Enterprise Architectures

Petr Křemen

petr.kremen@fel.cvut.cz

Winter Term 2017

Contents
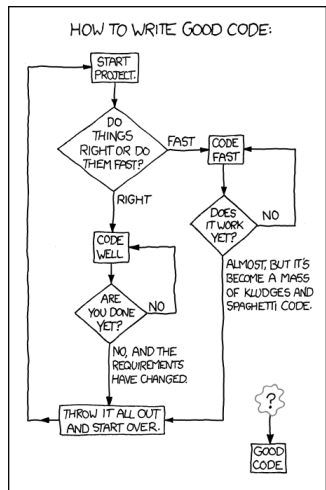
Information about the course

You will learn how to

design enterprise applications using Java web technologies, including pieces of the Java EE stack

implement the applications in Java, Spring, EclipseLink, and ReactJS

think about high-availability, clustering, security, and other stuff ...

HOW TO WRITE GOOD CODE:

source: https://techcodegeek.wordpress.com

Teachers

Lecturers:

- Petr Křemen, petr.kremen@fel.cvut.cz
- Miroslav Blaško, miroslav.blasko@fel.cvut.cz
- Martin Ledvinka, martin.ledvinka@fel.cvut.cz

Course Assistants:

- Petr Křemen, petr.kremen@fel.cvut.cz
- Miroslav Blaško, miroslav.blasko@fel.cvut.cz
- Martin Ledvinka, martin.ledvinka@fel.cvut.cz
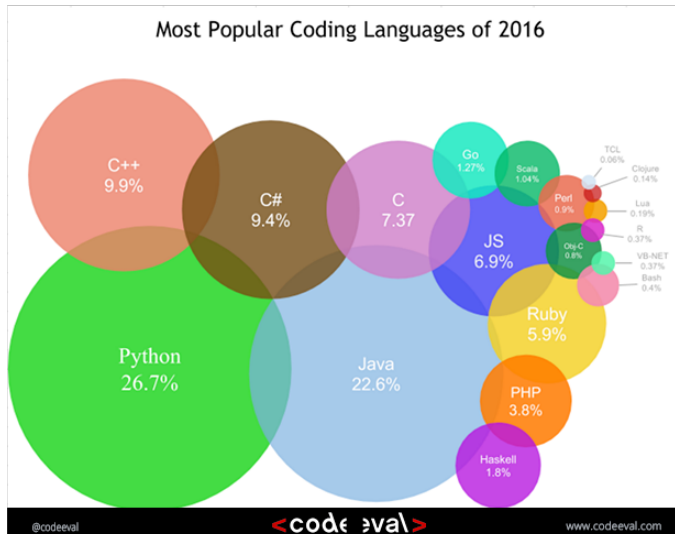- Bogdan Kostov, kostobog@fel.cvut.cz

Course Organization

- **Go through
  `https://cw.fel.cvut.cz/wiki/courses/b6b33ear`
  carefully, including subsections:**
    - Lectures

      https://cw.fel.cvut.cz/wiki/courses/b6b33ear/lectures
    - Seminars

      https://cw.fel.cvut.cz/wiki/courses/b6b33ear/seminars
    - Assessment

      https://cw.fel.cvut.cz/wiki/courses/b6b33ear/hodnoceni

- The course will be split into two parts:

  Basic topics – lectures 1-7
  Advanced topics – lectures 8-13

- 14th week – Panel Discussion with people from commercial sector

Enterprise Applications

Usage of programming languages in 2016
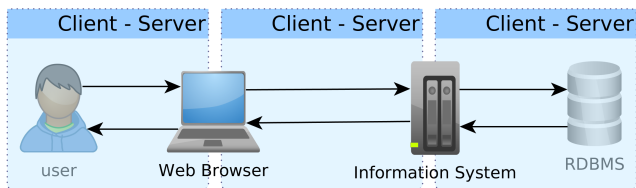

source: http://www.codeeval.com

Java

- simple, designed for small devices
- portability (bytecode)
- runtime optimization (compilation)
- public specifications JSR based on community discussion
- editions
  - Java ME
  - Java SE – Java 1.x, J2SE 1.2 & 1.3 & 1.4, Java SE 5 & 6 & 7 & 8
  - Java EE – J2EE 1.2 & 1.3 & 1.4, Java EE 5 & 6 & 7
  - (Android)

