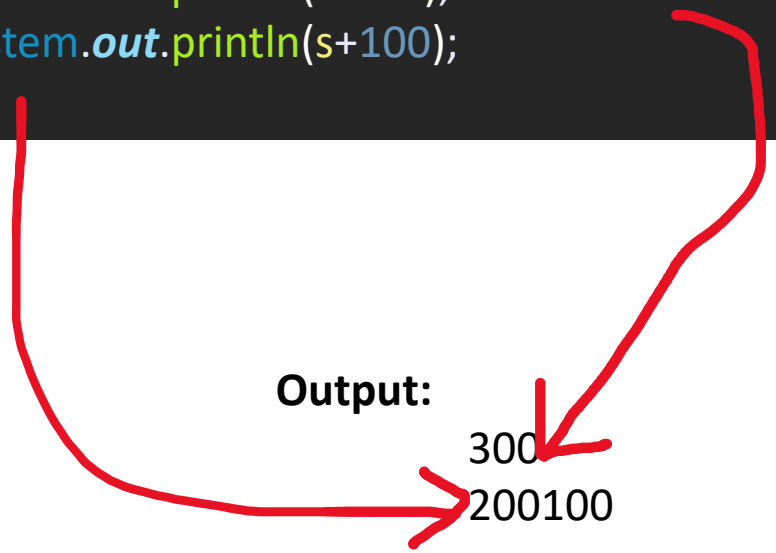
$ java MyProgram < input_file

$ java MyProgram > output_file

Although nothing forbids writing to and from binary files stream redirection usually uses text files

ToString()

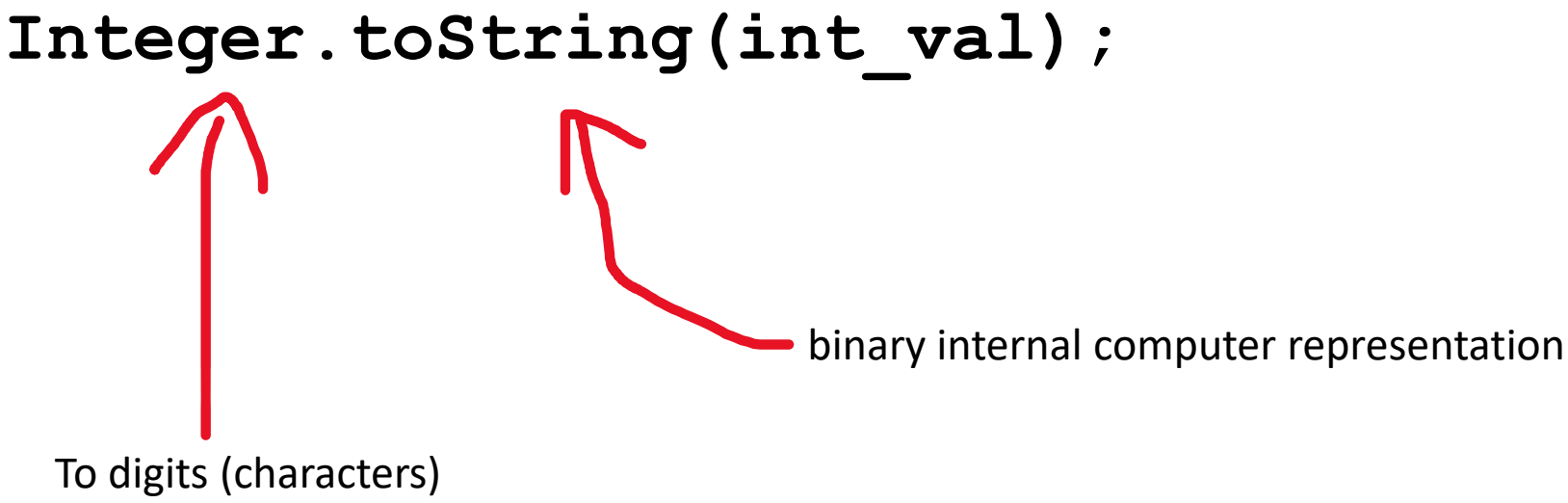
```
public static void main(String args[]){
        int i=200;
        String s=String.valueOf(i);
        System.out.println(i+100);
        System.out.println(s+100);
}
```



Output:

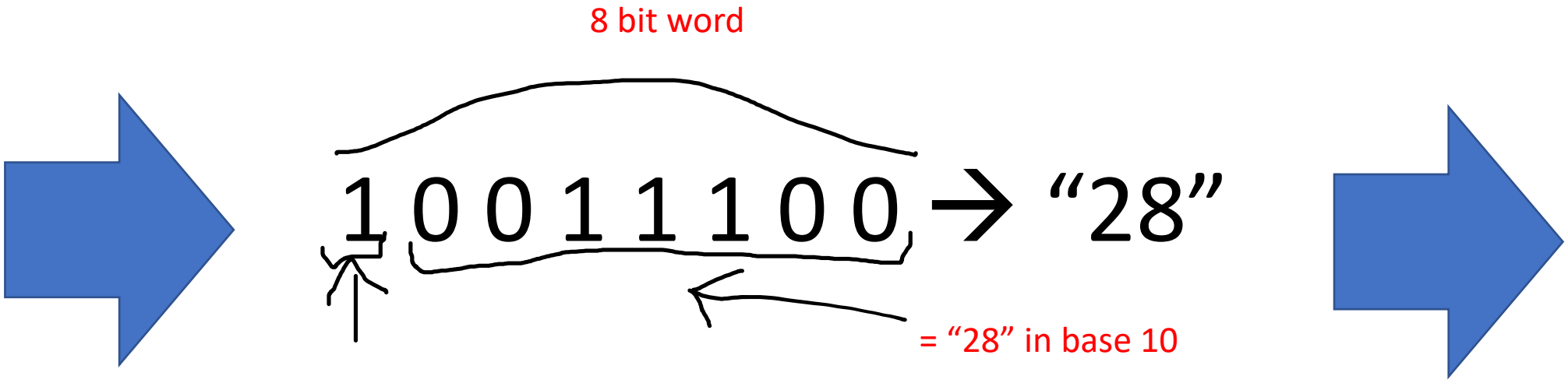
300
200100

Key Point: whether you are calling println with an integer, whether you are explicitly calling toString() or not, a conversion occurs.



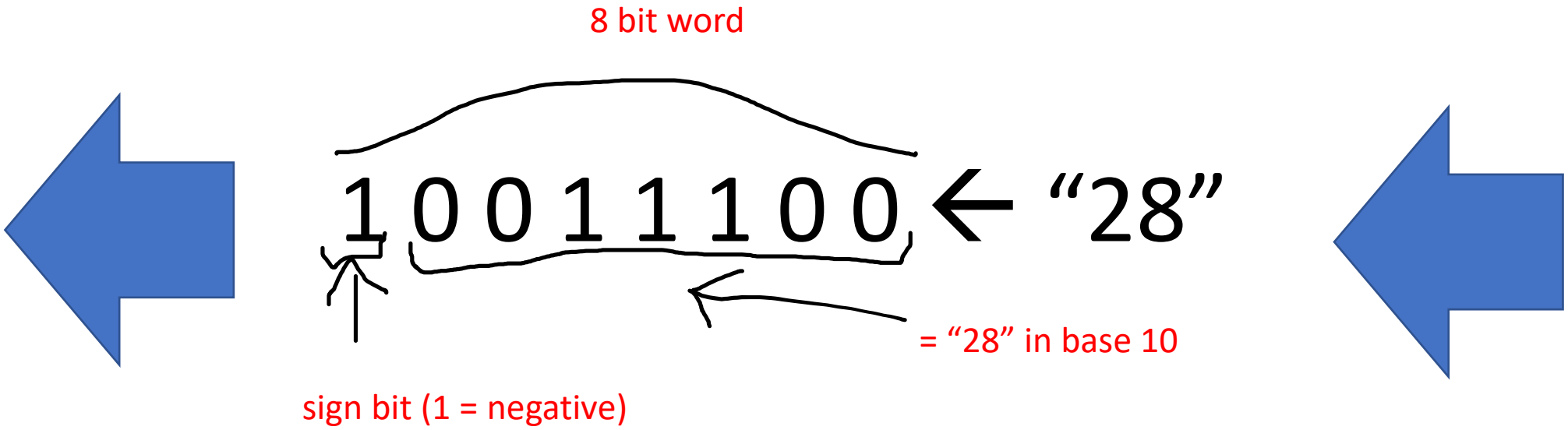
ToString()

A number has to be turned into a string of digits for output.



if number is negative display '-'
loop on decreasing powers of 10
    get the result r of the integer division of the number by the power of 10
    if we have already displayed a non zero digit
        display the digit corresponding to the code of '0' plus r
    else the digit is not zero
        record that we have found a non zero digit
        display the digit corresponding to the code of '0' plus r decrease the number by r times the power of 10
        processed
end loop

Input requires the opposite



Integer.parseInt() or the method nextInt() of a Scanner object perform the reverse operation

How to interpret the 0s and 1s

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello!");
    }
}
```

A program has, to "understand a file", a number of options.

How to interpret the 0s and 1s

A program has, to "understand a file", a number of options.

1. Assume that it is what we expect…

   For instance text.

This is the simplest option: ”when the only tool is a hammer everything looks like a nail!”

```
As-MacBook-Pro:code ag$ javac Hello.java
As-MacBook-Pro:code ag$ ls
Hello.class     Hello.java
As-MacBook-Pro:code ag$ cat Hello.class
????:

java/lang/Object<init>()V




java/lang/SystemoutLjava/io/PrintStreamHello!

java/io/PrintStreamprintln(Ljava/lang/String;)VHelloCodeLineNumberTablemain([Lja
va/lang/String;)V
SourceFile
??llo.java!*??  %       ?
As-MacBook-Pro:code ag$
```

cat writes everything to the screen – here the result looks like garbage!

How to interpret the 0s and 1s

A program has, to "understand a file", a number of options.

2. Assume that the extension is correct

Hello.class is a java bytecode file

I renamed Hello.class to Hello.c and tried to compile it. I got 105 warnings and 11 errors but the compiler tried

How to interpret the 0s and 1s

A program has, to "understand a file", a number of options.

3. Check the header for a “magic number”

Binary files usually contain a "signature" in their header, a small number of bytes that are very specific to one type of files. You don't need to trust the extension.

- hexdump just dumps the file contents
- All class files start with the same bytes.

cafe babe

