LINKÖPING UNIVERSITY

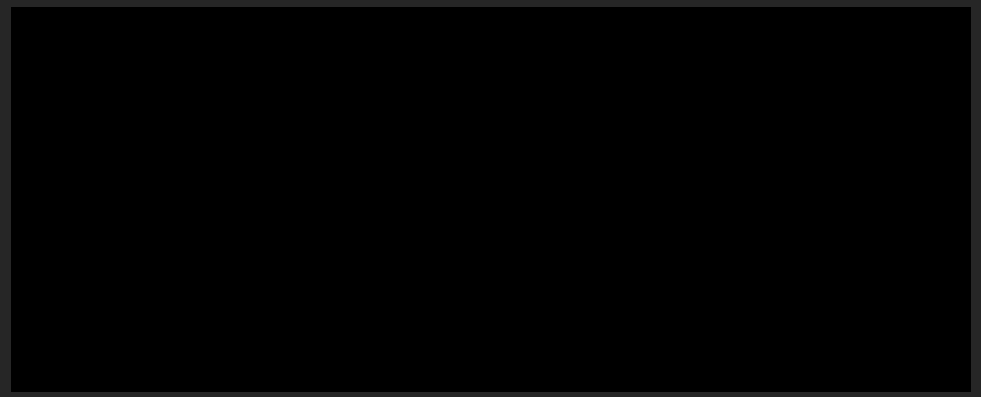
729G87 Interaction Programming

Lecture 4 – Advanced concepts

Philipp Hock, PhD
philipp.hock@liu.se

Transitions

https://www.w3schools.com/css/tryit.asp?filename=trycss3_transition1



```
<style>
div {
  width: 100px;
  height: 100px;
  background: red;
  transition: width 2s;
}

div:hover {
  width: 300px;
}
</style>
</head>
<body>

<h1>The transition Property</h1>

<p>Hover over the div element below, to see the
transition effect:</p>
<div></div>
```

CSS Animations

https://www.w3schools.com/css/css3_animations.asp

```
 /* The animation code */
@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}

/* The element to apply the animation to */
div {
  width: 100px;
  height: 100px;
  position: relative;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```

https://codepen.io/Philipp-Hock/pen/KKbKmaB

CSS Animations - Easing

