**Web Technologies – IDATA2301 notes from PowerPoints.**

Content

* Course Introduction
  + Web Technology
  + Learning outcomes
  + Course content
  + Project and examination
  + Learning resources
* How the web works – History, architecture, and trends
  + Roles in the web development
  + Web history and trends
  + How the web works
  + Microservices
  + REST and other APIs
* Accessibility, top-down web design
  + Accessibility and Universal Design
  + Different screen sizes
* HTML introduction, text, links, code style

Course Introduction

**Web Technology**

* *The web consists of four fundamental components:*
  + *Website browser*
  + *HTTP protocol*
  + *Web server*
  + *HTML documents*
* *The process between these components ensures the functionality of the web.*

**Learning outcomes**

**Competence**

* *Create a simple website according to a given specification.*
* *Understand and extend code of an existing website.*
* *Design a webpage layout based on a specification.*
* *Implement interaction using JavaScript.*
* *Implement data exchange with a backend system (API)*
* *Implement authentication.*
* *Present their ideas, knowledge, and result.*

**Knowledge**

* *The history of web application development*
* *The web architecture, main functions, and concepts of the different web solution components:*
  + *Web server, HTTP protocol and web browsers*
  + *Languages: HTML, CSS, JavaScript*
* *Event-driven nature of websites*
* *The main features of JavaScript frameworks in general and ReactJS in particular*

**Course content**

* *History and common principles*
* *Universal design and accessibility*
* *Web design guidelines*
* *HTML for structure*
* *CSS for style*
* *JavaScript for interaction*
* *Deployment on a server*
* *Connection to a backend system*
* *Basics of JavaScript frameworks*
* *Basics of ReactJS framework*

**Project and examination**

**Examination**

* *Examination type: Portfolio (100%)*
* *Portfolio contents:*
  + *Source code*
  + *Presentation video*
  + *Git link (git log will be checked)*
  + *Sprint reports*
  + *Link to a deployed site on a public server*
* *Portfolio project: Group project*

**Project**

* *Definition:*
  + *Create a simplified shopping website.*
    - *According to:*
      * *Requirements from a customer*
      * *Design guidelines*
      * *Accessibility rules*
    - *Using best software development practices: Sprints, Git*
    - *JavaScript for interaction and dynamic updates*
    - *Communication with a backend API*
* *Fundamental requirement:*
  + *Functional requirements*
    - *An online shop with landing page, product page, shopping cart, and checkout page*
    - *Application state changes during the lifecycle*
      * *Example: Add product to cart*
    - *DOM modification with JavaScript*
    - *At least one REST API request*
  + *Non-functional requirements*
    - *Responsive design*
    - *Accessibility principles*
    - *Aesthetically pleasing*
    - *Follow a given theme or genre.*
    - *Code under version control*
    - *Agile work methodology in Sprints*
    - *Reasonable code design and structure*
  + *Extra functionality*
    - *Improve the website with more advanced technologies.*
      * *Authentication (front-end side)*
      * *Secured site with HTTPS (When on your own server)*
      * *Automated emails for order confirmation*
      * *Live updates with WebSocket’s*
    - *All extras must be web-related (no extras for statistics)*

**Learning resources**

* [Web design and programming](https://www.udemy.com/course/design-and-develop-a-killer-website-with-html5-and-css3/) by Jonas Schmedtmann
* [Internals of Javascript](https://www.udemy.com/course/understand-javascript/) by Anthony Alicea
* [IDATA2301 - Web Technologies](https://ntnu.blackboard.com/ultra/courses/_34431_1/cl/outline) (BlackBoard)