```
$ hexdump -n 1024 Hello.class
0000000 ca fe ba be 00 00 00 3a 00 1d 0a 00 02 00 03 07
0000010 00 04 0c 00 05 00 06 01 00 10 6a 61 76 61 2f 6c
0000020 61 6e 67 2f 4f 62 6a 65 63 74 01 00 06 3c 69 6e
0000030 69 74 3e 01 00 03 28 29 56 09 00 08 00 09 07 00
0000040 0a 0c 00 0b 00 0c 01 00 10 6a 61 76 61 2f 6c 61
0000050 6e 67 2f 53 79 73 74 65 6d 01 00 03 6f 75 74 01
0000060 00 15 4c 6a 61 76 61 2f 69 6f 2f 50 72 69 6e 74
0000070 53 74 72 65 61 6d 3b 08 00 0e 01 00 06 48 65 6c
0000080 6c 6f 21 0a 00 10 00 11 07 00 12 0c 00 13 00 14
0000090 01 00 13 6a 61 76 61 2f 69 6f 2f 50 72 69 6e 74
00000a0 53 74 72 65 61 6d 01 00 07 70 72 69 6e 74 6c 6e
00000b0 01 00 15 28 4c 6a 61 76 61 2f 6c 61 6e 67 2f 53
00000c0 74 72 69 6e 67 3b 29 56 07 00 16 01 00 05 48 65
00000d0 6c 6c 6f 01 00 04 43 6f 64 65 01 00 0f 4c 69 6e
00000e0 65 4e 75 6d 62 65 72 54 61 62 6c 65 01 00 04 6d
00000f0 61 69 6e 01 00 16 28 5b 4c 6a 61 76 61 2f 6c 61
0000100 6e 67 2f 53 74 72 69 6e 67 3b 29 56 01 00 0a 53
0000110 6f 75 72 63 65 46 69 6c 65 01 00 0a 48 65 6c 6c
0000120 6f 2e 6a 61 76 61 00 21 00 15 00 02 00 00 00 00
0000130 00 02 00 01 00 05 00 06 00 01 00 17 00 00 00 1d
0000140 00 01 00 01 00 00 00 05 2a b7 00 01 b1 00 00 00
0000150 01 00 18 00 00 00 06 00 01 00 00 00 02 00 09 00
0000160 19 00 1a 00 01 00 17 00 00 00 25 00 02 00 01 00
0000170 00 00 09 b2 00 07 12 0d b6 00 0f b1 00 00 00 01
0000180 00 18 00 00 00 0a 00 02 00 00 00 04 00 08 00 05
0000190 00 01 00 1b 00 00 00 02 00 1c
```

Next: Streams

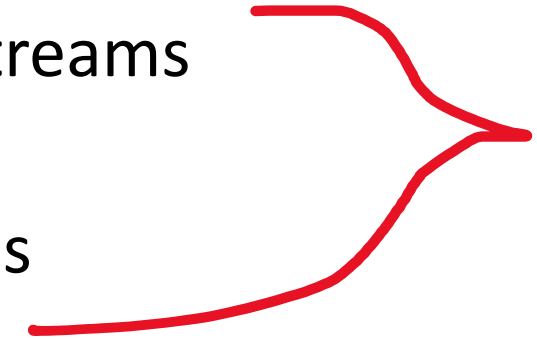
Files and Streams

Lecture 8 Part 3

Streams

- Most file-related classes are in the java.io package (there is also a java.nio which stands for “Network IO”). Two types of streams, Character or Byte, which can also be buffered or unbuffered.

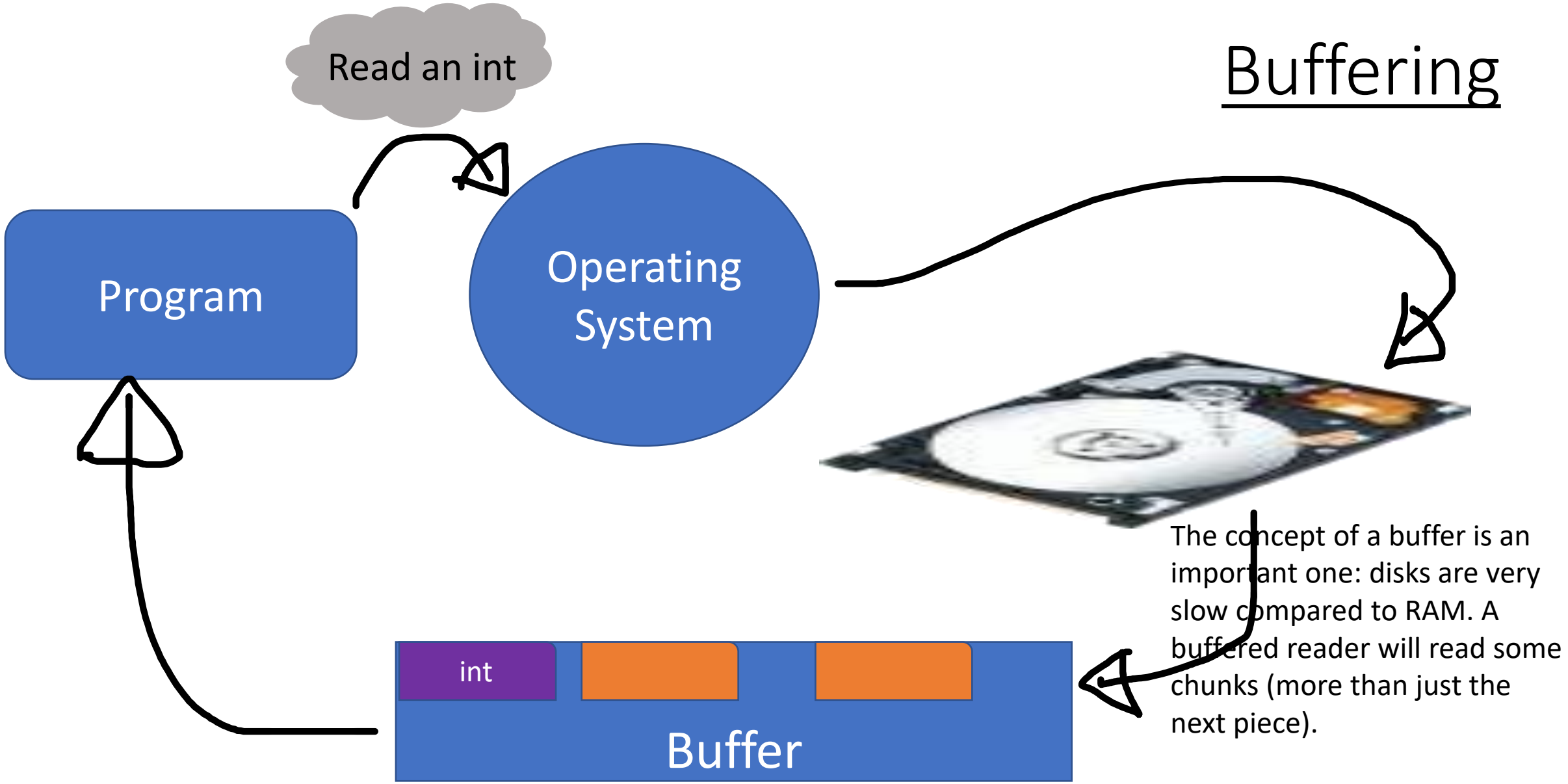
- Character streams
- Byte streams



Buffered or unbuffered

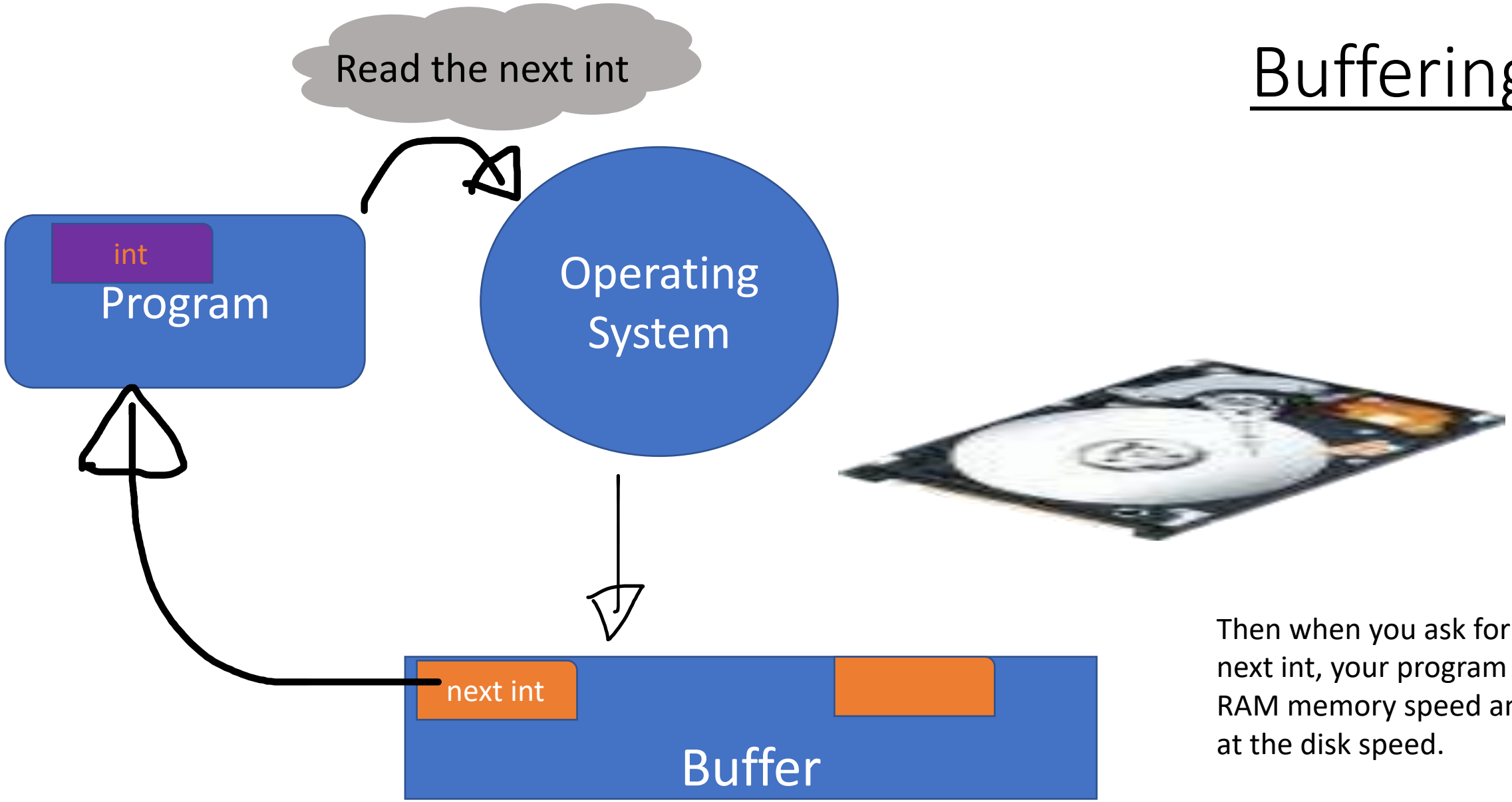
```
import java.io.*
```

Buffering



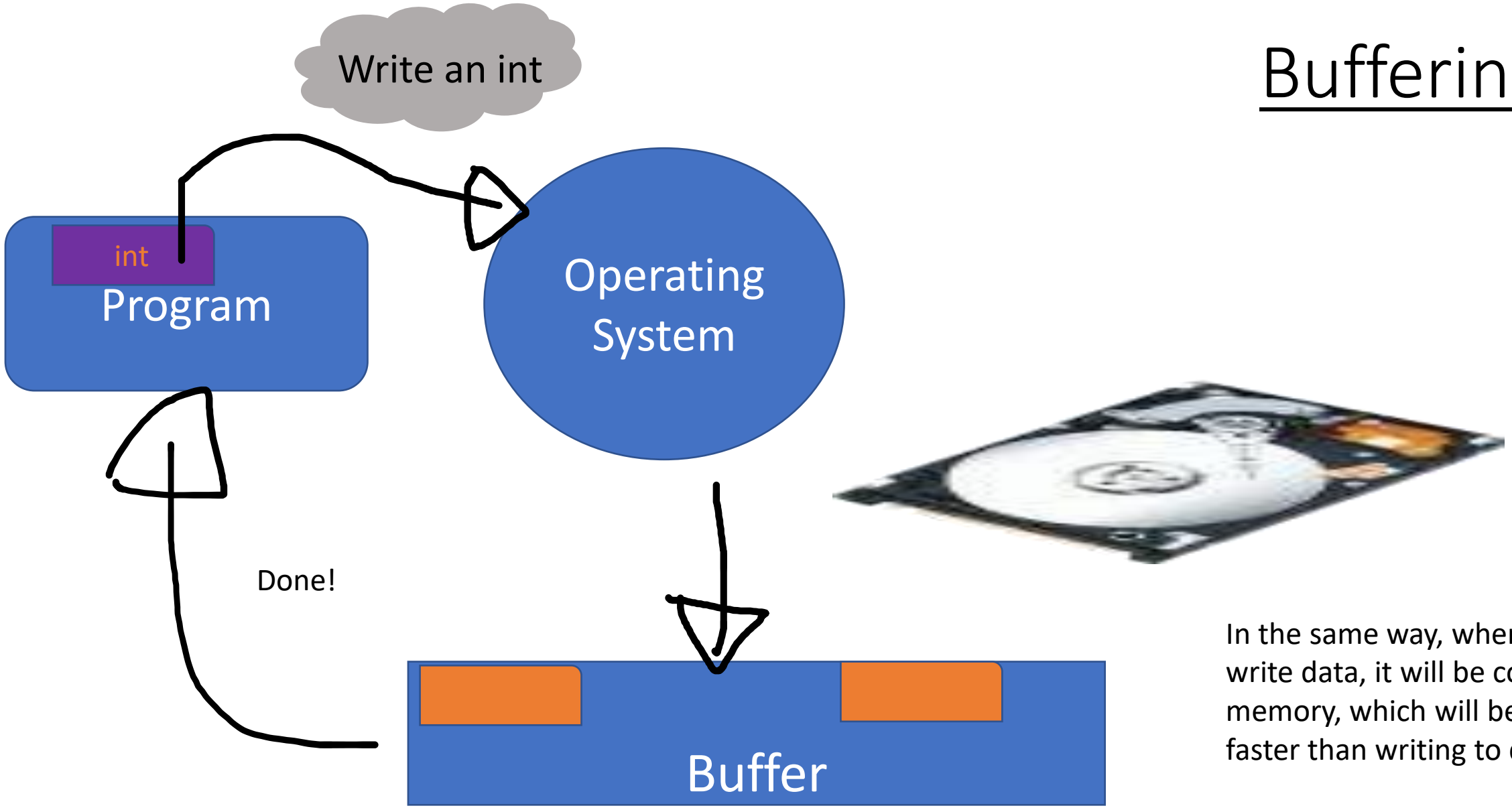
The concept of a buffer is an important one: disks are very slow compared to RAM. A buffered reader will read some chunks (more than just the next piece).

Buffering



Then when you ask for the next int, your program runs at RAM memory speed and not at the disk speed.

Buffering



In the same way, when you write data, it will be copied to memory, which will be much faster than writing to disk.

Buffering

Data will be bulk transferred to the disk when the buffer is full or you close the file (or you call a method to explicitly flush the buffers).

Buffering

Data will be bulk transferred to the disk when the buffer is full or you close the file (or you call a method to explicitly flush the buffers).

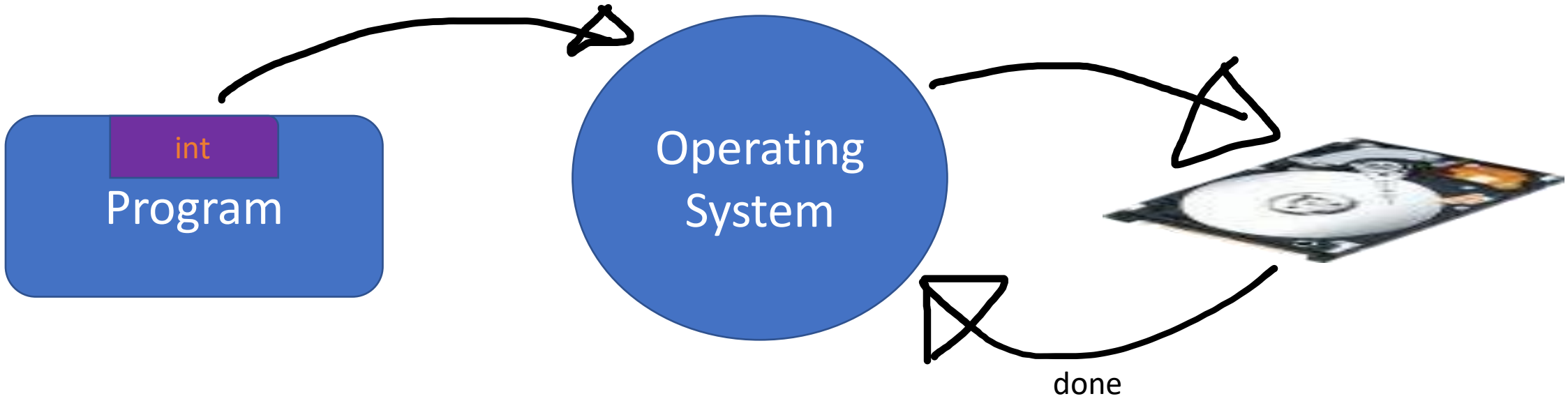
Buffering

- What happens if the system restarts after a crash?

The answer of course is that what was in buffers is lost. Not a problem when reading, big problem when writing. It's not always easy to know what to replay.

No Buffering (I/O direct to disk)

Performance: Should used buffered operations 99.9% of the time. Use unbuffered operations just when writing safely is a critical concern (or for debugging).



Instead can use slow unbuffered operations:

- Used for writing critical information
- Logs
- Messaging

Performance - copying a 11M CSV file char by char

- Test on Mac (internal SSD)*

Unbuffered	approx. 34.5s
Buffered	approx. 1s

- Test on Mac (External USB key)*

Unbuffered	approx. 36s
Buffered	approx. 1.4s

- Consider if its worth it: how often does your computer crash? How bad would it be to rerun the program after restart?

*Tests run by Stephane Faroult in previous years

Sometimes hard to know whats going on…

- Big disk systems have their own battery protected buffers (also called a cache)

Especially with high-end storage you rarely have one level of buffering (in which case unbuffered wouldn't be what it seems). It's a bit hard sometimes to know if the data is on disk or not, and the Cloud doesn't make it any simpler.

Unbuffered

```
InputStream in = null;
OutputStream out = null;


in = new InputStream(...);
out = new OutputStream(...);
```

The basis for all binary Input/Output operations are InputStreams and OutputStreams, which are unbuffered.

One thing that should not be forgotten with file operations if that it's probably the part of a program where everything can fail.

Lots of things can go wrong...

Wrong file directory

Not allowed permissions

Content not as expected

Hardware problem (rare)

Unbuffered

So everything should really be done with exception handling: either with a ”try with resources” or a “finally” block to make sure files are cleanly closed and not left in a “corrupted” state.

```
    InputStream in = null;
    OutputStream out = null;

    try {
        in = new InputStream(...);
        out = new OutputStream(...);

        ...
        } catch (...)  {

        } finally {
            if (in != null)  {
                try {
                    in.close();
                } catch (IOException e)
                    // ignore
                }

    }
}
```

Important: flush everything and close

Unbuffered

```
    FileInputStream in = null;
    FileOutputStream out = null;

    try {
        in = new FileInputStream("filename");
        out = new FileOutputStream("filename");


        } catch (...)  {

        } finally {
            ...
    }
}
```

Looks in the current directory unless you provide a full path

File location is always a practical problem. Use reflection and properties files.

Buffered

To turn an unbuffered stream into a buffered one, just wrap the call to the stream constructor in a call to a buffered stream constructor.

```
    BufferedInputStream in = null;
    BufferedOutputStream out = null;

    try {
        in = new BufferedInputStream(new InputStream(...);
        out = new BufferedOutputStream(new OutputStream(...));


        } catch (...)  {

        } finally {
            ...
    }
}
```

Next: Byte and Character Streams + Object Serialization

Object Serialization

The way of using byte streams to save objects

Lecture 8 Part 4

InputStream and OutputStream are the parent classes for all byte streams.

Byte Streams

Byte streams are not the most used and there are libraries specifically for multimedia…

Remember JavaFx: when you create a new Image or Media object, a binary file is read into memory by the constructor. There is necessarily a byte stream behind the scene, but it's all done by the constructor.

See also the docs: https://docs.oracle.com/javase/tutorial/essential/io/bytestreams.html

ByteStream (unbuffered)

```
    FileInputStream in = null;
    FileOutputStream out = null;

    try {
        in = new FileinputStream("filename");
        out = new FileOutputStream("filename");


        } catch (...)  {

        } finally {
            ...
    }
}
```

The examples in the previous presentation were for byte streams.

ByteStream (buffered)

```
    BufferedInputStream in = null;
    BufferedOutputStream out = null;

    try {
        in = new BufferedinputStream(new FileInputStream("filename"));
        out = new BufferedOutputStream( ... );


        } catch (...)  {

        } finally {
            ...
   }
}
```

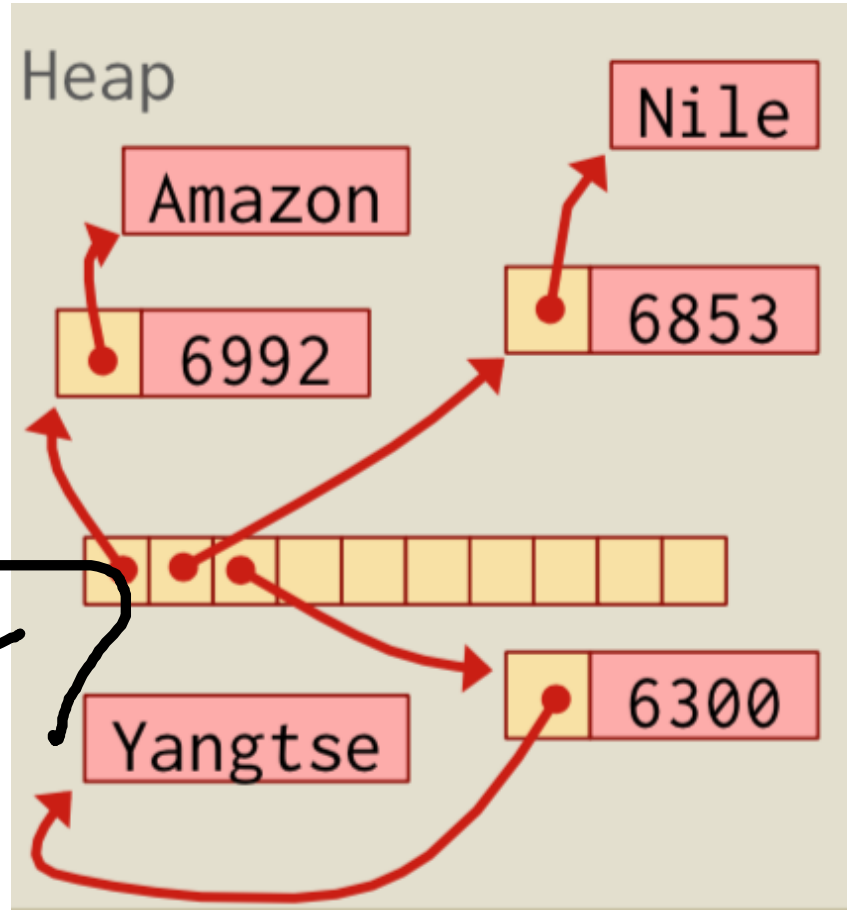
… including the buffered version.

Terminology: when you are referring to "bytes" in I/O operations you are really dealing with int variables. Not byte variables. For historical reasons they are called byte streams.

Don't be misled by the "byte" in "byte stream". Operations deal with more than one byte.

Object Serialization

```
class Obj2 {
    private String name;
    private int value;
    ...