```
#div1 {
  animation-timing-function: linear;
  animation-timing-function: ease;
  animation-timing-function: ease-in;
  animation-timing-function: ease-out;
  animation-timing-function: ease-in-out;
}
```

Animation shorthand

https://www.w3schools.com/css/css3_animations.asp

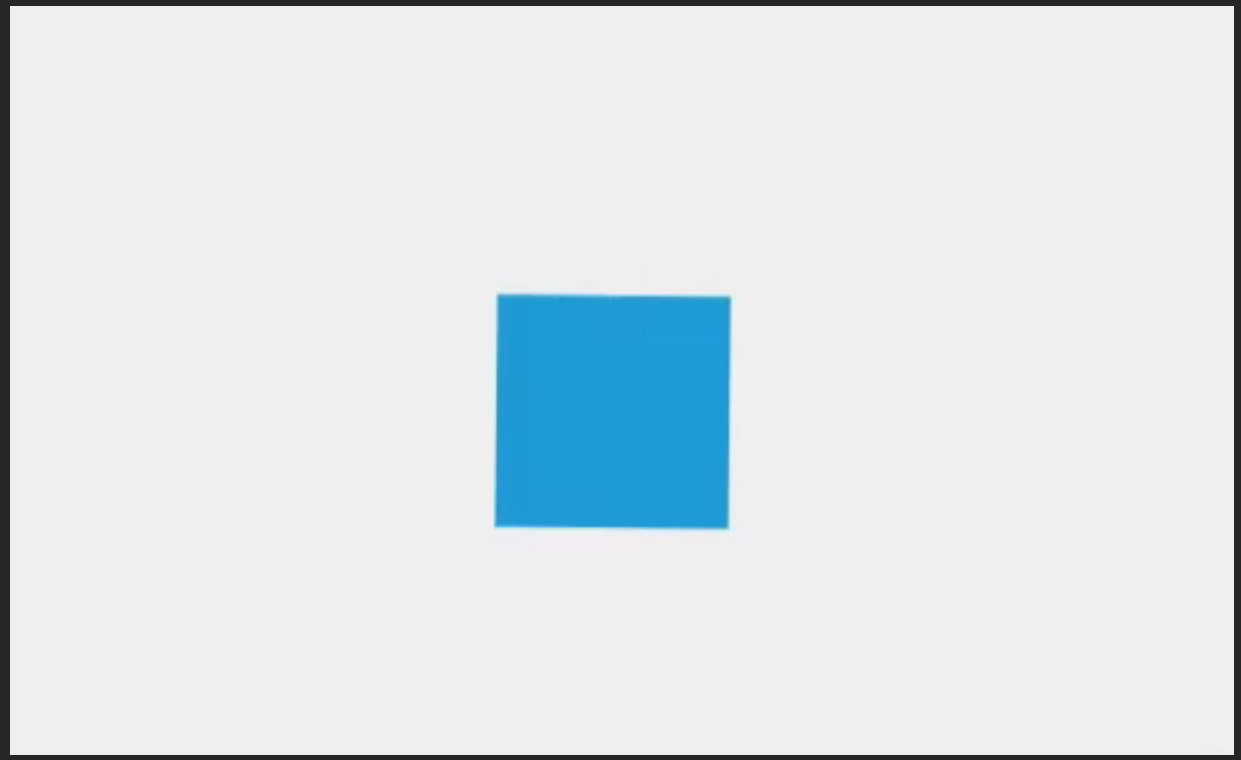
```
div {
  animation-name: example;
  animation-duration: 5s;
  animation-timing-function: linear;
  animation-delay: 2s;
  animation-iteration-count: infinite;
  animation-direction: alternate;
}
```

Equals:

```
div {
  animation: example 5s linear 2s infinite alternate;
}
```

More complex animation

https://codepen.io/Philipp-Hock/pen/JjwjNJo



Revisit some code

- DRY: Don’t repeat yourself
- Select a list, select items from a list
- Style elements with toggle classes

Revisit some code

- DRY: Don’t repeat yourself

```
function inc(){
  index++;
  document.querySelector("img").src="...";
}

function dec(){
  index--;
  document.querySelector("img").src="...";
}
```

Revisit some code

- DRY: Don’t repeat yourself

```
function update(i){
  index+=i;
  document.querySelector("img").src="...";
}

update(1);
update(-1);
```

Revisit some code

- Select a list, select items from a list

```
<div class="images">
  <p id="p1">hello</p>
  <p id="p2">world</p>
  <p id="p3">and </p>
  <p id="p4">beyond</p>
</div>

<script>
  const p = document.querySelector("#p2");
  console.log(p)
</script>
```

Revisit some code

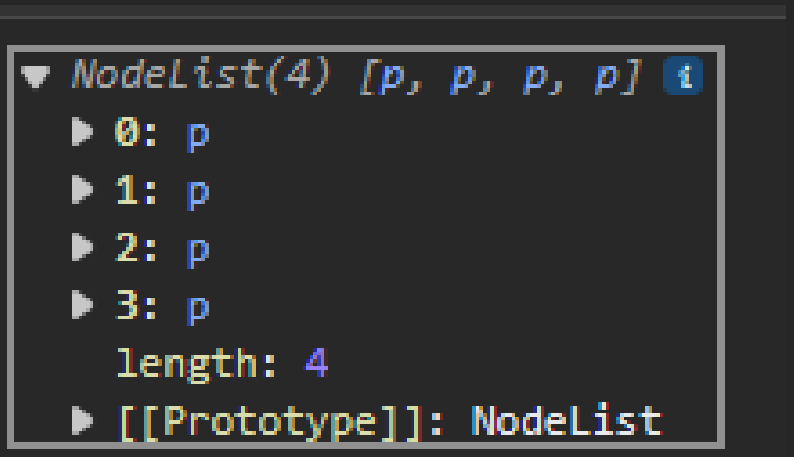
- Select a list, select items from a list