Client - Server Paradigm

Desktop Application
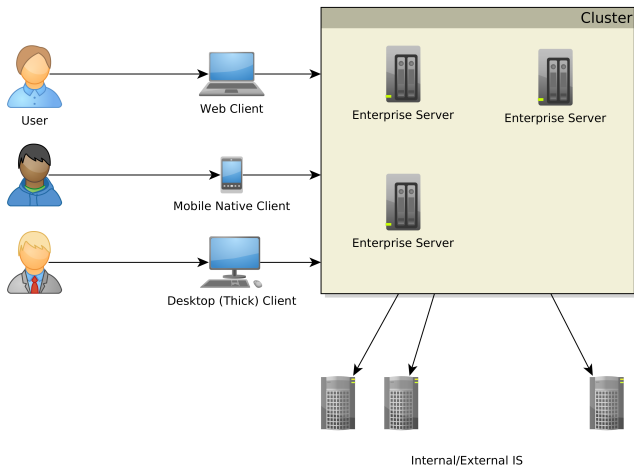

Desktop Application. Single-user access.

Web Application

Web Application. Multi-user access, single client (web), no integration with other systems.

Enterprise Application (EA)

Web Application. Multi-user access, multiple clients (web, mobile, desktop, terminal ...), integration with other enterprise systems (ERP, DWH, ...).

Multi-tier Architecture

Typically three-tier – presentation logic, business logic, data storage.
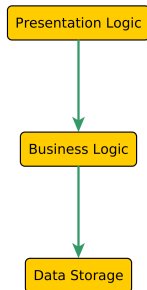**Separation** of tiers to

1. presentation
2. service
3. business logic
4. data access
5. data storage
6. ...

Each tier can access only a tier right below (**strict**), or all lower tiers
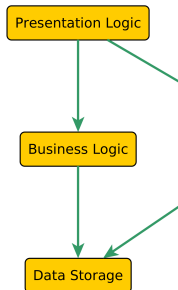(**relaxed**).
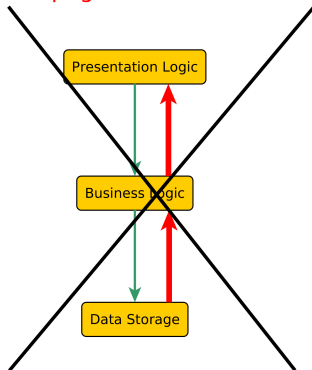
Multi-tier Architecture

Strict Architecture

Relaxed Architecture

Spaghetti Code

Enterprise Application Architecture

Martin Fowler: Patterns of Enterprise Application Architecture

"… display, manipulation and storage of large amounts of complex data and the support or automation of business processes with that data."

Enterprise Applications - Requirements

Persistent Data using relational databases, graph databases, NoSQL
databases, RDF triple stores,

Complex Data Integration of different volume, accurracy, update
frequency, quality and meaning $\rightarrow$ data integration,

Concurrent Data Access by many users at once with different scenarios
(writing, reading different parts of data),

Multiple Input Interfaces involving complex user interfaces (many forms,
web pages), (sensoric) data sources, operational data,

Process Automation involving integration with other enterprise
applications, batch processing, etc.

Performance, Robustness involving (horizontal/vertical) scalability, load
balancing, high-availability

Data Integration

Vocabulary/Ontology Management – Enterprise Conceptual Models,
  - SKOS
  - RDF(S),
  - OWL

Master Data[1] – data spanning the whole enterprise, like *customers*, *products*, *accounts*, *contracts* and *locations*

---

[1]**source**: http://www.ibm.com/developerworks/data/library/techarticle/dm-0804oberhofer

Integration with other EA

Messaging systems for asynchronous messaging
- JMS (JSR 343)

Remote Procedure Calls for synchronous calls
- RPC
- RMI
- CORBA
- Web Services

Vocabulary/Ontology Management – Is It Worth ?