}

class Obj1 {
    private Obj2[] o = null;
    private int count = 0;

    public Obj1() {
        o = new Obj2[10];
    }

    ...

}
```



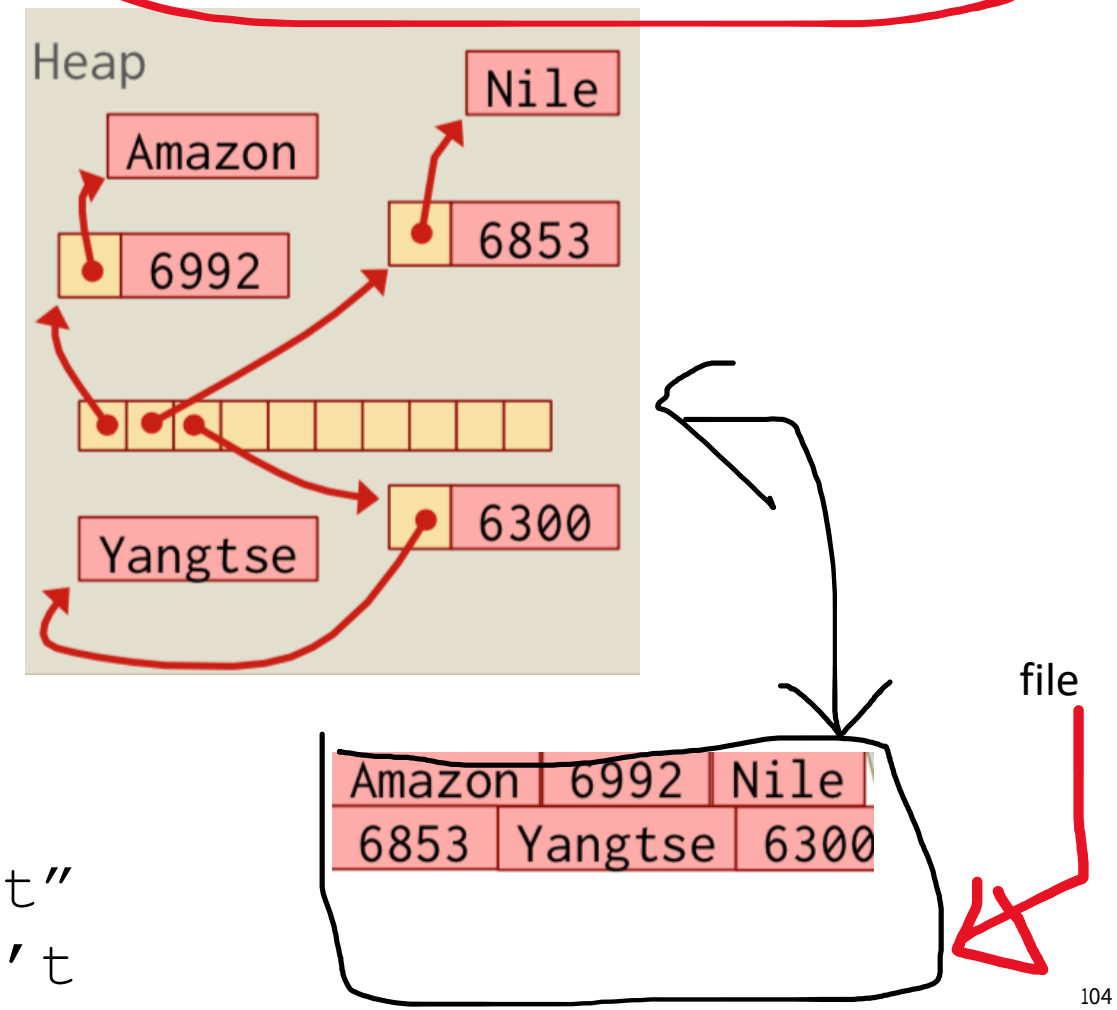
One application of byte streams is "serialization", dumping a memory object to file...

Object Serialization

Important detail: When doing this you want to store the data – **not** the memory addresses that change when you reload

```
class Obj2 {
    private String name;
    private int value;
    ...

}

class Obj1 {
    private Obj2[] o = null;
    private int count = 0;

    public Obj1() {
        o = new Obj2[10];
    }

    ...
}
```

You can also declare attributes "transient" meaning they shouldn't be dumped.



Object Serialization: requisites

(What you need to do to use it)

1. Interface

```
class Obj2 implements java.io.serializable {
        private String name;
        private int value;
…
}

class Obj1 implements java.io.serializable{
        private Obj2[] o = null;
        private int count = 0;

        public Obj1() {
                o = new Obj2[10];
        }}
```

No method!
Its just a declaration and there is no method to implement. It is just to tell javac to generate necessary requirements to save the data to disk.

Object Serialization: requisites

```
class Obj2 implements java.io.serializable {
        private String name;
        private int value;
…
}

class Obj1 implements java.io.serializable{
        private Obj2[] o = null;
        private int count = 0;

        public Obj1() {
                o = new Obj2[10];
        }}
```

2. Constructor

Object Serialization: requisites

And you need an "ObjectOutputStream" that is a special flavor of byte stream. This one comes by default with a buffer.

```
FileOutputStream fileOut = new FileOutputStream("file.dat");
ObjectOutputStream out = new ObjectOutputStream(fileOut);
out.writeObject(o);
out.close();
fileOut.close();
```

3. Stream

has a buffer included

A file written on one computer can be read on any computer

Next: using character streams and databases instead of byte streams to save program state

Character Streams

Lecture 8 Part 5

Character Stream

- Character streams are used more frequently than byte streams – probably in all the cases you have seen so far when you read or write a file you have been using a character stream

- Handle 16-bit unicode characters (char datatype)

- If you have a C background, don't forget that Java chars are 2 bytes, not one as in C.

Character Strea
(unbuffered)

```
FileReader in = null;
FileWriter out = null;

try {
    in = new FileReader("filename");
    out = new FileWriter("filename");

} catch ... {
} finally {
    if (in != null) {
        in.close();
    }
    if (out != null) {
        out.close();
    }
}
```

What was called "Input" and "Output" with byte streams becomes "Reader" and "Writer" with character streams…

Character Stream (buffered)

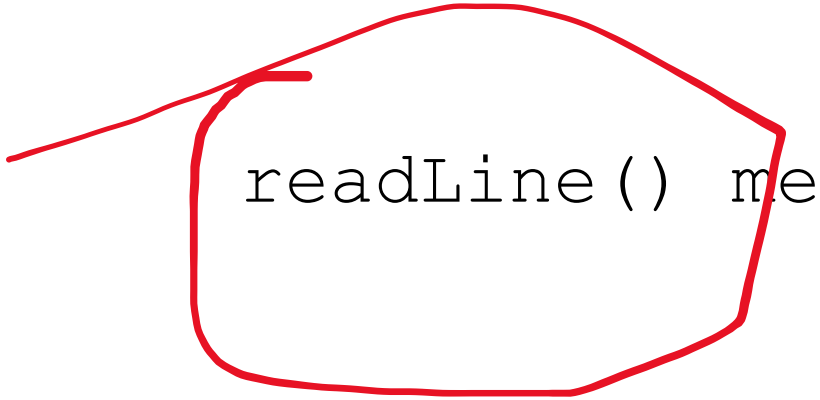
```
BufferedReader in = null;
BufferedWriter out = null;

try {
    in = new BufferedReader(new FileReader(...));
    out = new BufferedWriter(new FileReader(...));

} catch ... {
} finally {
    if (in != null) {
        in.close();
    }
    if (out != null) {
        out.close();
    }
}
```

Otherwise it's exactly the same thing.


So what…
can character streams do that byte streams can’t?

1. They can read (write) lines

Need to use a BufferedReader



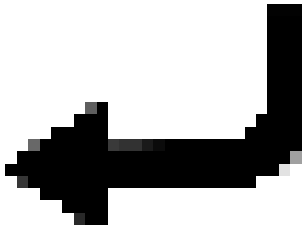
When buffered they can read or write a full line at once – because text files are usually a sequence of lines.

```
BufferedReader in = null; String line;
try {
    in = new BufferedReader(new FileReader("..."));
    while ((line = in.readLine()) != null) {
    ... }
} finally {
    if (in != null) {
        try {
            in.close();
        } catch (Ioexception e) {
            ...
        }
    }
}
```

But beware of lines when text files were written on a system and are read on a different system.

%n in Java format

Which brings the interesting problem of lines. You probably know the carriage return character, '\n'. Java prefers '%n' with printf(), which may be one or two characters depending on the system it runs on.



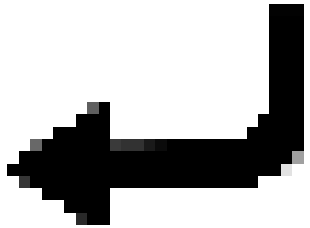
It all dates back to the glorious days of the typewriter, in which letters were always typed at the same place. It's the "carriage" around which the sheet was wrapped that moved from right to left.

> *Where is the Life we have lost in living?*
> *Where is the wisdom we have lost in knowledge? Where is the knowledge we have lost in information?*
>
> TS Eliot
> (1888-1965)

What was happening at the end of the line? You needed to scroll the paper (line feed) and push back the carriage to the right (carriage return).

Guess what, the first printers were computer-controlled typewriters (sort of).

UNIX



Carriage return

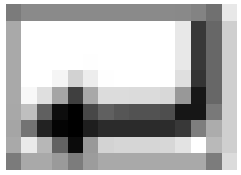
Where everything becomes fun, it's that a Unix system separates lines with a single 'carriage return' character.

Where is the Life we have lost in living? *Where is the wisdom we have lost in knowledge?* *Where is the knowledge we have lost in information?*

\n
hex: 0x0A
binary: 00001010

Windows

Carriage return + Line feed



While a Windows system still stores the two separate commands that used to be sent to the printer - line feed (represented by \r), then carriage return.

Where is the Life we have lost in living? *Where is the wisdom we have lost in knowledge?* *Where is the knowledge we have lost in information?*

\r\n
0x0D0A
0000110100001010

common ascii codes to know

Char	Dec	Oct	Hex	WhatAreThey
(nul)	0	0000	0x00	Null
(ht)	9	0011	0x09	Horizontal Tab
(nl)	10	0012	0x0a	New Line
(vt)	11	0013	0x0b	Vertical Tab
(cr)	13	0015	0x0d	Carriage Return
(sp)	32	0040	0x20	Space
0	48	0060	0x30	zero
A	65	0101	0x41	capital A
a	97	0141	0x61	lowercase a

Linux file in a Windows editor

Where is the Life we have lost in living?□ Where is the wisdom we have lost in knowledge?□ Where is the knowledge we have lost in information?□□ TS Eliot□ (1888-1965)

A basic Windows editor (as Linux became more common, many editors became cleverer), looking for \r\n as separator between lines, will see in a Linux text file only one big line with \n characters that it doesn't know how to represent and that it will replace with squares.

Windows file in a Linux editor

Where is the Life we have lost in living?^W
Where is the wisdom we have lost in knowledge?^W
Where is the knowledge we have lost in information? ^W
^W
TS Eliot^W
(1888-1965) ^W

Going the other way, a Linux the editor would understand the \n correctly from the windows file but it will not know what to do with the \r – it will display these as ^W.

There is nothing simple in IT… :-)

There are conversion programs.

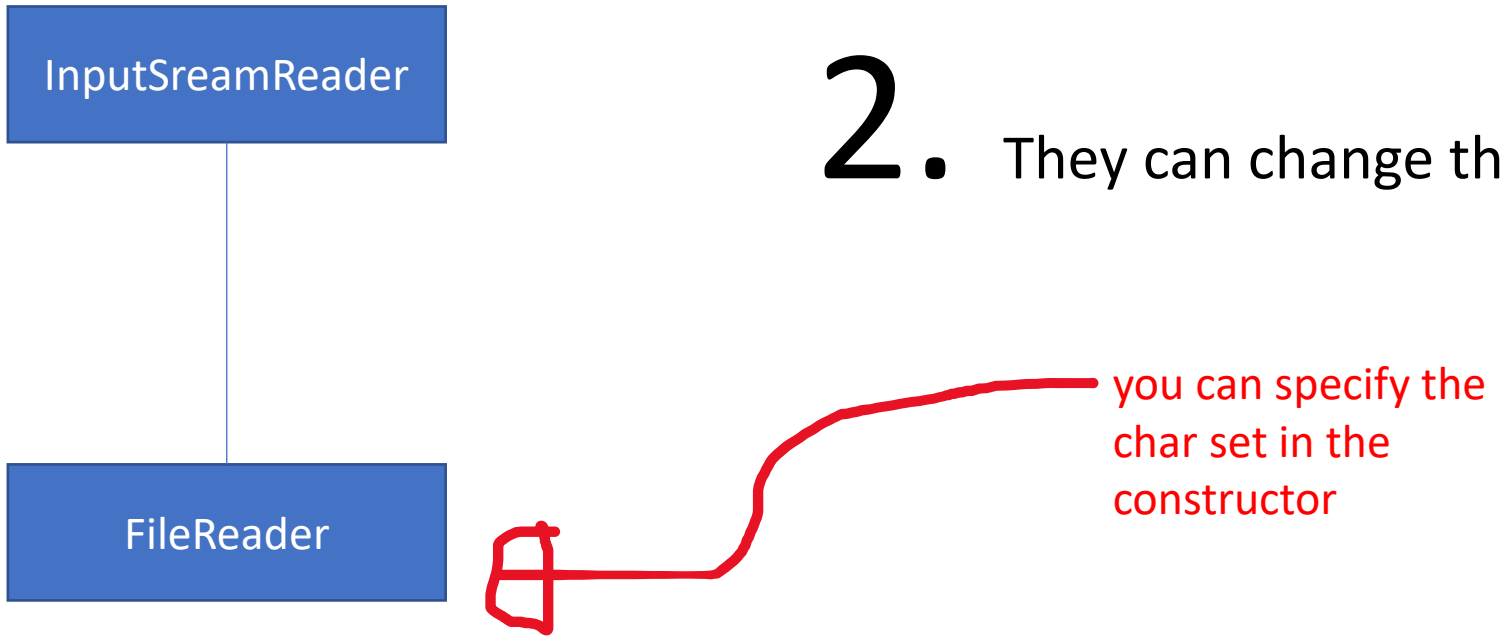
Many programs understand both.

MOST character files are organized in lines (variable or constant length)

BUT a big character file can sometimes contain a single line (HTML, JSON, XML ...)

U would of course be wrong to believe that a big text file always contains many lines.

Another interesting characteristic of character streams is that the parent class of FileReader allows you to specify the char encoding.



2. They can change the encoding

You can also use a Scanner object with a character stream, which is very good for parsing text input.

3. You can use a Scanner

Very similar to screen and keyboard which are character devices

Related: other alternatives for persistence (apart from serialization)

XML and Databases

Persistence and Databases

Lecture 9 Part 1

Benefits of Character Streams

- What can character streams do that byte streams cannot?
- Read and write line by line
  - When buffered
- Change encodings
- Can use a scanner and parse text input

File and Directory Operations

- Check the File class…
  - https://docs.oracle.com/javase/8/docs/api/java/io/File.html

- Copying and deleting files
- Listing and searching directories
- and more

You can do a lot of things with files other than reading and writing them. There are constants in the File class that take care of differences between Linux, Mac, and Windows.

Files USED to be very important

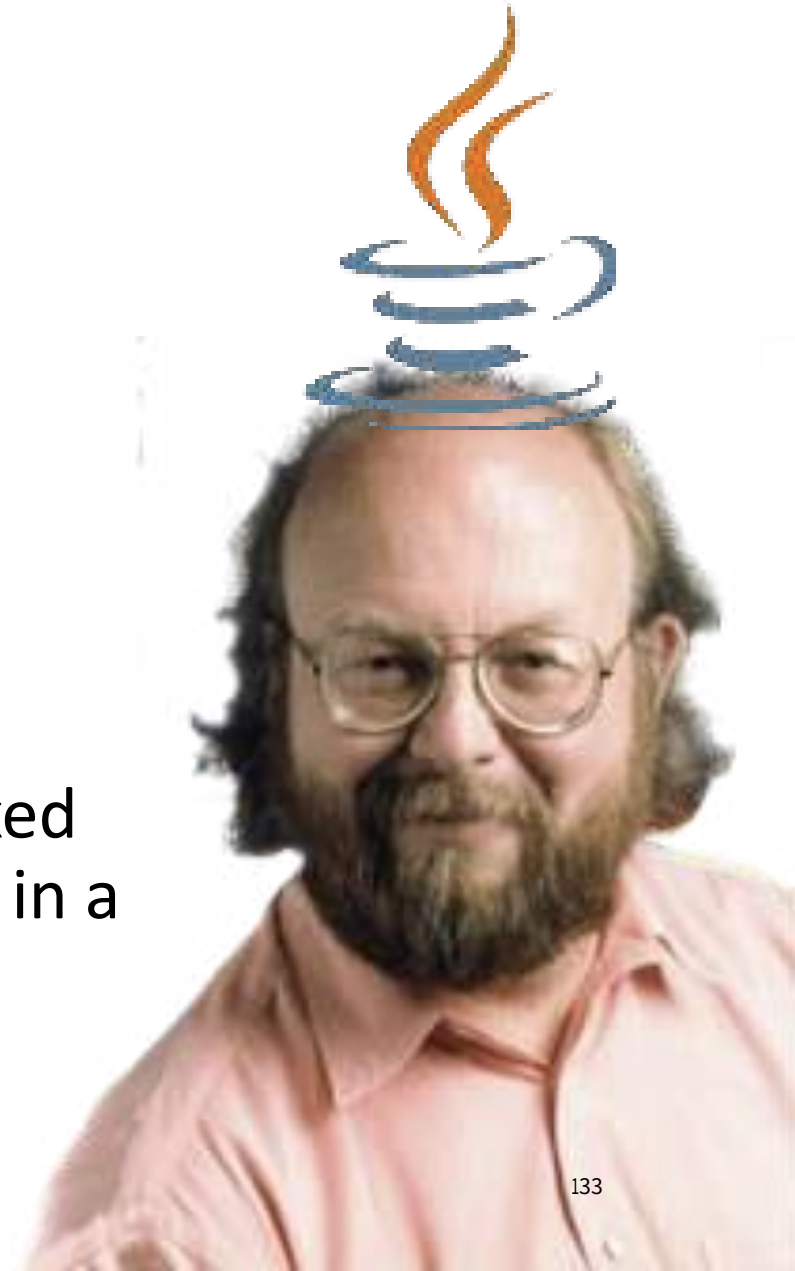
- They still are, to an extent, for specialized applications, and they are still the backbone of persistence.
- However, as an application developer, you are increasingly isolated from files.
- We saw with Image and Media you can add Properties, and we will see that in a similar way databases act as a layer between programs and files.
- A lot of data comes from networks as well.
- Every application, 40 years ago, used to open and close a lot of files, this is no longer the case. You open files mostly to load data in memory, and work there. All this is related to the cost of memory – if you are directly working with files the memory footprint of a program can be reduced…

In 1970 1 Mb cost ???

- When Dennis Ritchie created C memory was very very expensive.

- Programmers needed to be very careful to release memory as soon as it was no longer needed to try and save a few bytes…

In 1990 1 Mb cost ???



Twenty years later, James Gosling could take a more relaxed approach, have a garbage collector freeing memory once in a while, and afford the luxury of a 2- byte char.

2020: $0.10

- And today? Memory costs next to nothing. Just one problem: as the cost of memory was decreasing, applications were using more and more of it. And computers were supporting more and more users. You'd be wrong to believe that you no longer have to worry about memory. In some languages and environments (I'm thinking of Web servers running PHP) memory-per-user is limited to keep everything under control.
- But you don't need to fear if you sometimes load quite a lot of data in memory.

The result? A common way of working:

- Because memory is so cheap these days many people load everything to memory perform the work, and finally save everything to disk when done – or they use databases (which requires a different approach)

1. Load all data required initially
2. Work in memory (e.g. using Collections)
3. Save everything at the end
4. Use databases when data needs to be shared or it is in very large volumes beyond memory size

IMPORTANT

- When working with databases, the logic is very different from when you work with files - few Java developers understand it!

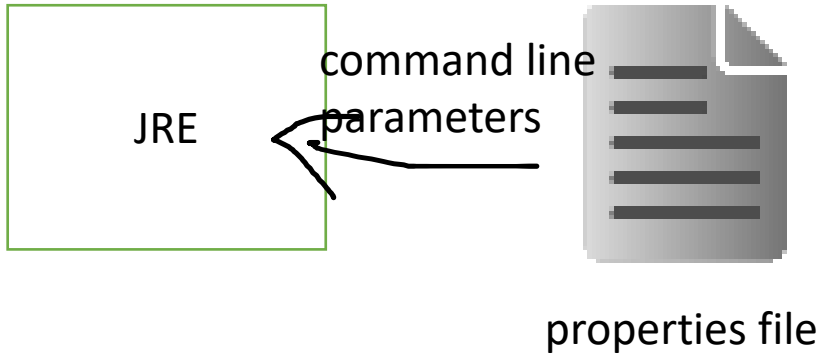
- The issue is that is the correct way of working with files is the wrong way when with databases
- This is because databases are **systems** that are optimized for data retrieval and processing and work much faster than anything you can do in Java

How are files largely used in industry?

- Multimedia documents are handled by specialized routines
- Parameters

We saw Media and Images in JavaFx as well as Property files. In many cases all the file reading is performed by a method in a specialized object and the developer doesn’t directly work with File I/O.

Parameter files



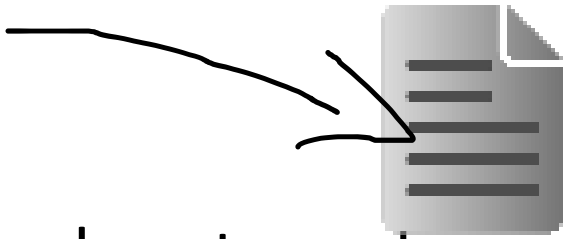
One important use of parameter files is when we would like a user to be able to modify important settings when they just have a .class file

- Specify directories
- Remote connections

This occurs when the user does not need and probably cannot use a JDK (ie they are not a java developer)

Parameter Files – Usage Pattern

- First you should assign some reasonable default values
- Then as required replace values and save in a parameter file

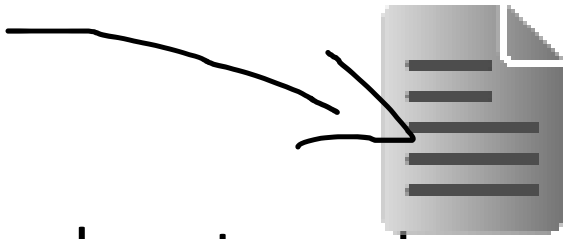


- The values to replace may be specified on the command line

Locations for resources should always be specified in property files

Parameter Files – Usage Pattern

- First you should assign some reasonable default values
- Then as required replace values and save in a parameter file



- The values to replace may be specified on the command line

Locations for resources should always be specified in property files

java.util.properties

- loading key/value pairs into a Properties object from a stream,
- retrieving a value from its key,
- listing the keys and their values,
- enumerating over the keys, and
- saving the properties to a stream.

https://docs.oracle.com/javase/8/docs/api/java/util/Properties.html