```
<div class="images">
  <p>hello</p>
  <p>world</p>
  <p>and </p>
  <p>beyond</p>
</div>

<script>
  const p = document.querySelector(".images p:nth-child(2)");
  console.log(p)
</script>
```

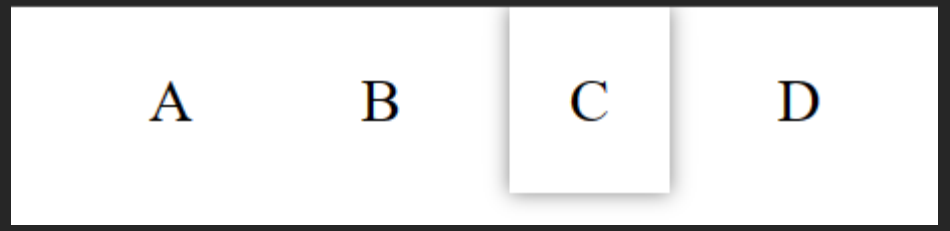
Revisit some code

- Select a list, select items from a list

```
<div class="images">
  <p>hello</p>
  <p>world</p>
  <p>and </p>
  <p>beyond</p>
</div>

<script>
  const p = document.querySelectorAll(".images p");

  console.log(p[1])
</script>
```



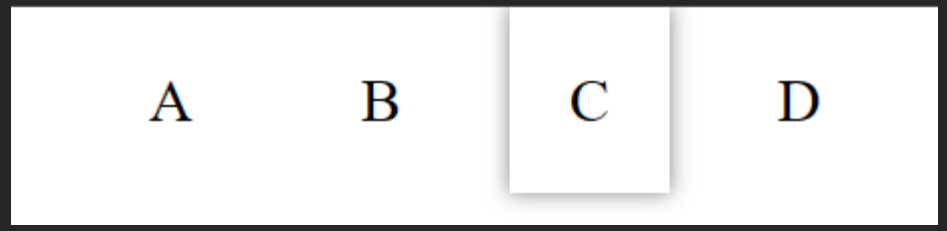
Revisit some code



- Style elements with toggle classes

```
<div class="images">
 <span>A</span>
 <span>B</span>
 <span>C</span>
 <span>D</span>
</div>

<script>
  const p = document.querySelectorAll(".images span");
  p[2].style.boxShadow = '0 0 10px rgba(0, 0, 0, 0.5)'; // Box shadow

</script>
```

Revisit some code



- Style elements with toggle classes

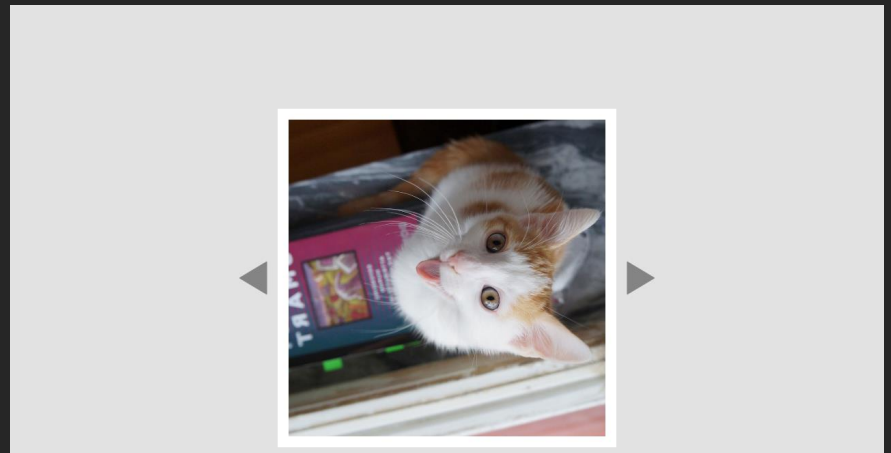
```
<div class="images">
 <span>A</span>
 <span>B</span>
 <span>C</span>
 <span>D</span>
</div>

