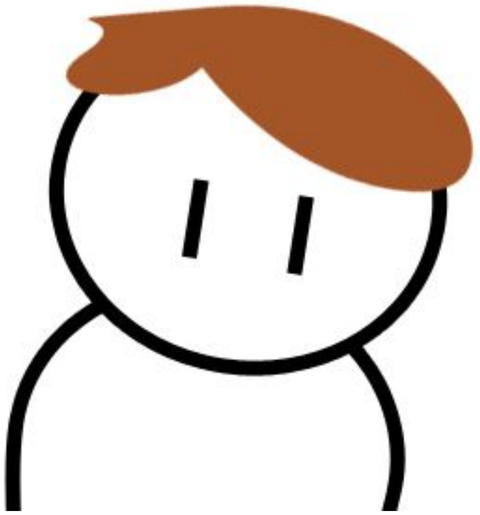
HÖGSKOLAN I SKÖVDE

WEBBPROGRAMMERING

Kursintroduktion
Arbetsstrategi

PEOPLE

- Henrik Gustavsson (Main Teacher)
- Simon Butler (Examiner)



COURSE SITE / CANVAS

All course information will be available on LenaSYS, which you can reach via Canvas or directly via https://dugga.iit.his.se

COURSE OBJECTIVES

- Explain concepts related to languages and technologies used for client-side dynamic web page programming
- Describe and compare the characteristics of languages and technologies for programming dynamic web pages
- Describe and compare the characteristics of languages and technologies for programming embedded content in dynamic web pages
- Independently program dynamic web pages, using different languages and technologies.
- Using different types of web programming toolkits.

EXAMINATION MOMENTS

Home exam 2.5 credits U/G/VG

Assignments Scripting 2 credits U/G/VG

Assignments embedded content 2 credits U/G/VG

Seminar assignment 1 credit U/G

EXAMINATION Content

Assignments 2+2 credits

Programming task - Examining practical course objectives (U/G/VG)

Report

- Must contain the most important parts of the program code including screenshots of the application in a browser
- Discussion around justifications, design choices and possible alternative solutions
- What is built from scratch with JavaScript and what is built with WEB API
- Discussion that reflects on, for example, difficulties, lessons learned and parts of your solution that you are particularly satisfied with

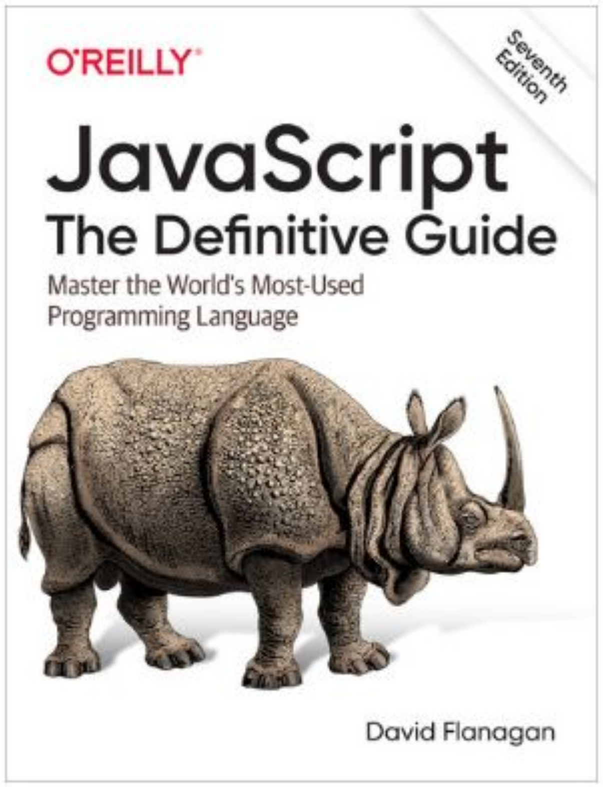
Seminar – Compulsory 1 credit (U/G)

Reporting and demonstration of the assignment (approx. 10 min) including discussion with examining student.

Home exam 2.5 Credits (U/G/VG)

BOOK

- Flanagan, David (2020). *JavaScript: The Definitive Guide* (7th edition): O'Reilly Media.