```
. . .
// create and load default properties
Properties defaultProps = new Properties();
FileInputStream in = new FileInputStream("defaultProperties");
defaultProps.load(in);
in.close();

// create application properties with default
Properties applicationProps = new Properties(defaultProps);

// now load properties
// from last invocation
in = new FileInputStream("appProperties");
applicationProps.load(in);
in.close();
. . .
```

Saving Properties

```
FileOutputStream out = new FileOutputStream("appProperties");
applicationProps.store(out, "---No Comment---");
out.close();
```

Getting property information

- contains(Object value) and containsKey(Object key)
  Returns true if the value or the key is in the Properties object. Properties inherits these methods from Hashtable. Thus they accept Object arguments, but only String values should be used.
- getProperty(String key) and getProperty(String key, String default)
  Returns the value for the specified property. The second version provides for a default value. If the key is not found, the default is returned.
- list(PrintStream s) and list(PrintWriter w)
  Writes all of the properties to the specified stream or writer. This is useful for debugging.
- elements(), keys(), and propertyNames()
  Returns an Enumeration containing the keys or values (as indicated by the method name) contained in the Properties object. The keys method only returns the keys for the object itself; the propertyNames method returns the keys for default properties as well.
- stringPropertyNames()
  Like propertyNames, but returns a Set<String>, and only returns names of properties where both key and value are strings. Note that the Set object is not backed by the Properties object, so changes in one do not affect the other.
- size()
  Returns the current number of key/value pairs.

Setting property values

- setProperty(String key, String value)
  Puts the key/value pair in the Properties object.
- remove(Object key)
  Removes the key/value pair associated with key.

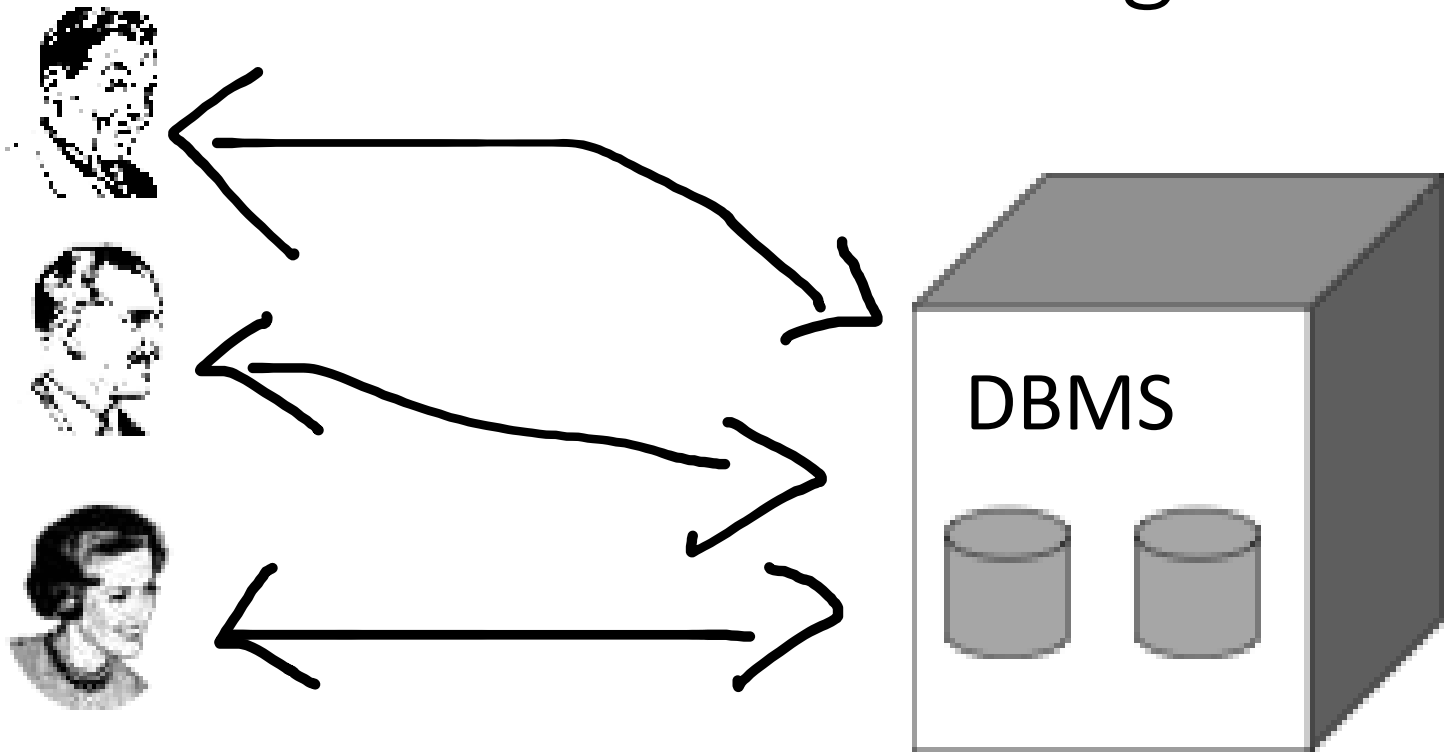
How are files largely used in industry?

- Multimedia documents are handled by specialized routines
- Parameters

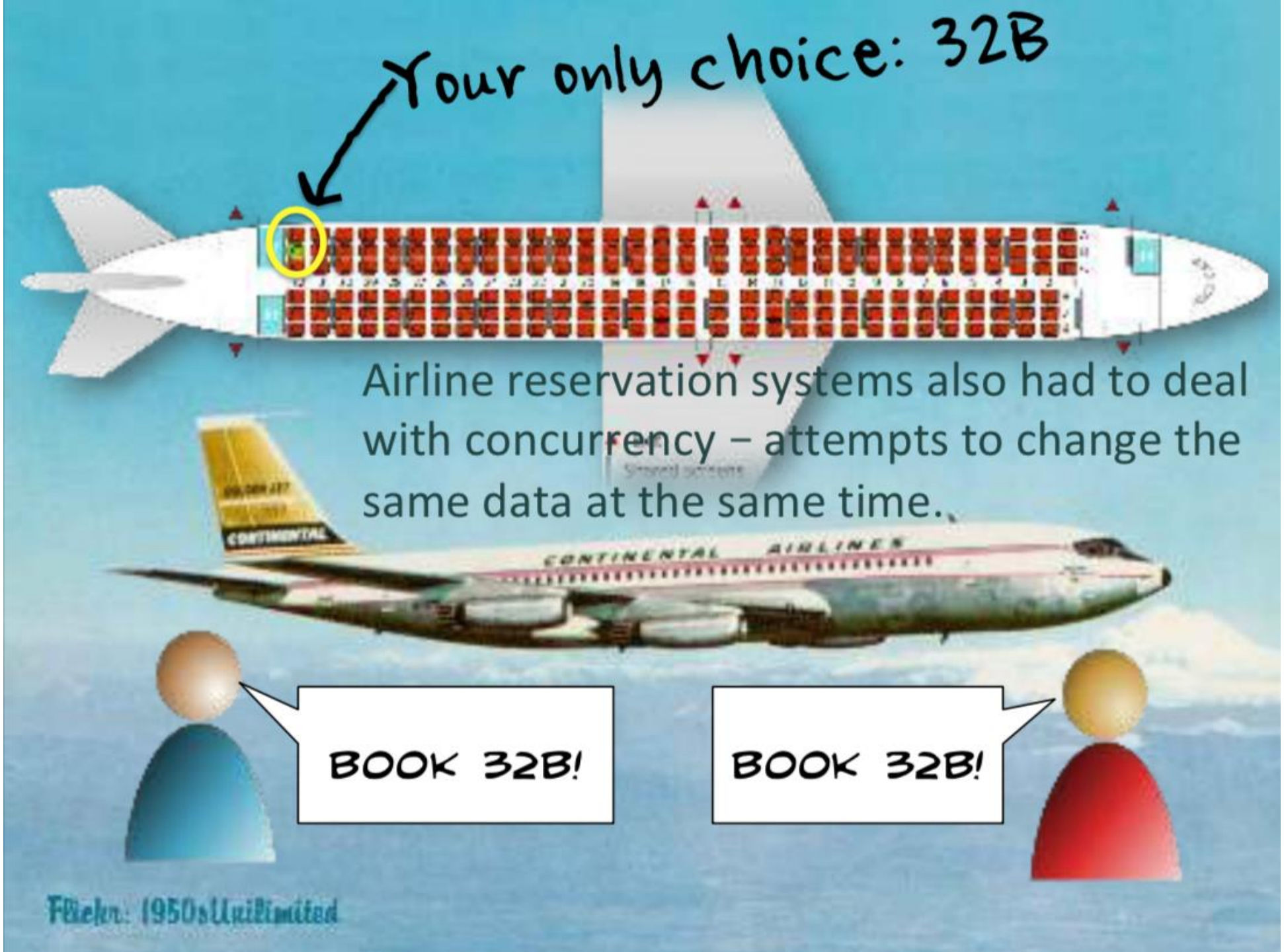
- Data Collection and Exchange

You mostly read and write data for exchanging data between systems – which include database systems.

DBMS – DataBase Management System



Very early systems were developed systems for managing files and retrieving data to abstract away “lower” software layers that deal with things like disk access and data retrieval… To avoid having to bother with lower software layers.


Your only choice: 32B
Airline reservation systems also had to deal with concurrency – attempts to change the same data at the same time.
CONTINENTAL AIRLINES
BOOK 32B!
BOOK 32B!
Flickr: 1950sUnlimited

CONTROL changes

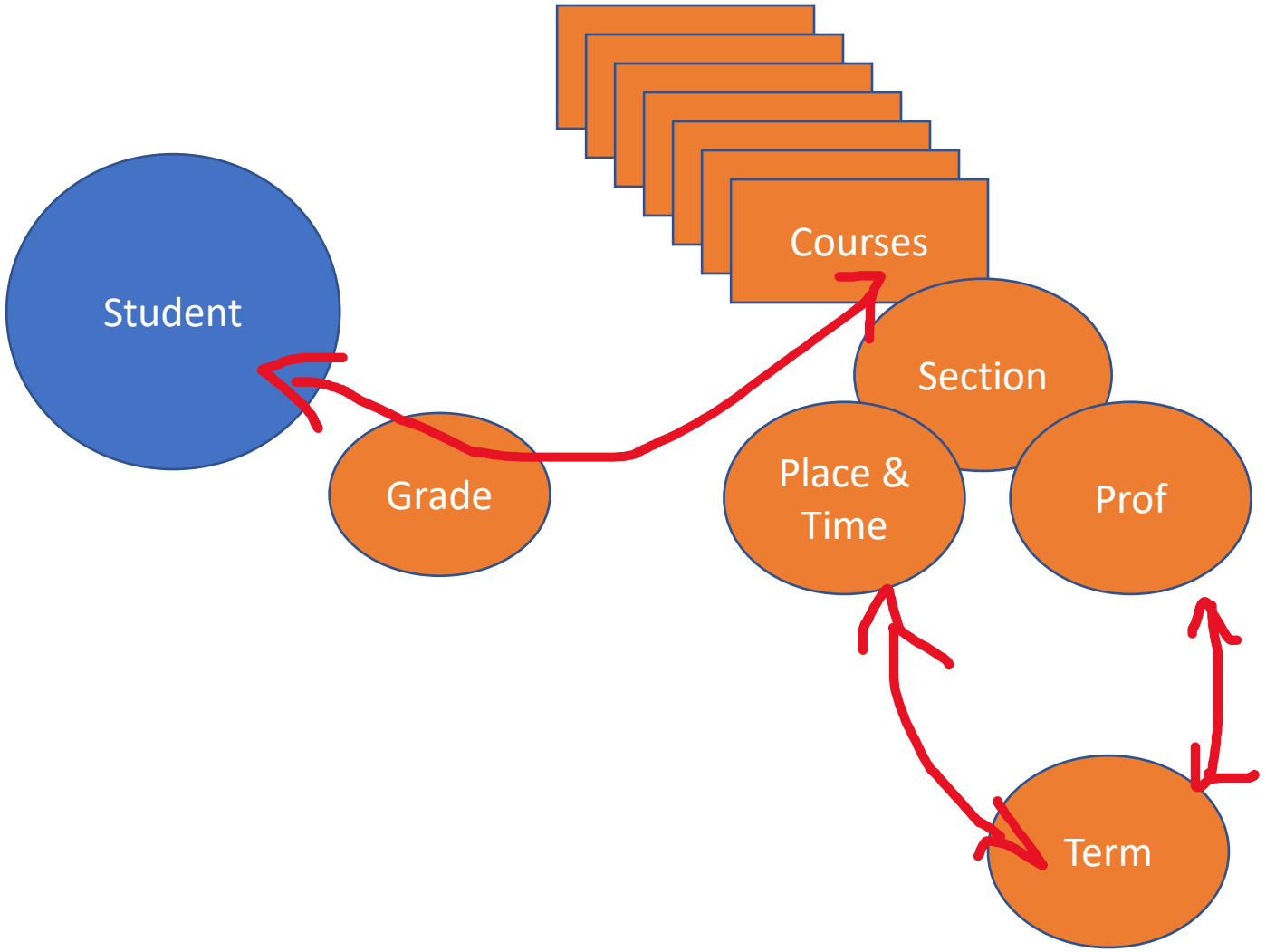
RETRIEVE data

This is what database systems where designed for: checking that changes don't lead to an inconsistent state (two people in the same seat, reservation for a non-existing flight), and also to retrieve data as fast as possible, using a "high-level" language (find this that satisfies those conditions ...) instead of looping on file records and checking each one.

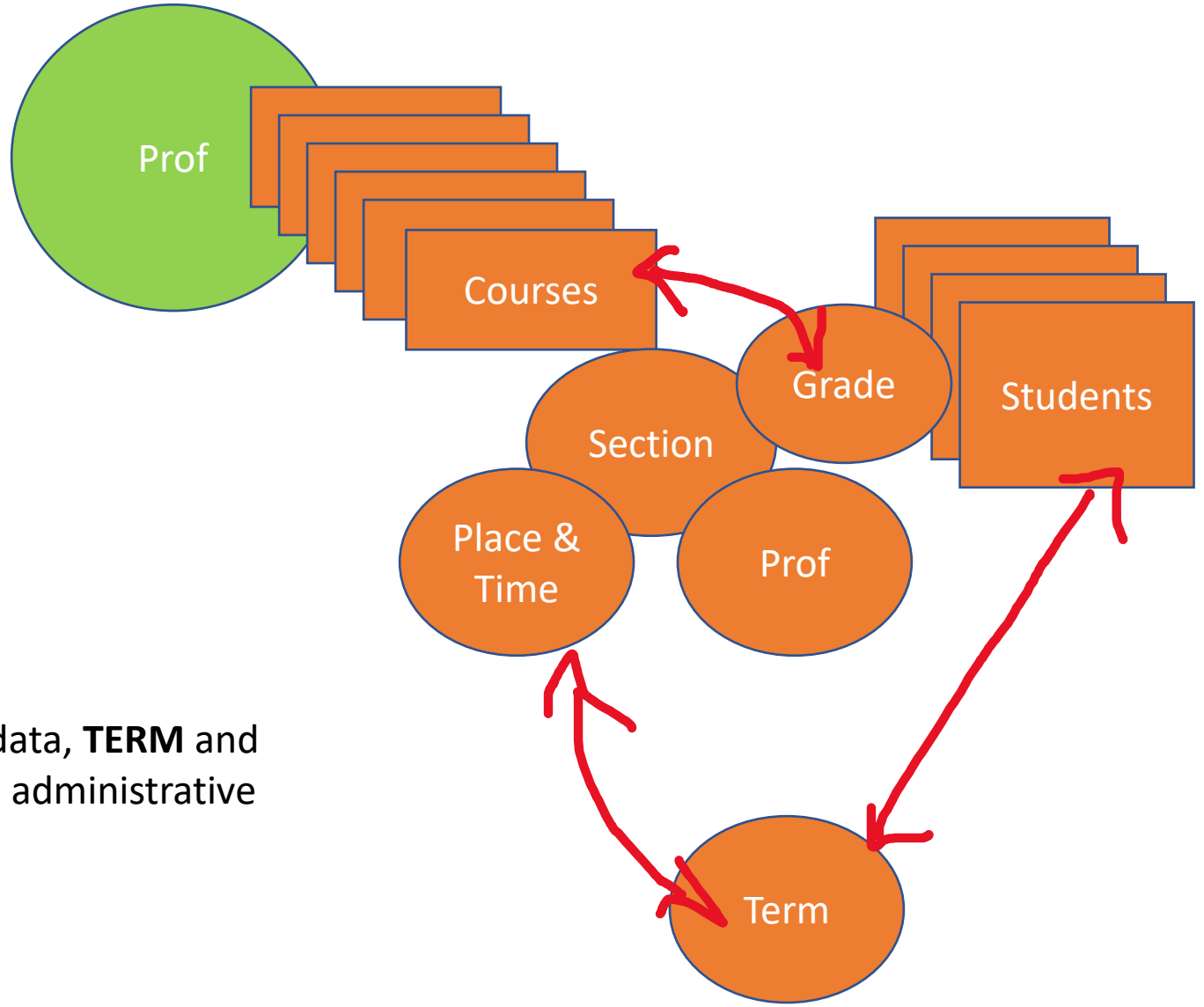
DIFFERENT VIEWS

A key problem with shared data is that people have different views of the same things. When you design an object for a single application, it's relatively easy.

For instance a Student object might contain a collection of Course objects, and the professor is an attribute of the course.



For a prof, each course is associated with a collection of students enrolled in the course. Same data, presented differently.

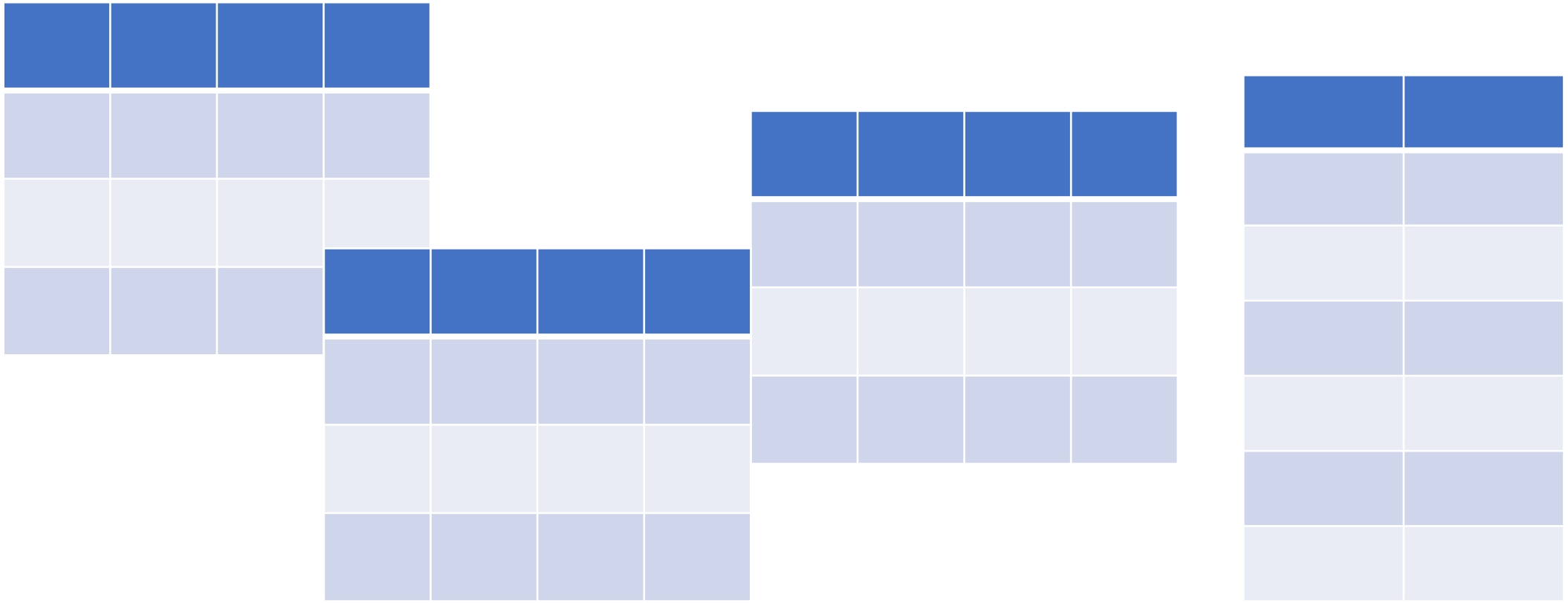


A database must
allow both views of the data, **TERM** and possibly more views (the administrative one ...)

Relational Databases

In 1970, a (British) IBM computer scientist described a way to store data in tables in a database, derived from the mathematical set theory. It was the start of a revolution in data management, and most databases in use today are based on his ideas.

Edgar F. Codd
1923 - 2003



Codd said that the database should give a tabular ("as tables") view of data, which is quite natural. Each table contains a set.

Sets

- Sets are unordered. If columns are ordered differently, the information isn't changed. Same with rows.

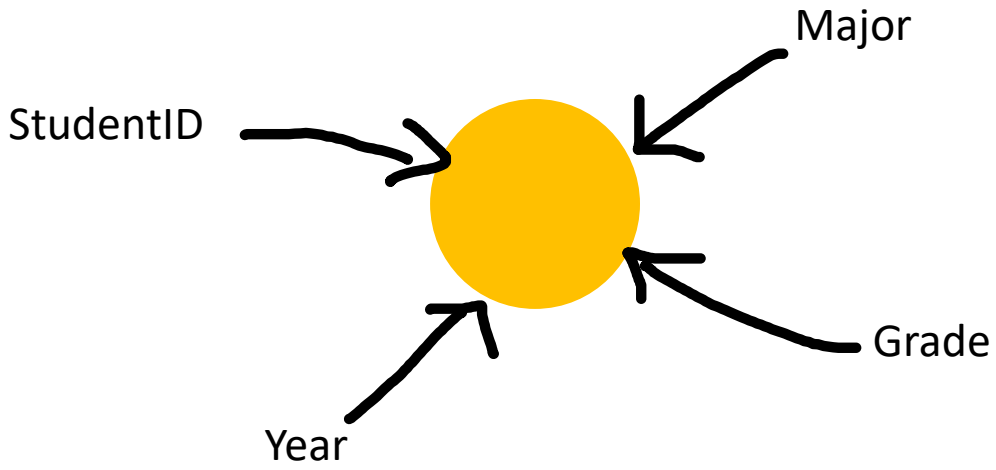
- And Sets don't allow duplicates. It's important that each different row is stored only once – this is done by defining "keys" that allow to specify one particular row.

Stduent ID	Major	Grade	Year

Relational Databases

Because all the attributes are related to a single event, the table is also known as a "relation", hence "relational theory of databases" and "relational database".

A Relation



Coming up next…

Relational Databases
JDBC – Java Database Connectivity API

2 – Relational Databases

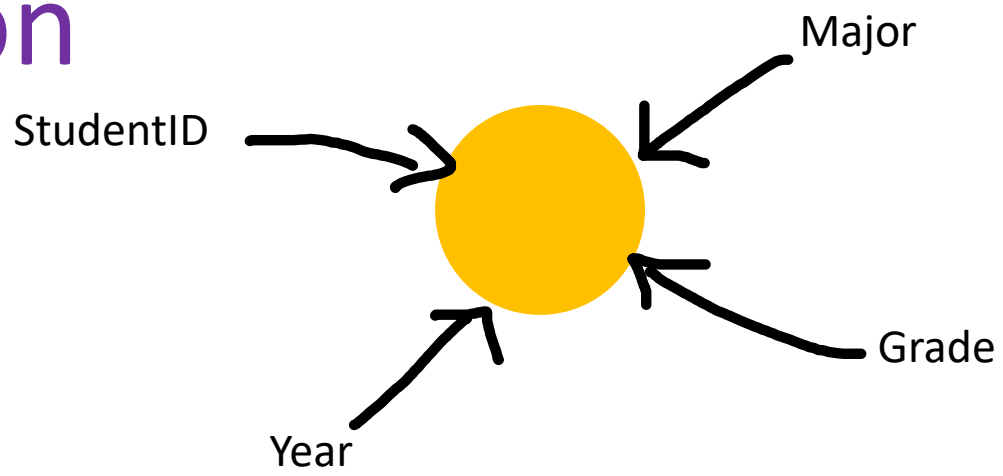
Lecture 9 Part 2

Relational Databases