How the web works – History, architecture, and trends

Roles in web development

* Website creation roles
  + Content
    - Information architecture
      * Organizes content logically for ease of findability and maneuverability.
    - Content strategy
      * Ensures that text supports the brand/marketing goals.
      * May include data modeling and updating schedules as well as extending brand voice to social media.
  + Design
    - User Experience (UX)
      * Makes sure that the whole experience with the website and the product is favorable and based on user testing.
    - Interaction Design (IxD)
      * Focuses on how to use the site, including its User Interface (UI)
    - Virtual Design
      * Creates the “look and feel” of the website.
  + Front-end
    - Authoring/Markup (HTML)
    - Styling (CSS)
    - JavaScript and DOM Scripting
  + Back-end
    - Server software (Apache, Nginx, etc.)
    - Web application languages (Java, PHP, Ruby, .NET, etc)
    - Database software (MySQL, Oracle, MongoDB, etc.)
* Other web roles
  + Product manager
    - Guides the product in a way that meets the business goals.
  + Project manager
    - Coordinates the team, schedules, and manages processes.
  + SEO specialist
    - Ensures that the website is highly ranked in web searches
  + Multimedia producer
    - Creates sound, video, animation, and interactive media.

Web history and trends

Internet vs. Web

* Internet
  + International network of connected computers
* Protocols
  + A standardized method for transferring data or documents over a network (e.g. FTP, SMTP, HTTP)
* Web
  + Information shared over the internet using the Hypertext Transfer Protocol (HTTP), which is one of many ways to share information over the internet.

A Brief History of the Web

* Started in a particle physics lab at CERN in Geneva, Switzerland
* 1989: Tim Berners-Lee proposed a system for sharing documents through “hyperlinks.”
* 1990: Prototypes built for the first time by Tim B-L, later by Robert Cailiau.
* 1992: Approximately 25 servers worldwide.
* 1993: Web opened for commercial usage.

Generations of the Web

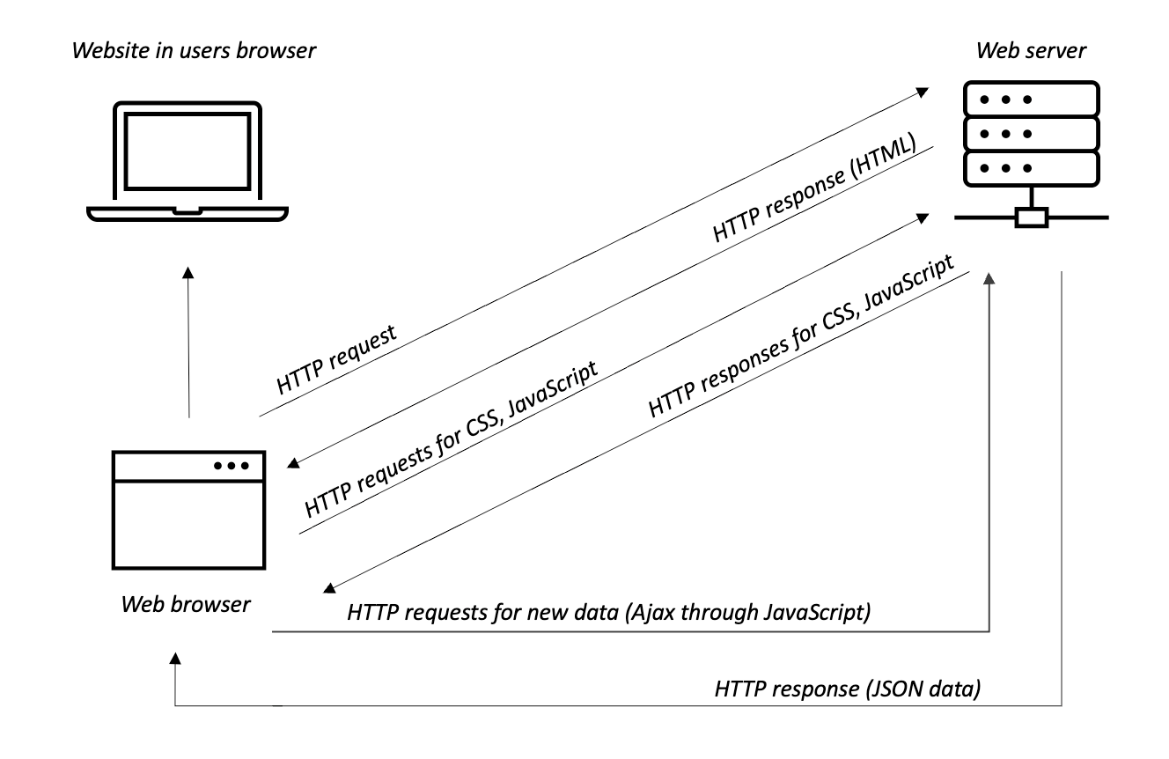
* Generation 1 (1990)
  + Structured and linked information
  + Websites were quite unpersonal and non-interactive
* Generation 2 (1994)
  + Cascading Style Sheets (CSS) were proposed
  + Added more customization possibilities for the appearance of webpages
* Generation 3 (1995)
  + The invention of JavaScript provides the means for building interactive websites
  + It was the first scripting language on Netscape browser
  + JS where developed by multiple institutions, with different versions, supporting different browsers
  + The ECMAScript standard came in 1997