<script>
  const p = document.querySelectorAll(".images span");
  p[2].classList.add("active");

</script>
```

In your css file

```
.active{
  box-shadow: 0 0 10px rgba(0, 0, 0, 0.5);
}
```

LINKÖPING UNIVERSITY

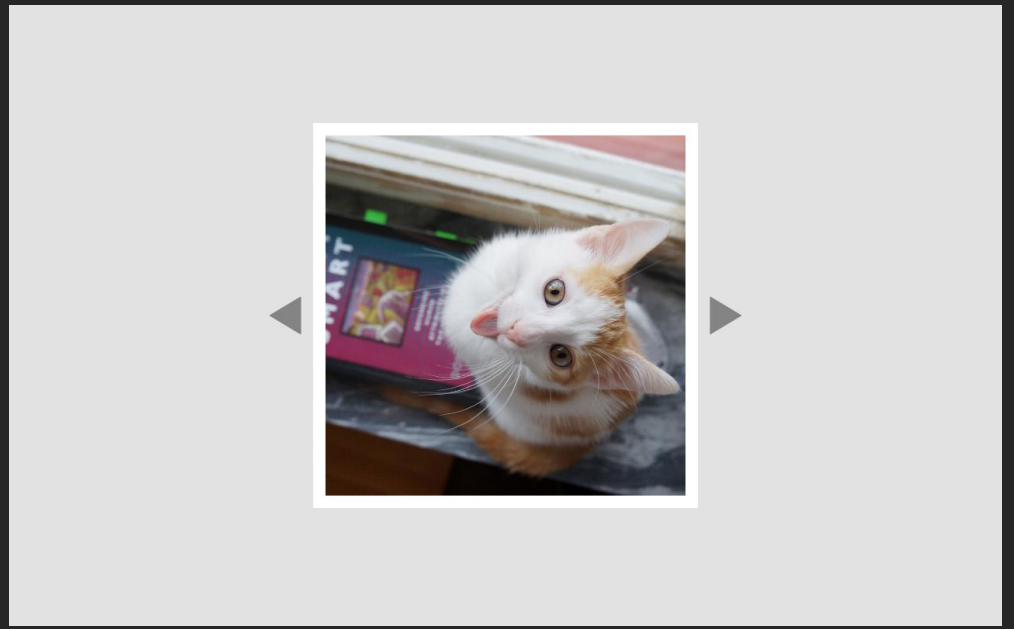
From Concept to Code



From Concept to Code

Functional description

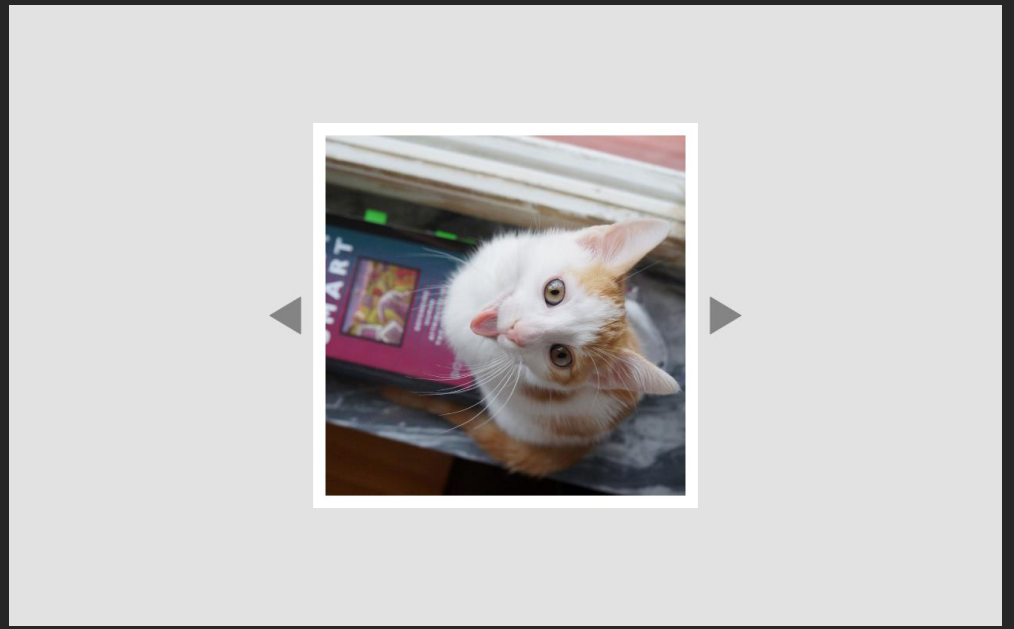
