Web Engineering M1-M5

Web Applications : TCP/IP ← HTTP ← HTML / ← XML

...application which is accessed via the web over a network

Categories :
- Document centric : static documents
- Interactive : dynamically generated for a request
- Transactional : read + write, database ; online banking
- Workflow based : workflows between / within companies
- Collaborative : wiki
- Portal oriented : point of access to various informations ; search engine
- Ubiquitous : customized service ; context: location, device, time
- Semantic web : human understanding + automatic manipulation

Levels : Presentation
Hypertext
Content

Traditional software engineering methods : may be inadequate
have to be adapted

HTML

HTML5 : Web Documents -> Web Applications
Form : datalist, keygen, output, new input types
pattern, required, placeholder
Graphics : canvas, svg, picture?
Multimedia : audio, video
Storage : window . localStorage / sessionStorage
Application cache : manifest

Markup: header, nav, section / article, aside, footer

CSS : Inline, Internal, External

Specificity : Inline - IDs - Classes - Elements
Box Model
Positioning
Flex-Box : Container (alignment), Item (grow, order)
Media queries

first class functions

Javascript: ECMA-262, object oriented, classless, dynamic variables
internal head, internal body, external head, (+ inline)

=> Prototypes „global namespace"
Browser Object Model: window, screen, location, alert
Document Object Model: Tree structure for HTML elements

Web Accessibility: W3C Web Accessibility Initiative

Web Content Accessibility Guidelines: (WCAG)
- Perceivable: presentable in different formats
- Operable: ie keyboard navigation
- Understandable:
- Robust: interpretable by different user agents

Accessible Rich Internet Applications: adding Semantics
- Role: input, list, banner, ...
- Property / State: aria-label, aria-checked

-> HTML5 -> default semantics

Responsive Design

Techniques: Media queries, Fluid / Gridbased Layout, Responsive Images, Font Scaling..
Viewport: Visible area of webpage

Layout Patterns: Mostly Fluid
Column Drop
Layout Shifter
Tiny Tweaks
Off Canvas

HTTP: Resource, URL

Synchronous, Stateless

Request: resource, type, application metadata, request metadata, application data
↳ methods: Get, Post, Put, Delete, Patch, Options, Trace, Connect

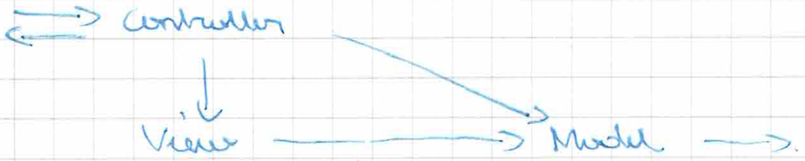
Response: status code, application metadata, response metadata, application data

Servlet: dynamically generating response
<> CGI

-> user, files
-> sessions

JSP: Translation to servlet

MVC: Model: state, data (, application logic)
View: Renders model data.
Controller: Interprets user input, maps input to model



Patterns: reusable solution to recurring problems
has a name
language independent
has consequences

- Architectural
- Design Patterns: Creational
Structural
Behavioral
Concurrency

Architectural: Model View Controller
Result Callback

Creational: Factory Method
Abstract Factory

Structural: Adapter
Facade

Behavioral: Strategy
State (lightweight, heavyweight)