Transmission of web content

* Default: Always load full files
  + Every request returns the whole HTML-file
  + There is no possibility to reload only a single element with the new data.
* Asynchronous requests (Ajax)
  + Microsoft introduced XMLHttpRequest in 1999
  + The idea was to send an HTTP request from a browser to a server, and to parse the HTTP response in JavaScript

Evolution of Ajax

* Update parts of the page
* Single-page applications
* Total separation of responsibility
  + Back-end – Dynamic application, server data, using an API.
  + Front-end – Empty page frame + JavaScript which loads data from an API

**

**Web is an excellent platform.**

* Browsers on different devices and platforms
* All users have a browser, without installation.
* Relatively well standardized
* No one owns web standards.
* Most programmers know the web.

**How the web works**

**The Web Server**

* Server
  + A program that delivers documents and data upon request
* Web server
  + Any computer which is running a web server software

Computer Network

* IP-address
  + A unique number assigned to a device connected to the internet
  + IP – Internet Protocol
  + Example: 199.27.145.64
* Domain Name System (DNS)
  + A system that allows users of the internet to refer to servers by name rather than their IP-address
* Domain Name
  + A name assigned to a web server.
* DNS Server
  + Server that matches domain names to their respective IP-addresses

The Browser

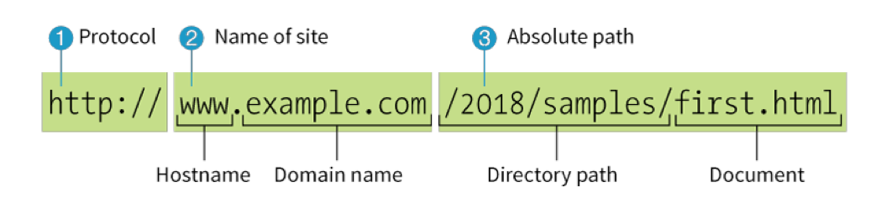
* The software that requests data or documents from the web server
* Also referred to as the client or user agent
* Could be on a desktop machine, smartphone, other connected device, or an assistive device such as a screen reader
* The program in the browser that interprets HTML/CSS/JS is called the rendering engine.

Server-side vs. Client-side

* Indicates which machine is doing the processing
  + Client-side applications run on the user’s machine.
  + Server-side application use the processing power of the server

Web Page Addresses (URLs)

* URL: Uniform Resource Locator
* Every page and resource on the web has its own URL
* Parts of a URL



1. Identifies the protocol as “web” (HTTP)
2. Identifis the site by its domain name
3. Path through directories on the server to the target file