9/11 – One or Two Events ?

... matter of billions of USD

source:https://www.metabunk.org/larry-silversteins-9-11-insurance.t2375

Performance Testing[2]

Metrics

response time – server-side request processing time,

     latency – client-side request processing time for NOP operation (i.e. for zero response time),

  throughput – transactions per seconds,

   scalability – sensitivity to resource (hardware) addition/removal,

              scaling up (vertical) – add resource (RAM) to one server
              scaling out (horizontal) – add more servers

Contextual Information

         load – number of requests/transactions

load sensitivity – sensitivity of a metric w.r.t load

---

[2]https://nirajrules.wordpress.com/2009/09/17/measuring-performance-response-vs-latency-vs-throughput-vs-load-vs-scalability-vs-stress-vs-robustness

Use Case – External B2C System

e.g. e-shop, social network

- Many Concurrent Users
- Web Client
- Relational Database with a simple model
- Enterprise Data Store Integration

Use Case – Internal Enterprise System
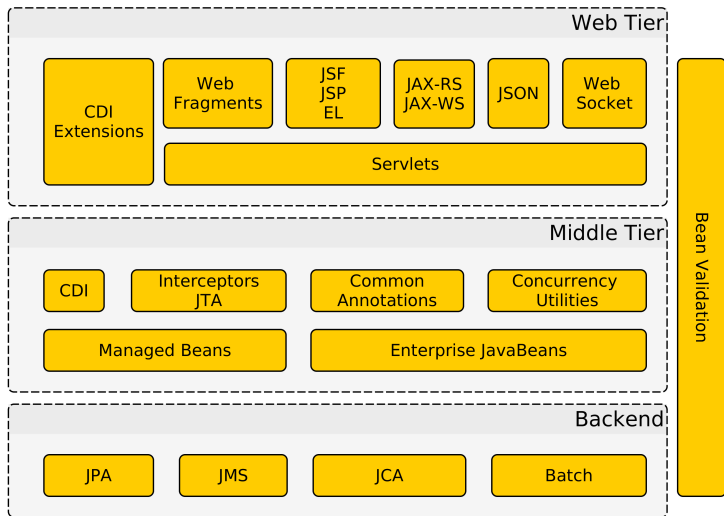e.g. Car Insurance System

- (Not so many) Concurrent Users – mainly company employees
- Thick client for company employees
- Relational Database, complex domain model capturing enterprise know-how
    - e.g. conditions for obtaining an insurance contract
- ERP, CRM Integration

Java EE

Java EE = Java Enterprise Edition

Web Tier

CDI Extensions	Web Fragments	JSF JSP EL	JAX-RS JAX-WS	JSON	Web Socket

Servlets

Middle Tier

CDI	Interceptors JTA	Common Annotations	Concurrency Utilities

Managed Beans	Enterprise JavaBeans

Backend

JPA	JMS	JCA	Batch

Bean Validation

Java EE Principles

- single specification, more implementations
- bunch of technologies integrated in Java EE Application Server

Application servers – Glassfish, WebLogic (Oracle), WebSphere (IBM), WildFly (RedHat), Payara (Glassfish fork), . . .

Web Containers – Tomcat (Apache), Jetty (Eclipse Foundation), . . .

Technologies Used in This Course

JPA (EclipseLink) ($\in$ Java EE stack)

Websockets ($\in$ Java EE stack)

Servlets ($\in$ Java EE stack)

Spring (provides similar functionality as Enterprise Java Beans, CDI, Common Annotations and other)

$\rightarrow$ Apache Tomcat

ReactJS (modern JS-based UI system, more flexible alternative to JSF)

$\rightarrow$ NodeJS (only for efficient compilation)

Alternatives

Spring	vs.	Java EE Session Beans
Spring Web Services	vs.	JAX-RS
EclipseLink	vs.	Hibernate, OpenJPA
ReactJS	vs.	JSF
ReactJS	vs.	AngularJS, ExtJS

Resources

`https://cw.fel.cvut.cz/wiki/courses/ear/materials`