LECTURES

The course is conducted in laboratory form

- Lectures will be introductory only
- You are expected to acquire the necessary knowledge yourself
- Course literature
- Code example
- Tutorials
- Other sources, e.g. Stack Overflow, etc.
- Lectures

Introduction / work strategy

JavaScript

WEBAPI / Booking API

HTML5

Introductory lectures can also occur during supervision sessions

TUTORIALS

A number of tutorials / lectures

- The lectures can be moved around if necessary, if many are ready in advance, the lecture can be brought forward
- Use the time well!
- More time than has been scheduled will be required to complete the course

Many people take the course

- Read the instructions for the submission task
- Read in the course literature
- Look at the code examples
- Always try for yourself before asking

COME TO THE SUPERVISED SESSIONS

- A large percentage of those who do not pass the course have not attended the supervised sessions
- Introductory lectures can also occur during supervision sessions

GRADING CRITERIA

Layout/aesthetics are of minor importance and do not affect the grade, but only the criteria below determine the grade for the course element:

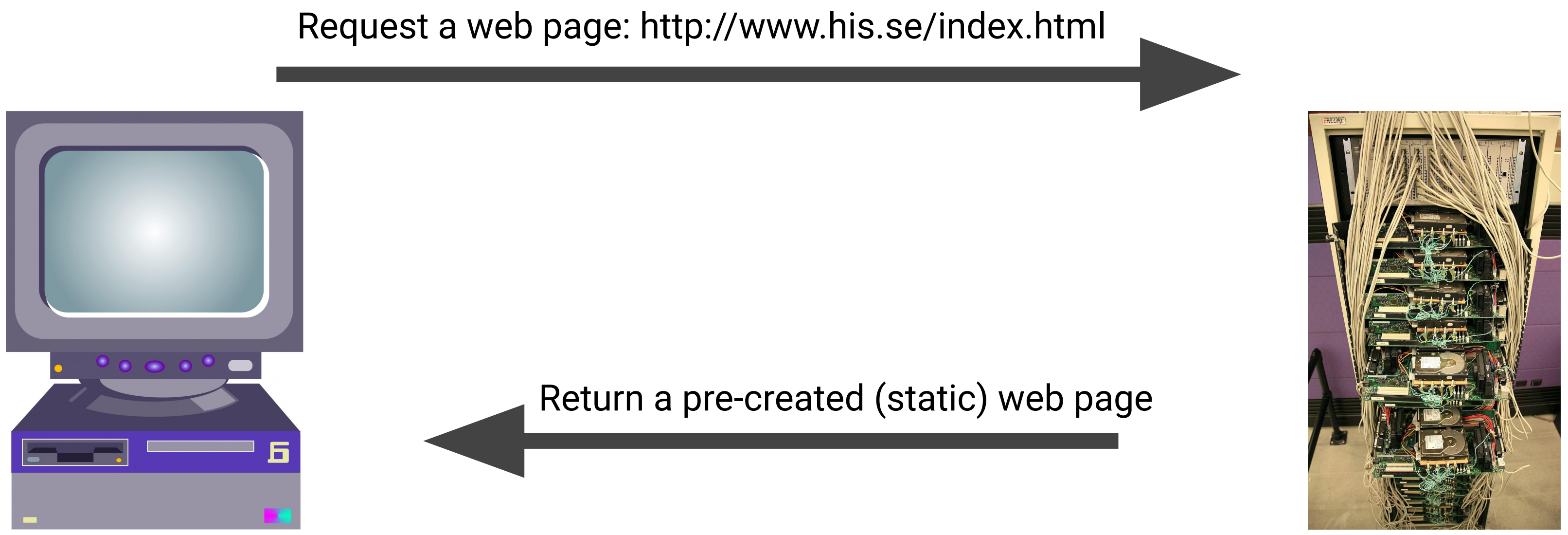
On LenaSYS you can see, for each task:

A list of requirements required for G

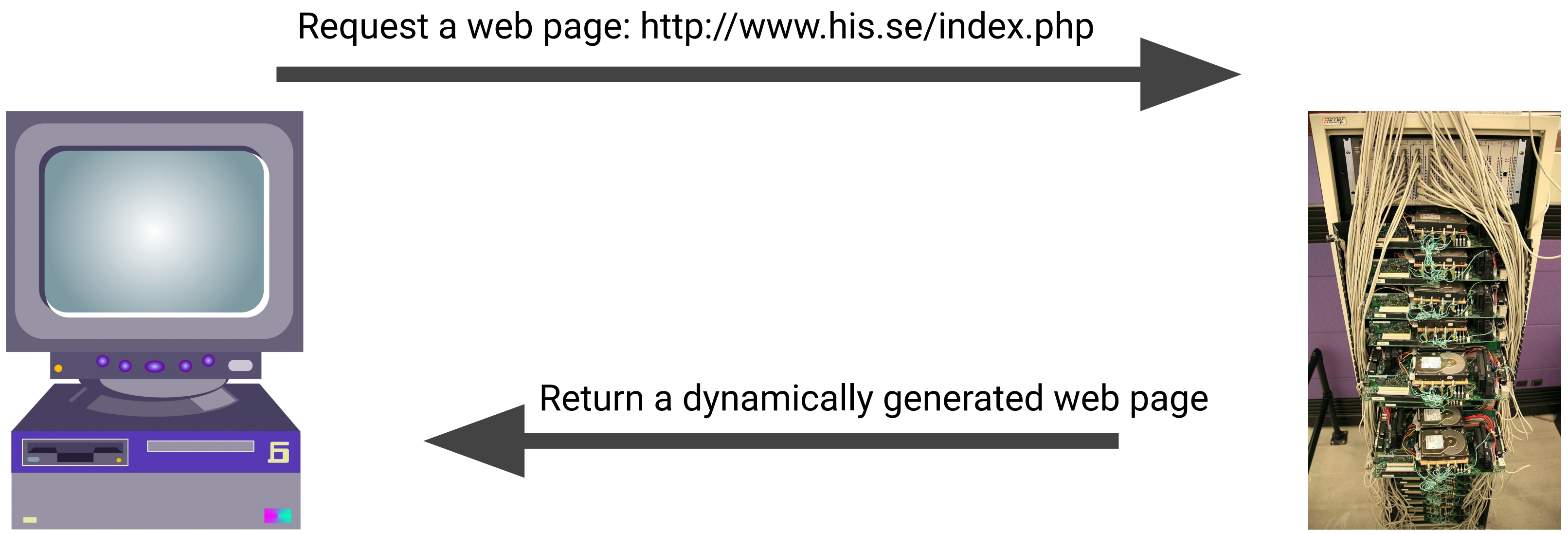
A list of requirements required for VG

You need to achieve a VG on the programming tasks as well as the written exam

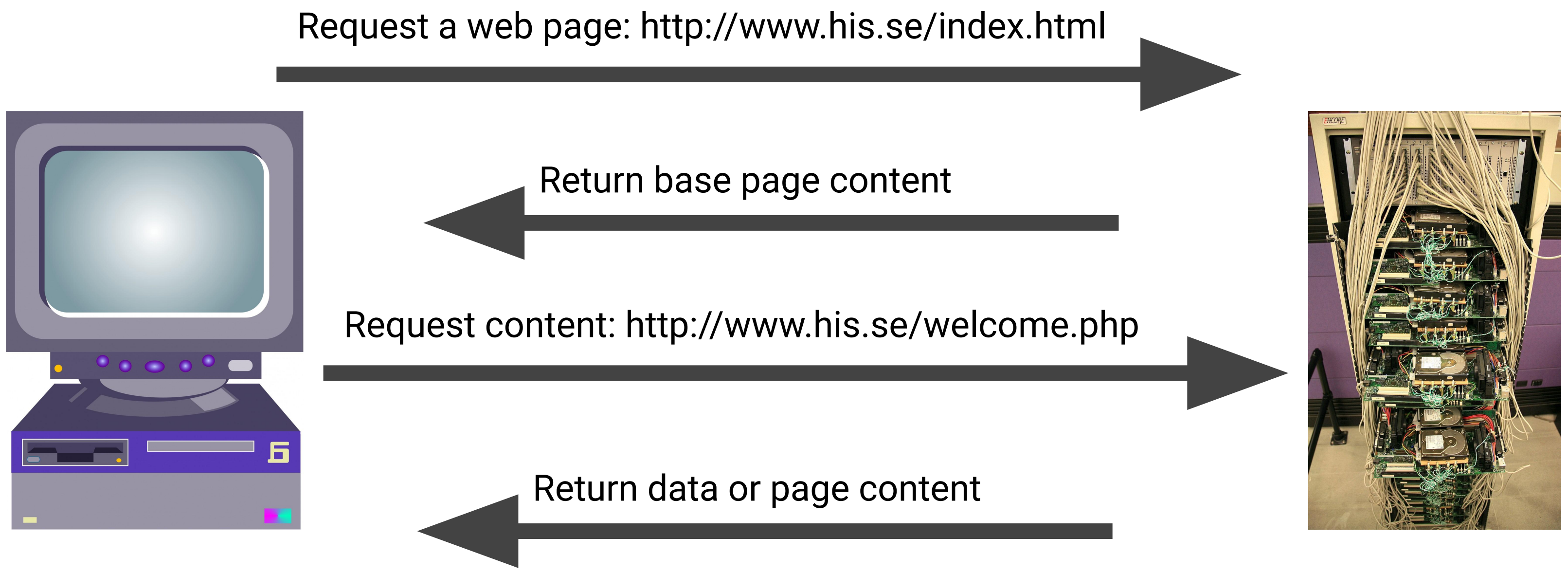
Web pages really old style (1990s)


Request a web page: http://www.his.se/index.html
Return a pre-created (static) web page

Web pages server-side generated


Request a web page: http://www.his.se/index.php
Return a dynamically generated web page

Web pages updated using Javascript (2003-)


Request a web page: http://www.his.se/index.html
Return base page content
Request content: http://www.his.se/welcome.php
Return data or page content

Event-based Programming

Event-based code execution means that we run code when something specific (an event) occurs.

Examples of events are: "Key is pressed", "mouse pointer moves", "button is clicked", etc

Code that runs when an Event has occurred is called an EventListner

The same process happens in android and in ios, which are also seen as event-based programming platforms.

Statically defined EventListner

We can define a static EventListner directly in the HTML document.

Every time we click in the DIV, a click event will be generated and our code in the function myHandler will be executed.

Note that we send with the generated click event. We do this because there is a lot of useful data in events.

```
<!DOCTYPE html>
<html lang="en">
<head>
    ...
    <script>
        function myHandler(e){
            alert("Event just fired!");
            console.log(e);
        }
    </script>
</head>
<body>
    <div onclick="myHandler(event);">Klicka</div>
</body>
</html>
```

Define EventHandler dynamically

We can declare Event Handlers dynamically in javascript.