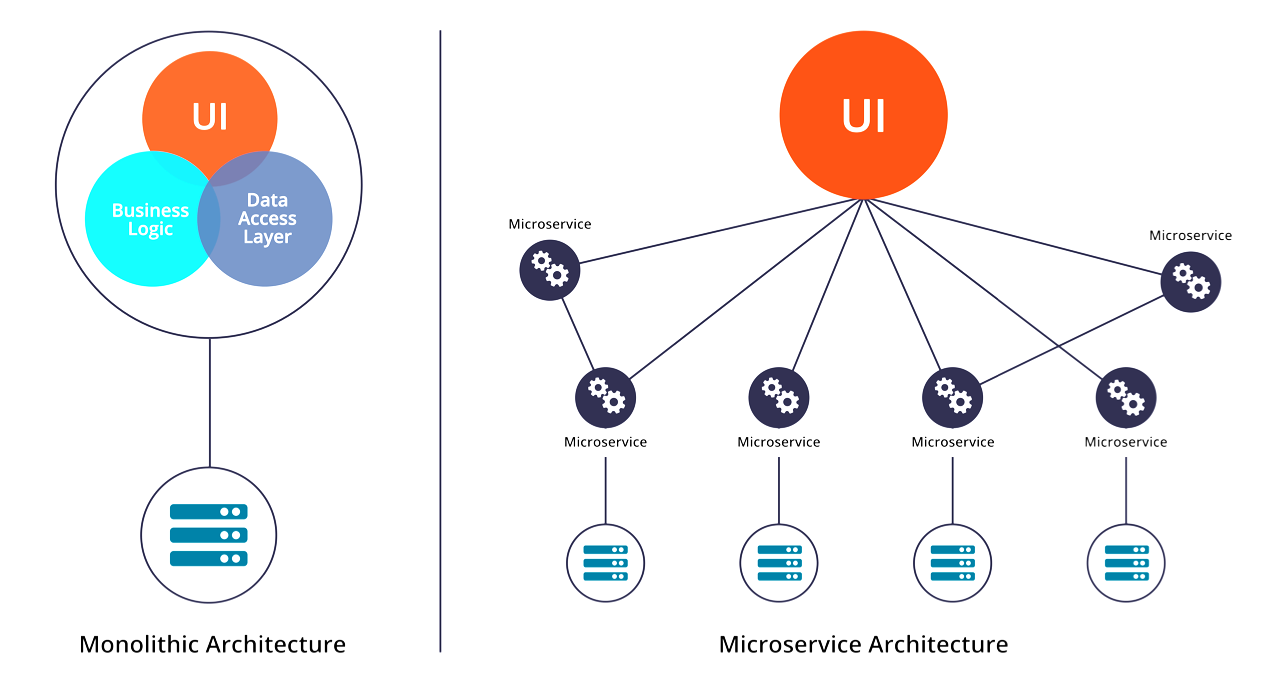
* Simple URLs
  + The protocol is implied and will be added by the browser
  + Domain name is identified
  + If there is no path of filename, it means that the URL is pointing to a default file (usually index.html)

Anatomy of a web page

* The page you see in the browser window is nearly always made up of multiple files, including:
  + An HTML document – Gives the content structure.
  + Style sheets – Describes how it should look.
  + Images and other media – Embedded on the page on the fly.
  + Scripts – Adding behaviors and functionality.

**Microservices**

***Monolith vs. Microservices***

**

**Advantages of Microservices**

* Scalability
* Easier to deploy updates.
* Other platforms (mobile, desktop) can re-use the same back-end system
* Machine-to-Machine services
* Split complexity into smaller pieces
* Distribute work among team members.

**Disadvantages of Microservices**

* Complexity grows.
* More work on the user’s device
* More technology to learn and maintain.

**Rest and other API’s**

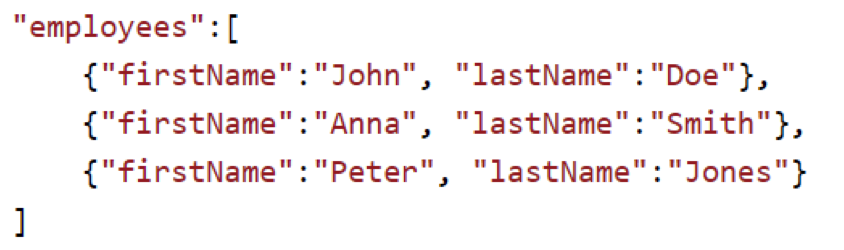
HTTP – Well established transport

* Simple
* Web uses HTTP anyway.
* Supported everywhere.
* Not blocked by firewalls
* Disadvantage: Not efficient

HTTP for data exchange

* Idea: Use HTTP to transport data between services
  + Call them Web Services, because HTTP = Web
* Using the JSON format to encode data objects.

JSON – Simple data format

**

REST API

* In simple terms:
  + Exchanging data between services
  + Using HTTP as the transport
  + Using JSON as serialization (marshalling)

Alternative to REST

* XML data over HTTP
* WebSockets – Bi-directorial asynchronous data exchange (The server can push data to the client)
* Messaging protocols: MQTT, Kafka, ZeroMQ, etc.