Because all the attributes are related to a single event, the table is also known as a "relation", hence "relational theory of databases" and "relational database".

Stduent ID	Major	Grade	Year

A Relation



So…

- Table = Class
- Column = Attribute
- Row = Object or instance

… Right?

It is tempting to equate the database elements with the elements used in Object Oriented Programming. However this is misleading.

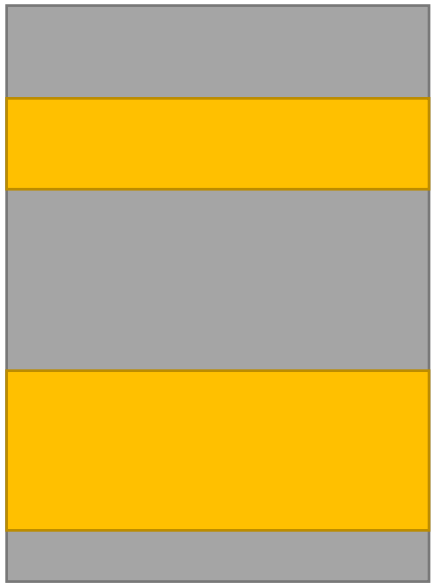
Big Mistake!

- That would be a very big mistake, unfortunately made by many people.
- Because in OO programming the focus is on the objects
- But in a relational database the focus is on the relations

OPERATE on relations

- Codd's big idea was to define operations on sets to define subsets.
- He insisted that links between pieces of data in different sets should be established by common data values.
- No references – Link through values

Relational Operations



Codd defined several operations, here are the 3 most important ones

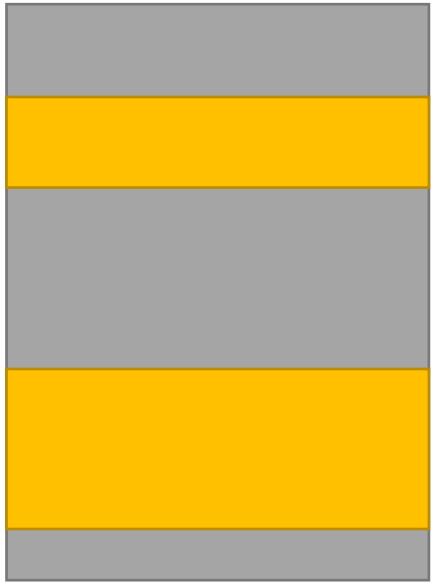
- Select
- Project
- Join

Select

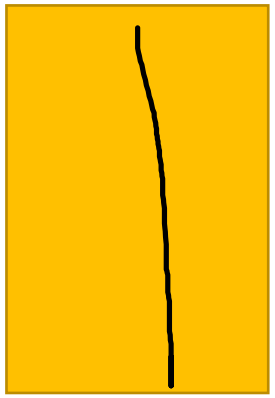
- Select rows with some particular values
- For example:

SELECT students with grade > X

(returns rows with grade more than X)

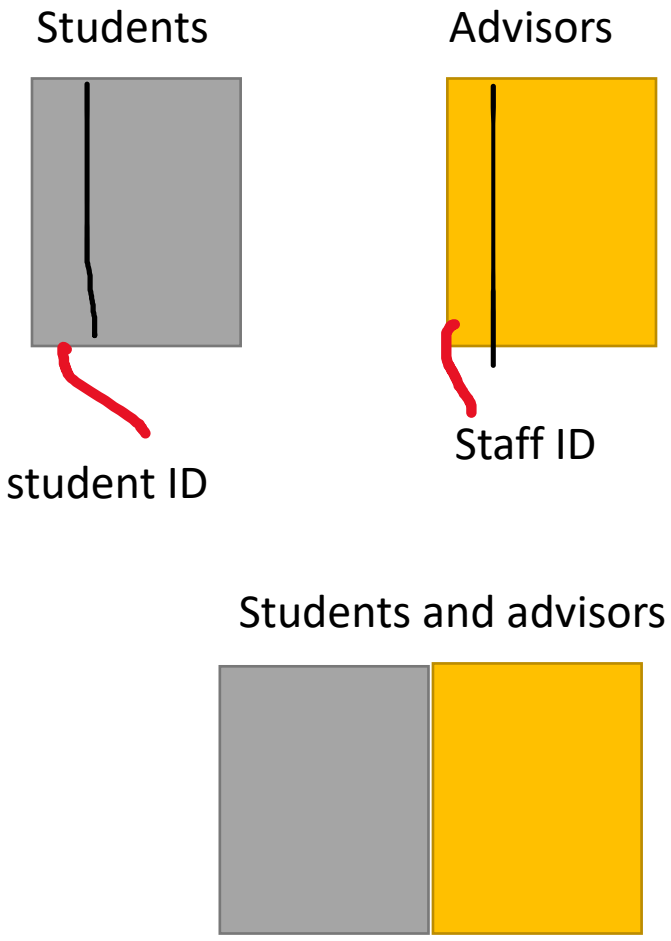


Project



- Get a subset of the columns
- In SQL a projection is done by selecting only certain columns

Join



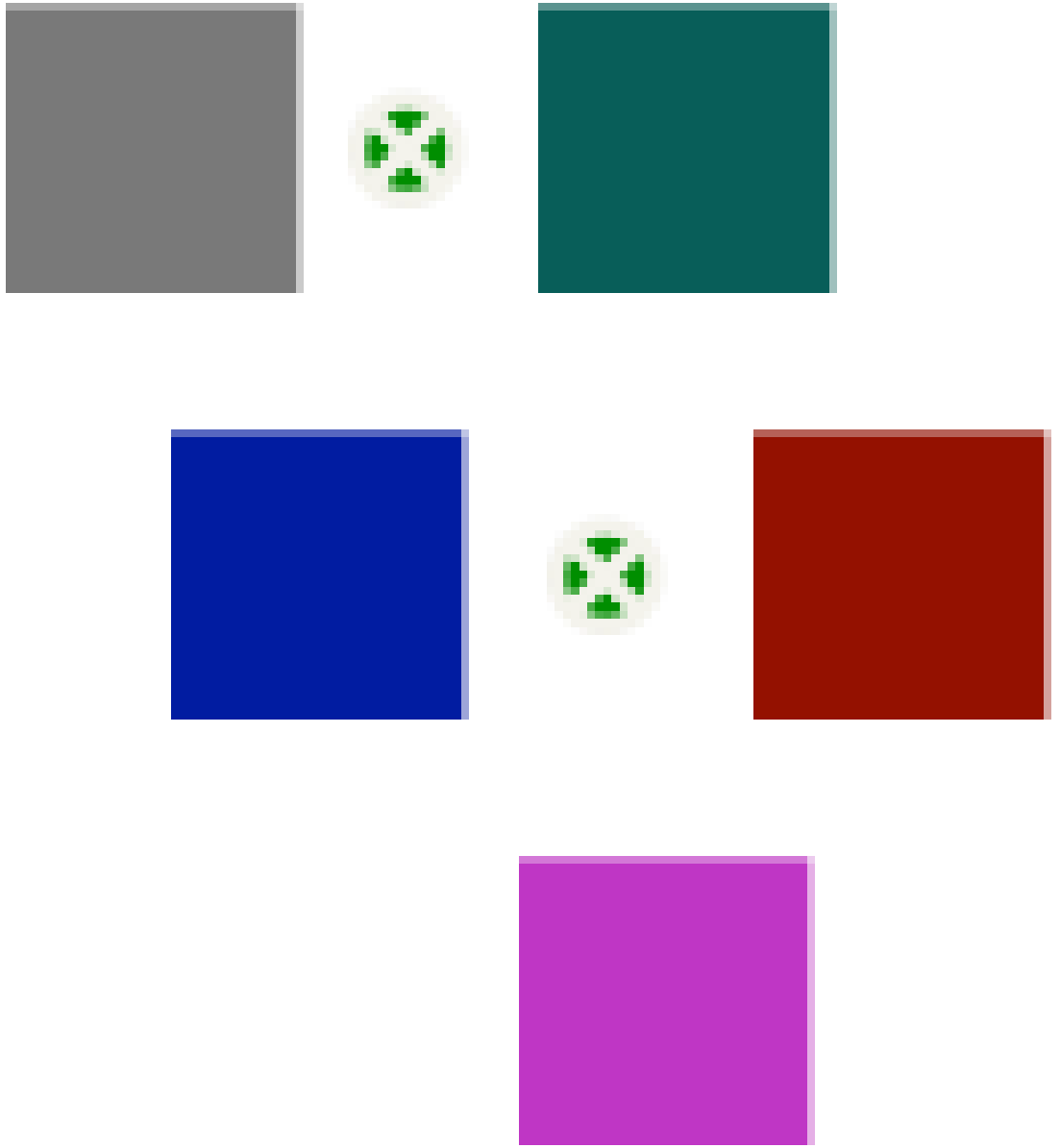
- Join 2 or more relations together
- For example here create a new relation with students and their advisors from two separate relations (students, advisors)
- The join depends on there being a key eg here the students table could have an advisor ID, then the joined relation would have additional information about the advisors for each student (from the advisor table)

Fundamental Concept

- Tables are like variables
- But instead of containing one value they contain one set

KEY POINT: Tables are Variables

And with operations you can combine sets to obtain a result set and then may combine this with further operations until you get exactly the result set that you want…



Good Modelling is Needed

One thing that is very important is that tables should be well designed (they must follow a number of rules, called normal forms). You don't want for instance data to be repeated many times, because it would make changes difficult. You must also know precisely what uniquely identifies a row (it may be all columns, but most often it's one or a few columns)

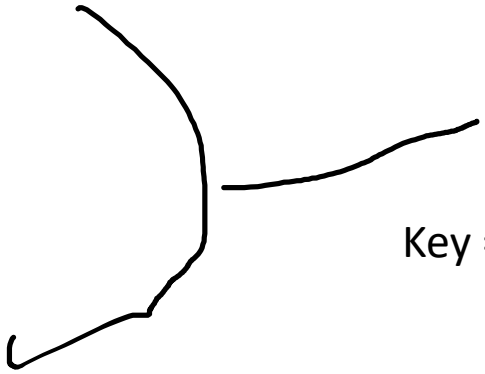
Entities = “Existing Things”

- Student
- Advisor
- Course

When you design one, you look for “entities”, things that exist independently of the others (a course can be on a catalogue without anyone taking it or teaching it). You must know what identifies each item: a code, a student/employee id, mail address, phone number may be good identifiers. A persons name is not good because several people could have the same name. You then have attributes of the entities (such as name). One entity will be one table.

Relationships

Then you can have relationships between entities – that link entities together. If an entity can be linked to only one other entity (for instance Student → Dormitory) it can be an attribute of an entity.



Session

Key = combination of entity keys (StudentID, Advisor ID, Course ID)

Often it will be a table relating identifiers because a student takes many course and there are many students in a course.

Normalization

- Distinguishing between what is an attribute and what should be in a relation is often difficult.
- All this business of organizing tables is rather difficult and often underrated.
- If poorly done it can cause many problems.

What identifies a customer?

- Name
- Email
- Phone number
- Generated customer ID

If you wanted to make a database of customers of an ecommerce site you could potentially use a name (but not unique), an email or a phone, but you are not supposed to modify an identifier. So it may be better to generate an ID.

SQL – Structured Query Language

SEQUEL: A STRUCTURED ENGLISH QUERY LANGUAGE

by

Donald D. Chamberlin
Raymond F. Boyce

IBM Research Laboratory
San Jose, California

Databases are usually associated with a query language called SQL. That was invented in the early 1970s.

It was originally proposed in the paper ”A Structured English Query Langauge” by Don Chamberlin and Ray Boyce in 1974: https://researcher.watson.ibm.com/researcher/files/us-dchamber/sequel-1974.pdf

ABSTRACT: In this paper we present the data manipulation facility for a structured English query language (SEQUEL) which can be used for accessing data in an integrated relational data base. Without resorting to the concepts of bound variables and quantifiers SEQUEL identifies a set of simple operations on tabular structures, which can be shown to be of equivalent power to the first order predicate calculus. A SEQUEL user is presented with a consis-

SQL

- SQL was designed to be a very simple language with a syntax similar to English that could be used by people who are not familiar with computers or programming.