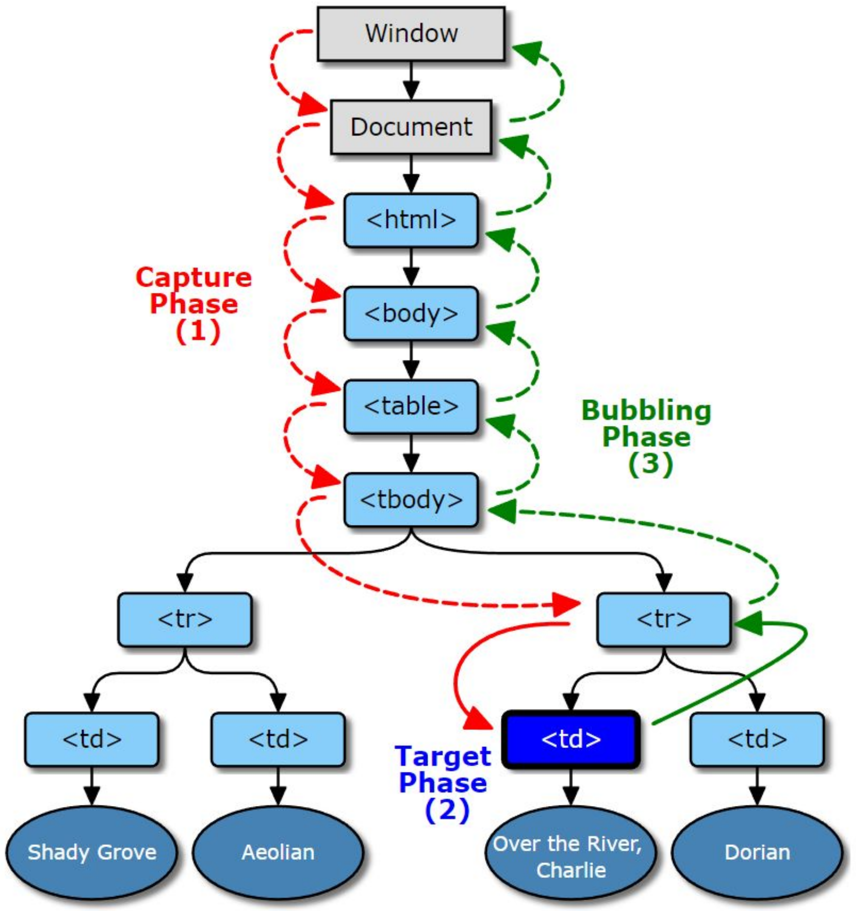
In this case we use querySleector which returns first matching element in document

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Dynamic event</title>
</head>
<body>
  <div class="div">Klicka</div>

  <script>
      function myHandler(e){
          alert("Event just fired!");
          console.log(e);
      }
      // QuerySelector returns first matching element
      document.querySelector(".div").addEventListener('click', myHandler)
  </script>
</body>
</html>
```

Event phases

Normally only “target phase” and “bubbling phase” are used



From https://www.w3.org/TR/DOM-Level-3-Events/

Javascript APIs

There are many APIs (Application Programming Interfaces) that you can use when programming in javascript.

For a good overview of available APIs:
https://developer.mozilla.org/en-US/docs/Web/API

Assignment

- Build your own booking system!
- Using:
- JavaScript
- Web Services
- Frameworks

- The focus is on client-side programming
- We provide a database and an API for bookings
- You must use our database and API for booking

- You freely decide what your booking system will be used for

Single Page Layout

- Single-page-application layout (SPA) must be applied.
- This means that you depart from the working method from previous courses where each new web page is its own html document. Instead, you have a single html document that changes appearance depending on which "logical" page you are on.
- We will give the impression that our web application has several "pages", such as welcome page, profile page, booking page, etc.
- This single html document loads information from the API when needed without doing a reread of the html document.

There are many ways to make SPA applications

When making a SPA, you need to consider which style is most suitable. When we don't have to be limited to just links and navigation between pages, we have significantly greater opportunities for interaction.

A page that looks exactly like a "regular page" that navigates between pages

Page that scrolls up and down in a large document

Page with dropdowns

Page with dialog boxes that appear when needed

Page that scrolls left and right like a carousel

...and more...

We can also combine, for example, some regular pages, combined with some dialogs.

One Page Layout (kod från SPA template)

```
function showpage(pageid)
{
    var pages = document.getElementsByClassName("page");
    for(page of pages)
    {
       page.style.display="none";
    }
    document.getElementById(pageid).style.display="block";
}
```

•••

```
<div id="menu">
    <button onclick="showpage('page1');">Page1</button>
    <button onclick="showpage('page2');">Page2</button>
    <button onclick="showpage('page3');">Page3</button>
</div>
```

•••