Summary

* Multiple roles in web development
* Web has evolved from static pages to dynamic applications.
* Web pages are stored on web servers.
* Recent trend – Microservices – Separate applications
* Most popular microservice communication: REST API

Accessibility, top-down web design

What is Accessibility? What is Universal Design?

* Accessibility – make interface accessible for people with disability or special needs.
* Universal design – make interface easily accessible for everyone.
* Better accessibility = better for everyone except developers.

Why do we need this concepts in programming?

* 2.2 Billion people worldwide have poor vision
* 8% of men are color blind (0.5% of women) = 640 million!
* 295 million have severe vision impairment.
* 43 million are blind.
* 7.3 million americans use screen readers.
* 5-10% of population have some form of dyslexia.

What does universal design include?

Different principles. Simplified:

* Equitable use – accessible for everyone.
* Simple, intuitive use.
* Tolerance of error.

Accessibility for web

For visitors with poor sight:

* Good contrast
* Color-blind safe
* Large items
* Ability to zoom in

For blind people – screen-reader friendly:

* Text explanation for media
* Logical structure and sequence in HTML
* Extra meta-data

Guidelines and laws

* W3C Web Content Accessibility Guidelines (WCAG)
  + Version 2.0 – current (newest: 2.1)
* Europe: EN 301549 - Accessibility requirements for ICT products and services.
* Each country issues different laws
  + Norway: websites MUST comply with WCAG 2.0!

WCAG

* 4 principles -> 12 guidelines -> 61 testable criteria
* Principles:
  + Perceivable – information presented in a way that is easy to perceive.
  + Operable – navigation must be operable.
  + Understandable – understandable information and operation of user interface
  + Robust – website works in different user agents, including assistive technologies.
* ARIA – Accessible Rich Internet Applications
  + Standards for making content accessible for different user agents, including screen readers.
  + Extra tools for web developers – describe semantics and meta information.

Web for different screen sizes

A multitude of Devices

* Your web pages will be viewed on all manner of devices, large and small, fast and slow, visual and non-visual.
* One of the challenges of being a web designer is creating a good experience regardless of the browsing device.

Web Standards

* The World Wide Web Consortium (W3C) writes the specifications for web technologies.
* Sticking with web standards ensures consistency across browsers and forward-compatibility.

Progressive Enhancement

Progressive Enhancement is a strategy for coping with unknown browser capabilities.

* Start with baseline experience that provides content and basic functionality even on minimal browsers and assistive devices.
* Layer on styles, scripts, and advanced features for browsers that can handle them.
* Finish with “nice to have” effects (like animations) that aren’t critical to the brand or functionality.

Responsive web design

Responsive web design is a strategy for dealing with unknown screen size:

* The heart of the method is using one HTML source for all devices and swapping out the styles based on the size of the browser window (viewport).
* It is preferred to building separate sites just for mobile devices (“m.dot” sites).
* It may not be the solution for all sites, but making sites that adapt to screen size is now common practice.

Top-down approach for web development

What is top-down for web?

* Start with understanding the need:
  + What does the owner/customer want?
  + What does the user want?
  + What is the general theme of the site/company?
* What is the most important message or product?
  + What will the “call to action” button/link lead to?
* Layout
  + In what sequence do we want to lay out content?
  + What is typical for other sites?
* What components do we need on the site?
  + What elements do we need to use? (HTML)
  + How can we style the elements? (CSS)

Fanatic VS professional developer

* Fanatic programmer
  + Hear first sentence of customer.
  + -> code -> hacking -> more code
  + -> customer not satisfies -> customer is dumb
* Professional programmer:
  + Clarify need -> sketch -> design layout -> discuss with customer
  + Identify components.
  + Code WHICH SOLVES A NEED
  + Satisfied customer.

Visual hierarchy

* Top to bottom – most important things in top
* Images draw a lot of attention.
  + Especially faces.
* Use font size and font weight to create hierarchy.
* Increase importance with a highlight-color (also background color)
* Use whitespace to separate and emphasize.