```
SELECT ..
FROM …
WHERE …
```

- It became at the same time a major success and a major failure: today only computer programmers use SQL – but almost all of them have to use it, and sometimes very often.

Two Main Components

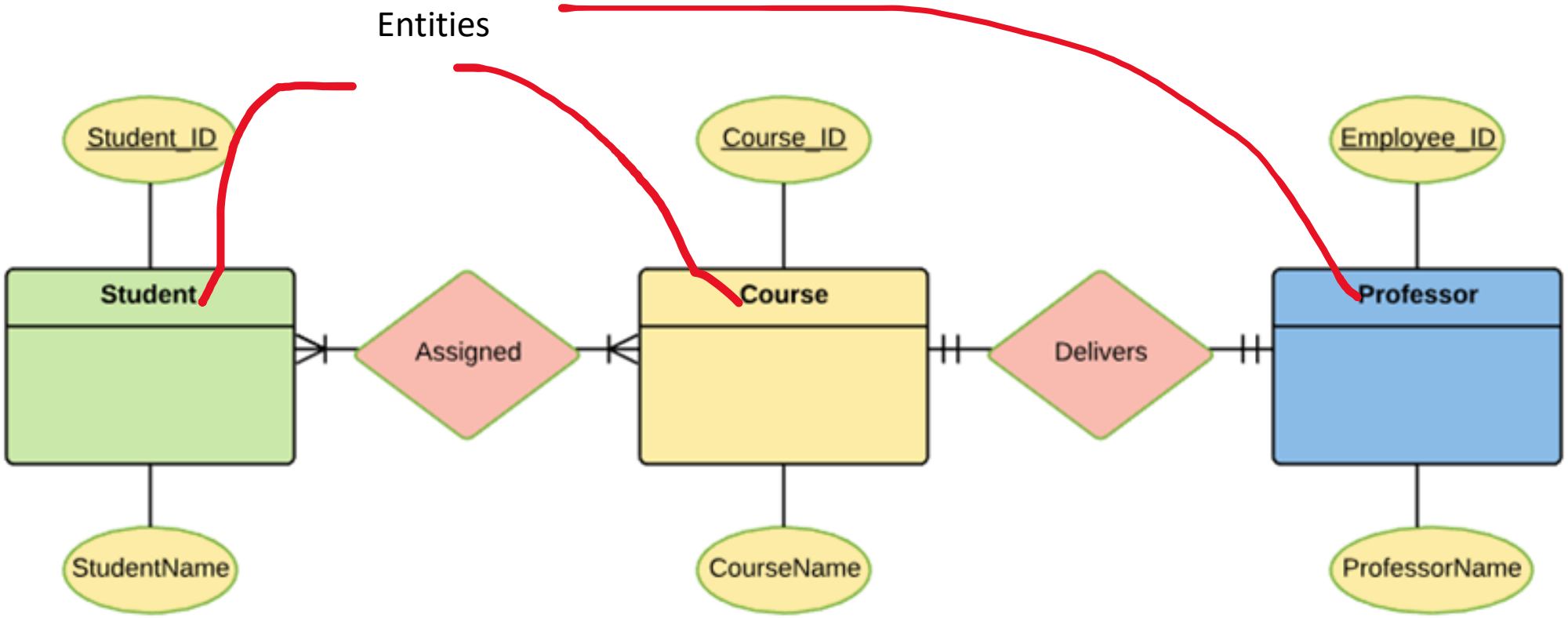
1. SQL provides commands for managing tables (creating, dropping, modifying them)
2. SQL provides commands for managing data (inserting new rows, updating or deleting existing ones – plus of course commands for retrieving data that satisfies some criteria).

You can learn SQL syntax in about the same time as you can learn how chess pieces move. As with chess, things however quickly become complicated.

Simple Database

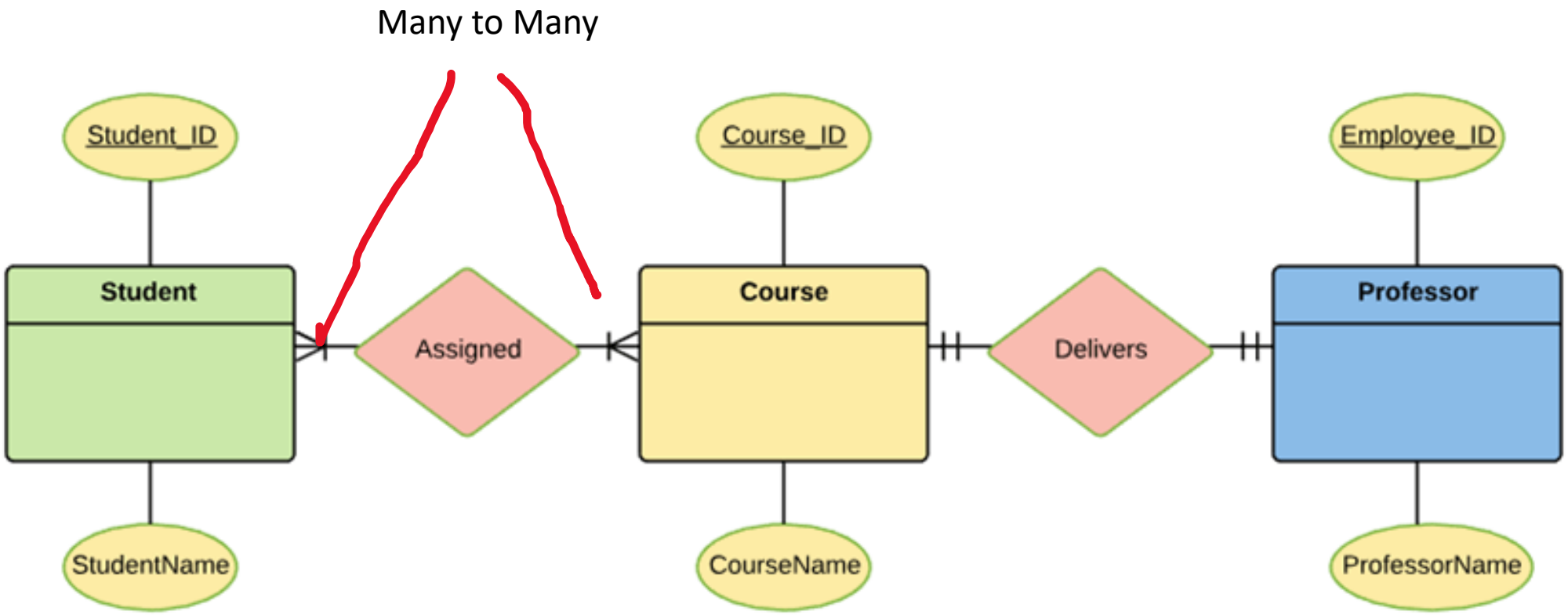
This small database represents Students, Courses and Professors.

Entity	Primary Key	Attribute
Student	Student_ID	StudentName
Professor	Employee_ID	ProfessorName
Course	Course_ID	CourseName



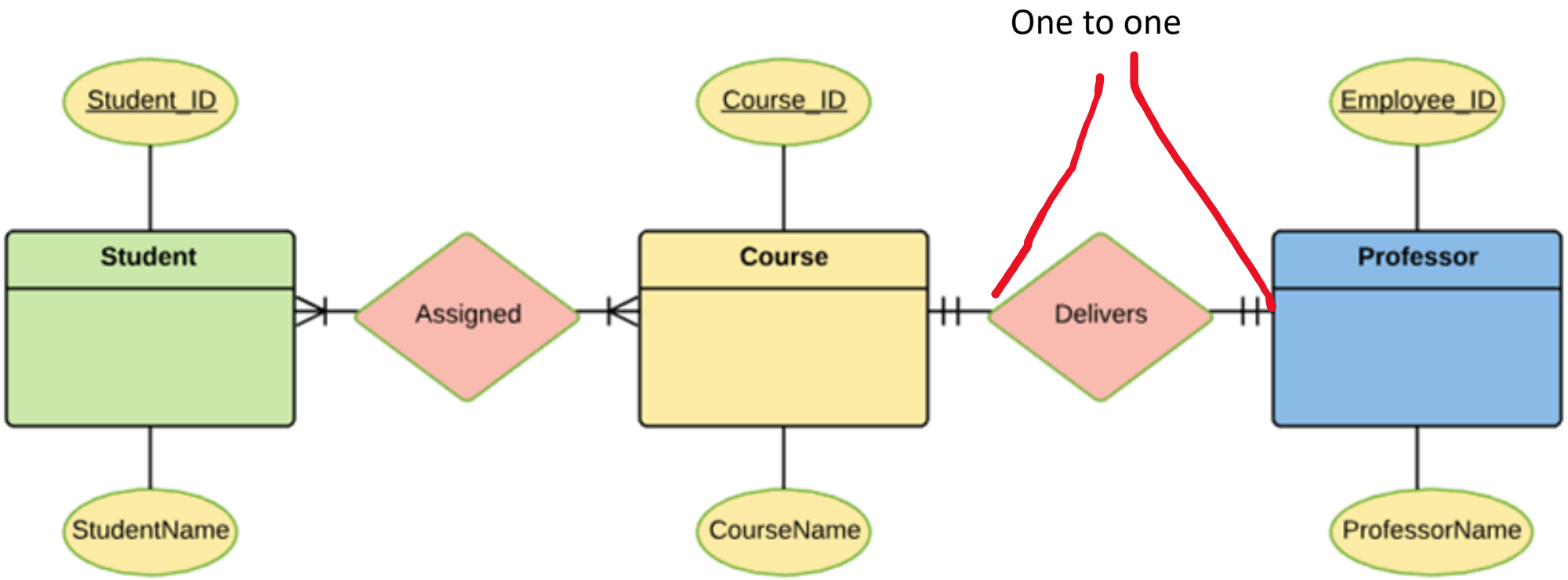
Simple Database

This small database represents
Students, Courses and Professors.



Simple Database

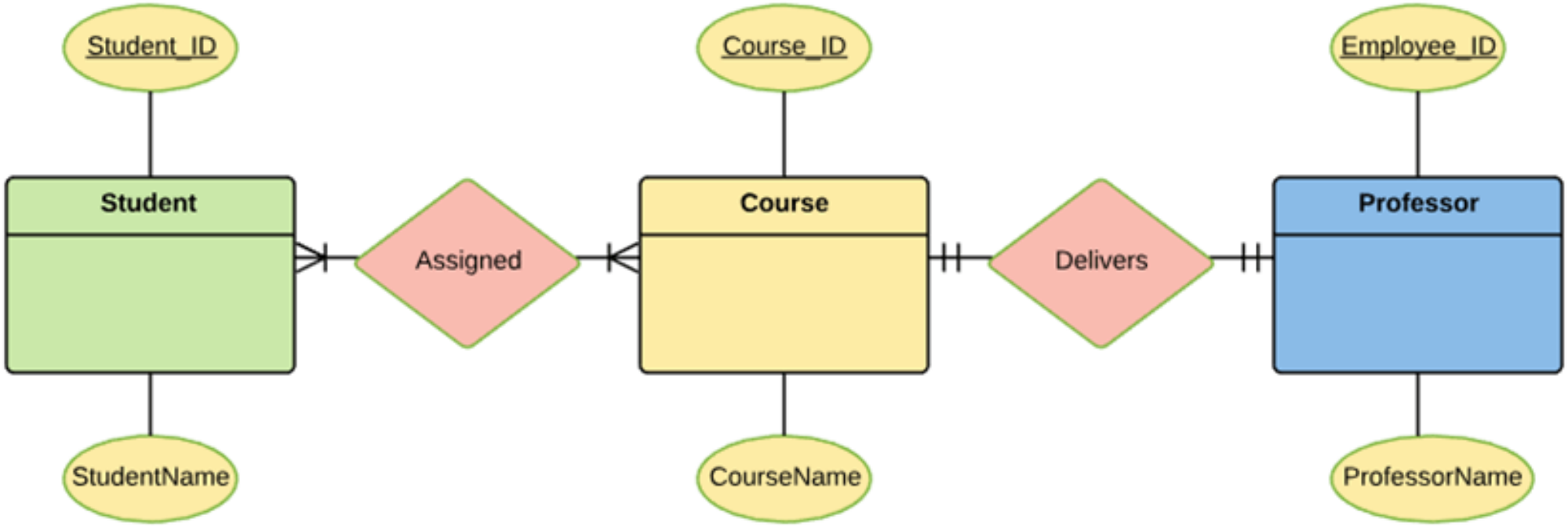
This small database represents
Students, Courses and Professors.

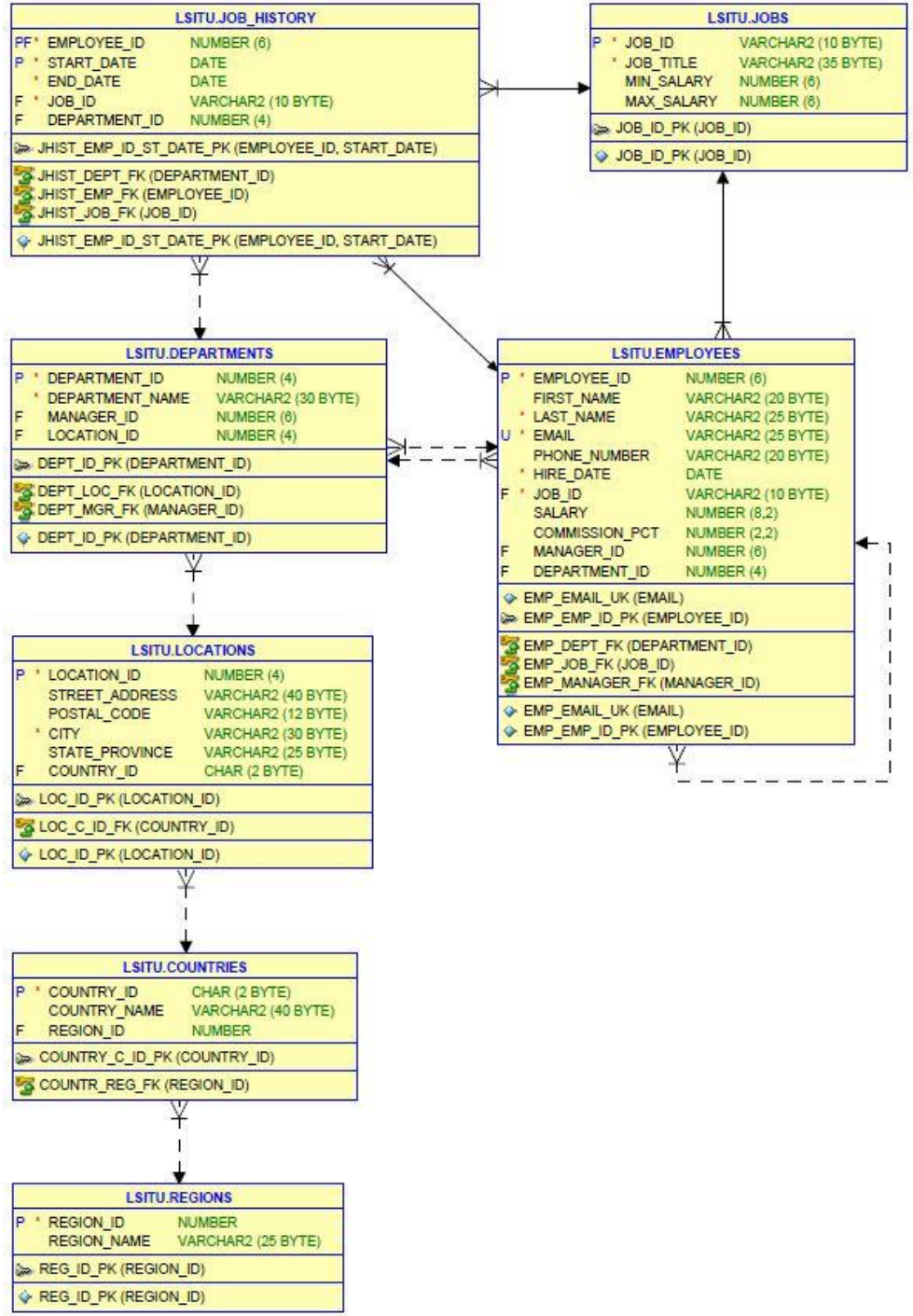


Simple Database

This small database represents Students, Courses and Professors.

Entity	Primary Key	Attribute
Student	Student_ID	StudentName
Professor	Employee_ID	ProfessorName
Course	Course_ID	CourseName



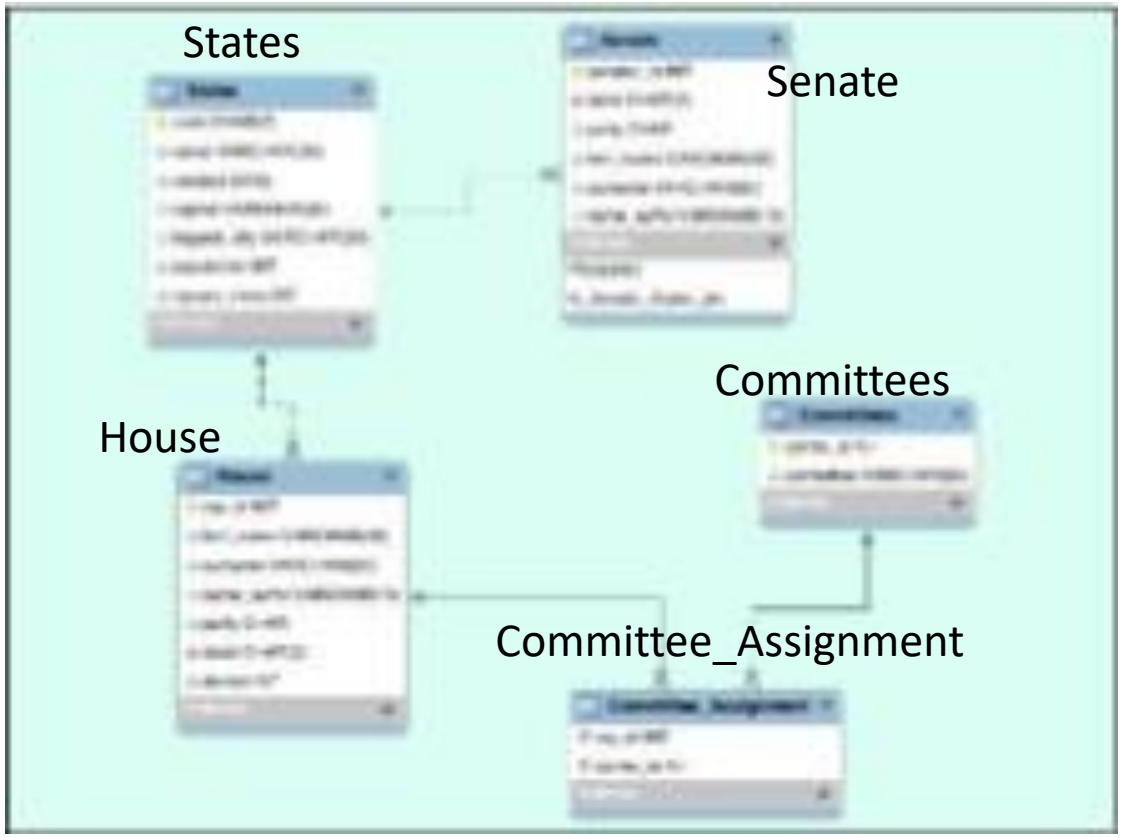


Here is another database which contains information about employees at a company. Not overly complex (7 tables with a few attributes in each).

And here is a query designed to fulfil a task that was "to display all employees and their related info even if some info is missing. Get as much information as you can about the employees".