```
<div id="page1" class="page">
    <h1>PAGE 1</h1>
</div>
```

Work Process for Assignment

I. Plan
1. Content
2. Structure
3. Interaction

- Sketch
- Notes
- HTML

II. Implementation
1. Structure
2. Validation
3. Connect interface to API
4. Interaction
5. Local Storage History Management
6. Polish
7. Graphics
8. Graphics Interaction

- Javascript
- Skapa grafik

PLANNING 1. CONTENTS

Plan your work and try to create a clear idea of what you want to do before you start programming

What will be built?

What type of resource should it be possible to book?

What requirements are placed on the booking system based on what can be booked?

Place, date, time, number, weight, etc.

Which functions should be present?

What should a user of the system do to complete a booking?

To be done through interaction with graphic representation of the resource

Map your thoughts to the task description!

Where should HTML5 Web Storage or cookies be used?

Which logical pages should your web application have?

How to apply frameworks for interaction

PLANNING 2. STRUCTURE

One or more virtual pages?

An html document that is updated with API

A "logical" page for each function of the web application?

Plan so that all pages are connected

- Welcome page/login
- Customer registration page
- Search page
- Booking page
- Shopping cart/booking history
- Other pages?

PLANNING 3.INTERACTION

Static content

- What should the website's static content look like?

Dynamic content

- Which parts of the website should have dynamic content?
- Ex: Which parts should the user be able to interact with?

Graphical content (HTML5)

- Where and for what?
- Should be part of the user interface

User interface

- How should the interaction take place?
- What forms and menus are needed?
- How should it go practically when the user books a resource?
- What should toolkits be used for?
- Ex: Forms, interface functionality, etc.

IMPLEMENTATION 1.STRUCTURE

Create static content

- Basic HTML and CSS code for the website

Implement dynamic content

- Implement navigation design
  - Links
  - Tabs
  - Menus
  - Etc.
- Create forms
  - Buttons
  - Text boxes
  - Radio buttons
  - List boxes
  - Date pickers
  - Etc.

IMPLEMENTATION 2.VALIDATION

Implement validation functionality for the website's various interfaces

- Validation of entered data in forms
    - Check if the field is empty
    - Check if the correct type of data has been entered
    - Ex: email address
    - Etc.
- Validation of the interaction of the graphical part of the interface
    - Check if incorrect interaction occurs
    - Ex: More bookings are allowed

IMPLEMENTATION 3.CONNECT INTERFACE TO API

- Implement the code that creates users (Registration of new users)
- Implement functionality to log in
- Validate that only Registered users are able to log in
- Retrieve and list available resources from the database
- Show which resources can be booked
- Implement the code that executes a booking in the database
- Only Logged in user are able to book a resource
- Retrieve and display which resources a user has booked
- Logged in user must be able to see what she/he has booked
- Implement a search function
    - It must be possible to search among available resources
    - Ex: Which trains depart from Örebro after 14:30 on 25 December 2012?

IMPLEMENTATION 4.INTERACTION

I. Create graphic elements (HTML5)
II. Implement the code that draws the graphics
III. We have an SVG to canvas tool that you can use
IV. Build the user interface
V. Implement the code that allows the user to interact with graphical elements and other parts of the website
VI. Implement the parts that use toolkits
VII. Ex: replace date input form elements with date pickers

IMPLEMENTATION 5. HISTORY MANAGEMENT

- What happens if the user:
  - Pressing the back or forward button?
  - Reloading the page?
  - Creating a bookmark?
- We have to handle this ourselves when using AJAX
  - Hash strings
  - WEB API History API
  - Either or:
  - Hash strings
  - Implement history management, including of hash strings
  - History API
  - Implement history management, including The History API

IMPLEMENTATION 6.FINISHING

- Appearance
  - Colors
  - Backgrounds
  - Gradients
- Extra animations
  - Animation of the interface with CSS/JavaScript/toolkits
  - Both to improve the appearance but also to facilitate interaction
- Extra functionality
  - Improve existing features
  - Ex: If it is only possible to book one resource at a time, it makes it possible to book several at the same time
- Adding additional features
  - Ex: Cancel resources

TODAY

- Read through the instructions for the submission task
- Decide which resource you will build your booking system for
- Start planning your work according to the proposed strategy
- The planning phase needs to be complete before the next session
- Download and familiarize yourself with the SPA template

  https://github.com/LenaSYS/Webprogrammering-Examples

- Read on and start building your page in HTML and CSS
- Read the course literature about JavaScript