```
SELECT
  e.employee_id AS "Employee #", e.first_name || ' ' || e.last_name AS "Name"
  , e.email AS "Email"  , e.phone_number AS "Phone"
  , TO_CHAR(e.hire_date, 'MM/DD/YYYY') AS "Hire Date"
  , TO_CHAR(e.salary, 'L99G999D99', 'NLS_NUMERIC_CHARACTERS = ''.,'' NLS_CURRENCY = ''$''') AS "Salary"
  , e.commission_pct AS "Comission %"
  , 'works as ' || j.job_title || ' in ' || d.department_name || ' department (manager: '
    || dm.first_name || ' ' || dm.last_name || ') and immediate supervisor: ' || m.first_name || ' ' || m.last_name AS "Curr
  , TO_CHAR(j.min_salary, 'L99G999D99', 'NLS_NUMERIC_CHARACTERS = ''.,'' NLS_CURRENCY = ''$''') || ' - ' ||
      TO_CHAR(j.max_salary, 'L99G999D99', 'NLS_NUMERIC_CHARACTERS = ''.,'' NLS_CURRENCY = ''$''') AS "Current Salary"
  , l.street_address || ', ' || l.postal_code || ', ' || l.city || ', ' || l.state_province || ', '
    || c.country_name || ' (' || r.region_name || ')' AS "Location"
  , jh.job_id AS "History Job ID"
  , 'worked from ' || TO_CHAR(jh.start_date, 'MM/DD/YYYY') || ' to ' || TO_CHAR(jh.end_date, 'MM/DD/YYYY') ||
    ' as ' || jj.job_title || ' in ' || dd.department_name || ' department' AS "History Job Title"
FROM employees e
-- to get title of current job_id
  JOIN jobs j
    ON e.job_id = j.job_id
-- to get name of current manager_id
  LEFT JOIN employees m
    ON e.manager_id = m.employee_id
-- to get name of current department_id
  LEFT JOIN departments d
    ON d.department_id = e.department_id
-- to get name of manager of current department
-- (not equal to current manager and can be equal to the employee itself)
  LEFT JOIN employees dm
    ON d.manager_id = dm.employee_id
-- to get name of location
  LEFT JOIN locations l
    ON d.location_id = l.location_id
  LEFT JOIN countries c
    ON l.country_id = c.country_id
  LEFT JOIN regions r
    ON c.region_id = r.region_id
-- to get job history of employee
  LEFT JOIN job_history jh
    ON e.employee_id = jh.employee_id
-- to get title of job history job_id
  LEFT JOIN jobs jj
    ON jj.job_id = jh.job_id
-- to get namee of department from job history
  LEFT JOIN departments dd
    ON dd.department_id = jh.department_id
ORDER BY e.employee_id;
```

If you want read more about this query and an explanation here (the purpose of the slide is just to give an example of how a query in SQL can become complex… https://dev.to/tyzia/example-of-complex-sql-query-to-get-as-much-data-as-possible-from-database-9he



is small database represents for instance part of the US Congress, with the Senate and House of Representatives.

Who participates in the greatest number of House Committees ?

```
1   select h.first_name, h.surname,
2                s.name as state, z.num_of_committees
3   from (select rep_id,
4           count(*) as num_of_committees
5       from committee_assignment group by rep_id
6       having count(*) =
7           (select max(num_of_committees) from (select rep_id,
8           count(*) as num_of_committees from committee_assignment
9           group by rep_id) x)) z
10      join house h
11          on h.rep_id = z.rep_id
12      join states s
13          on s.code = h.state
14  by h.surname, h.first_name, s.name
15
16
```

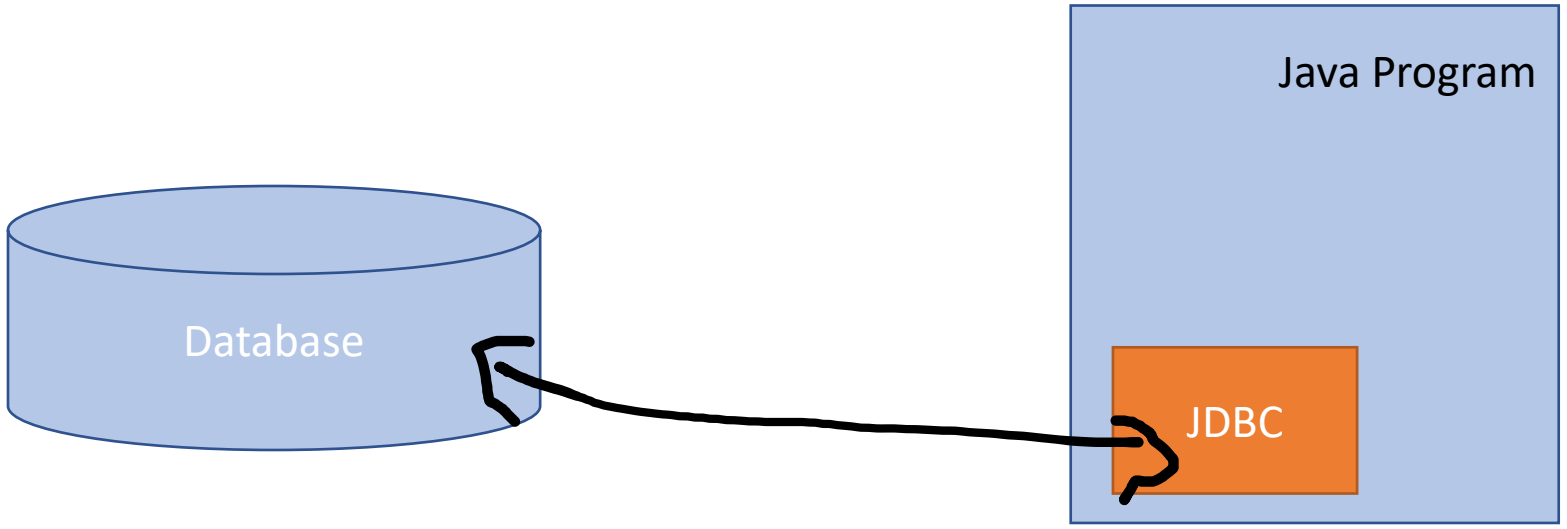
Next

Accessing databases in java

JDBC

Lecture 9 – Part 3

Java Database Connectivity (JDBC)

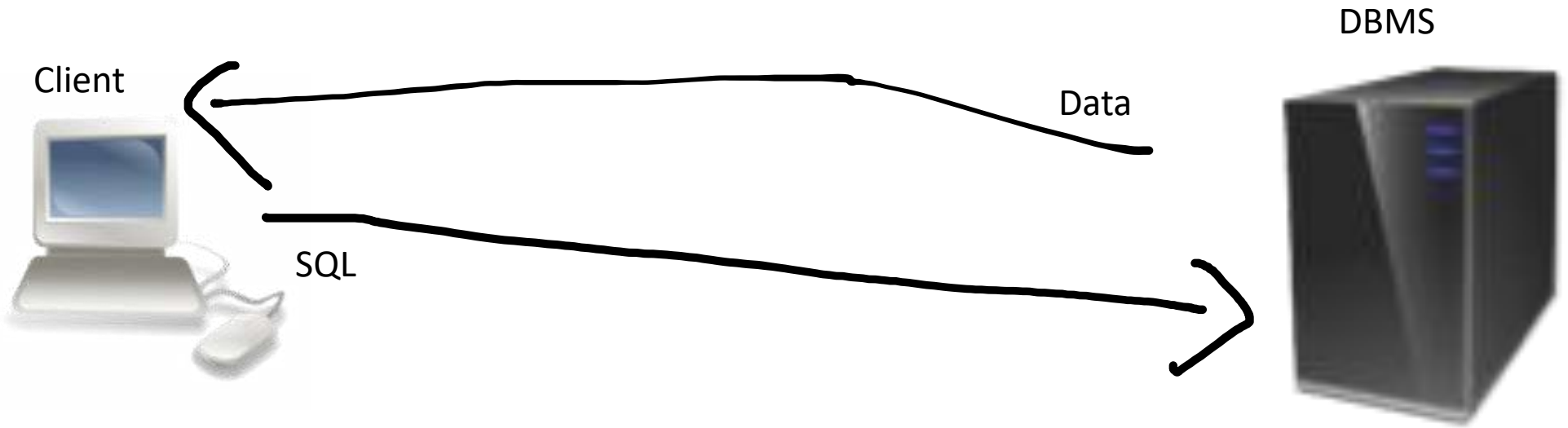


Accessing a database and sending/executing SQL from a Java program is easy, even without using some tools that try to write queries for you (sometimes these tools even write inefficient queries).

Accessing a database from a Java program

Classic case for a real database management server:

Your program must first connect to the database (which usually required supplying a username and a password) then will send strings that contain an SQL command and will be executed by the server. Some commands will only return success or failure (creating a table, for instance). A command such as an update will be able to tell you how many rows were changed. A query will return you rows, between 0 and n...



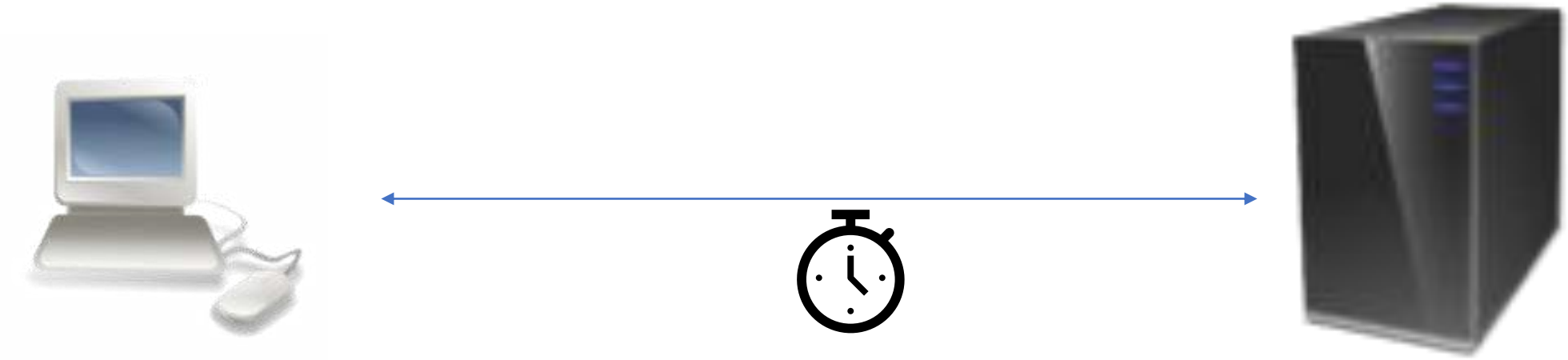
System Architecture: Classic Design Mistake

- Many Java developers do this wrong…
- Must SHARE the work between what the database does ...
  - Retrieve exactly what you need

  If you want a sum, don't retrieve all the data and iterate to sum it. Ask for the sum. The DBMS can compute it.

- And what the java program does
  - Presentation computations that cannot be performed by the DBMS

System Architecture: Consider Network Latency



Don't multiply exchanges between your application and the server. Travelling in networks takes time (it's called latency), even with a lot of bandwidth. Sometimes data centres are very far away.

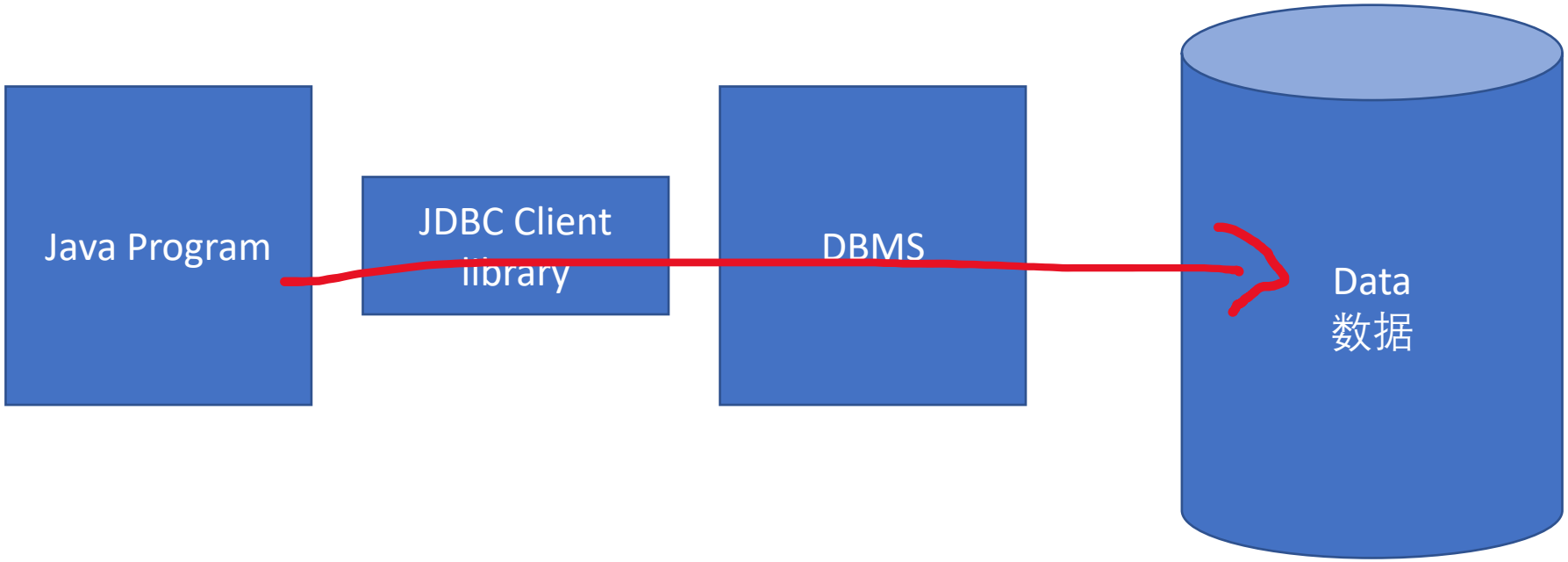
System Architecture: One INSERT is okay but 10K INSERTs is BAD

If you want to insert many rows into a remote database, send all the rows to the server and ask it to insert all the rows at once (this is called "batching"); it will do it very fast. If you insert rows one by one, you have to wait each time for the response from the server. A return trip between Singapore and Tokyo takes about 0.075s. If you assume that inserting 10,000 rows takes 0.050s* doing it as one batch between Singapore and Tokyo would take 0.125s. Inserting one row about 0.003s*. Inserting 10000 rows row by row would be (0.075 + 0.003) * 10000 = 780s, or 13 minutes ...

USE BATCHING

*tests by Stephane Faroult in 2017

Even when you are close…



Even when running on the same server, switches between a program and the DBMS are costly.

System Architecture Rules

- Share work between the DBMS and the program
- Consider network latency
- Use batching

Embedded Databases

An alternative to real database servers are "embedded databases", systems that let you use of file as if it were a database.

No server just for a single-user

"Connection" same as opening a file

Everything else like the real thing

Apache Derby

- Pure Java
- Part of the JDK
- Server or embedded

- Java is shipped with "Derby", sometimes called "Java DB"

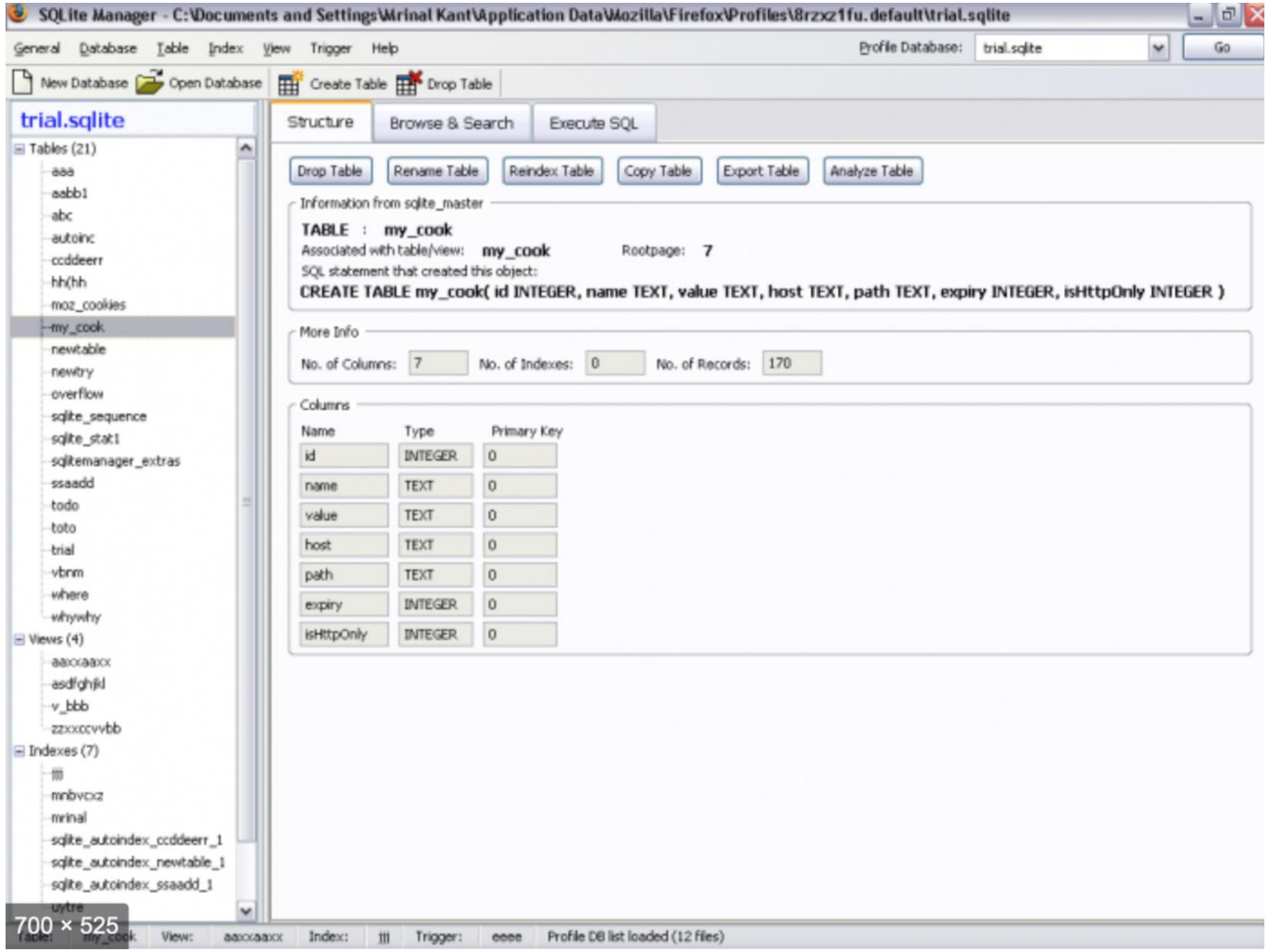
SQLite

- Public domain
  One file to download

- Used in mobile apps – and by Mozilla

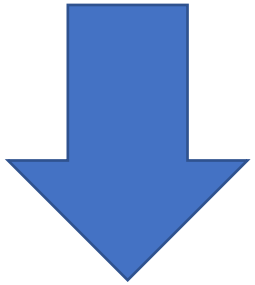
- Create tables in *SQLite Manager* (a firefox plugin)

The most popular embedded database is SQLite.



- Can directly execute the SQL from the manager

www.sqlite.org



Only for C programming language

JDBC: https://github.com/xerial/sqlite-jdbc

You just need a .jar file available from this address…

Java Database Connectivity (JDBC)

sqlite-jdbc refers to JDBC, which is a protocol allowing access to almost all databases from Java. SQL is often slightly different from database to database, not JDBC calls.

JDBC

- Load a specific connection driver (a .jar file)

  Database specific connection parameters

- Then java methods are the same with every database

- SQL is slightly different between databases

3 Main Objects

- **Connection**
  Link to a database
- **PreparedStatement**
  SQL commands:
- **ResultSet**
  Returned results

You can execute a query and do things like loop on rows returned. There is also a Statement object. A PreparedStatement can be re-executed with different parameters, but a Statement is only sent once.

Java/JDBC Example

```
import java.util.Properties;
import java.sql.*;
import java.util.Scanner;

class DBExample {
    static Connection con = null;

    public static void main(String arg[]) {
        Properties info = new Properties();
        String url = "jdbc:oracle:thin:@localhost:1521:orcl
        Scanner input = new Scanner(System.in);

        try {
            Class.forName("oracle.jdbc.OracleDriver");
        } catch(Exception e) {
            System.err.println("Cannot find the driver.");
            System.exit(1);
        }
```

- JDBC
- Caution: the driver must be in the classpath
- You connect to the database using a url string not a URL object…

Java/JDBC Example

```
try {
    Class.forName("oracle.jdbc.OracleDriver");
} catch(Exception e) {
    System.err.println("Cannot find the driver.");
    System.exit(1);
}

try {
    System.out.print("Username: ");
    String username = input.nextLine();
    System.out.print("Password: ");
    String password = input.nextLine();
    info.put("user", username);
    info.put("password", password);
    con = DriverManager.getConnection(url, info);
    con.setAutoCommit(false);
    System.out.println("Successfully connected.");
} catch (Exception e) {
        System.err.println(e.getMessage());
        System.exit(1);
}
```

- MySQL wants username and password independent parameters,
- Oracle passes them as Properties.
- Of course the connection can fail.

Java/JDBC Example

```
System.out.print("Date: ");
String utcDate = input.nextLine();
try {
    PreparedStatement stmt = con.prepareStatement("select region, count(*)"
            + " from quakes"
            + " where UTC_date >= ?" + " group by region");
    stmt.setString(1, utcDate);
    ResultSet rs = stmt.executeQuery();
    while (rs.next()) {
        System.out.println(rs.getString(1) + "\t" + rs.getString(2));
}
rs.close();
```

- Queries are simple. Placemarkers for parameters in a PreparedStatement are ? marks, implicitly numbered (from 1). Return columns are also numbered from 1. One can often only work with Strings, the DBMS can convert types.

Java/JDBC Example

```
            rs.close();
    } catch (Exception e) {
        System.err.println(e.getMessage());
        try {
            con.close();
        } catch (SQLException sqlE) {
        // Ignore
        }
        System.exit(1);
    }
    try {
        con.close();
    } catch (SQLException sqlE) {
    // Ignore
    }
```

Queries can fail too, but returning nothing or updating or deleting or inserting no rows doesn't throw any exception because the empty set is a valid set … However trying to insert the same key for instance will throw an exception

Common drivers

```
Class.forName("oracle.jdbc.OracleDriver");
Class.forName("com.mysql.jdbc.Driver");
Class.forName("org.postgresql.Driver");
Class.forName("org.sqlite.JDBC");
Class.forName("org.apache.derby.jdbc.EmbeddedDriver");
```

These are JDBC driver names for commonly use Database Management systems (note that for Derby there are two modes, embedded – single user – and not embedded). Note also that although most tutorials use reflection to load the driver, you can also use import with it, especially with Derby, as the driver will always be in your class path.

Connection Examples

```
con = DriverManager.getConnection(url, info);
    "jdbc:oracle:thin:@hostname:port:dbname"
    "jdbc:postgresql://hostname:port/dbname"
     user password

 con = DriverManager.getConnection(url,
                                   username,
                                    password);
    "jdbc:mysql://hostname:port/dbname"

 con = DriverManager.getConnection(url);
    "jdbc:sqlite:filename"
```

- And here are connection examples.
- With Sqlite you just provide a filename. It will be created if it doesn't exist already.

Additional practice: practical