- On button press (->): show next image
- On button press (<-): show prev. image
- Cover edge cases: no next/prev image



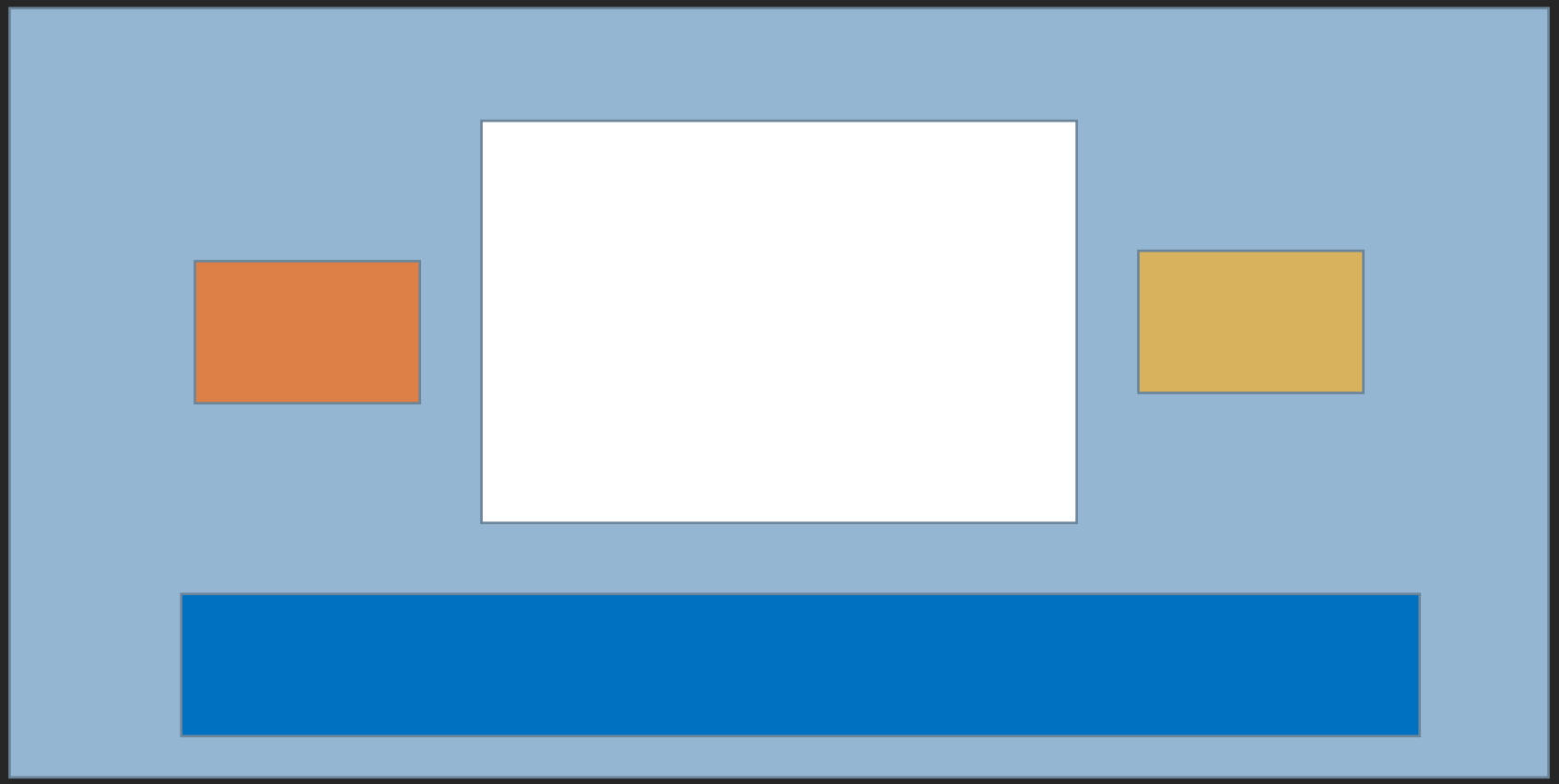
From Concept to Code

Functional description

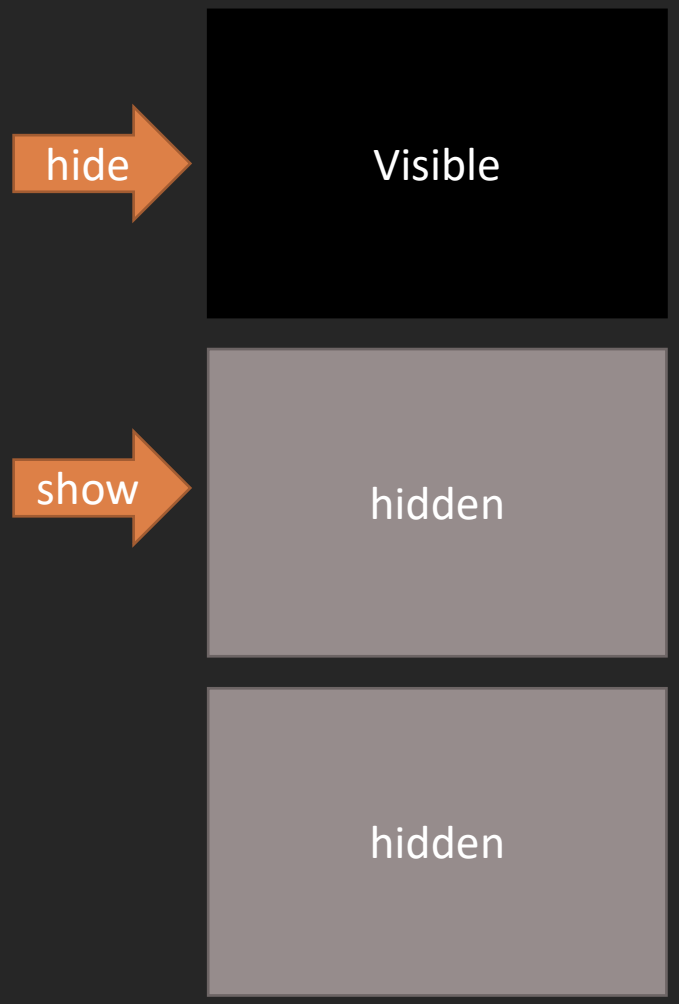
- On button press (->): show next image
- On button press (<-): show prev. image
- Cover edge cases: no next/prev image



Break down components



Think about a strategy



Break down logic

Show cat image

- Executable operations
- Events
- Queries
- Requests
- Styles
- …

Break down logic

What does it mean to show an image for the computer?

Show cat image

- Html element
  - visible
  - src tag points to url

Break down logic

Show cat image

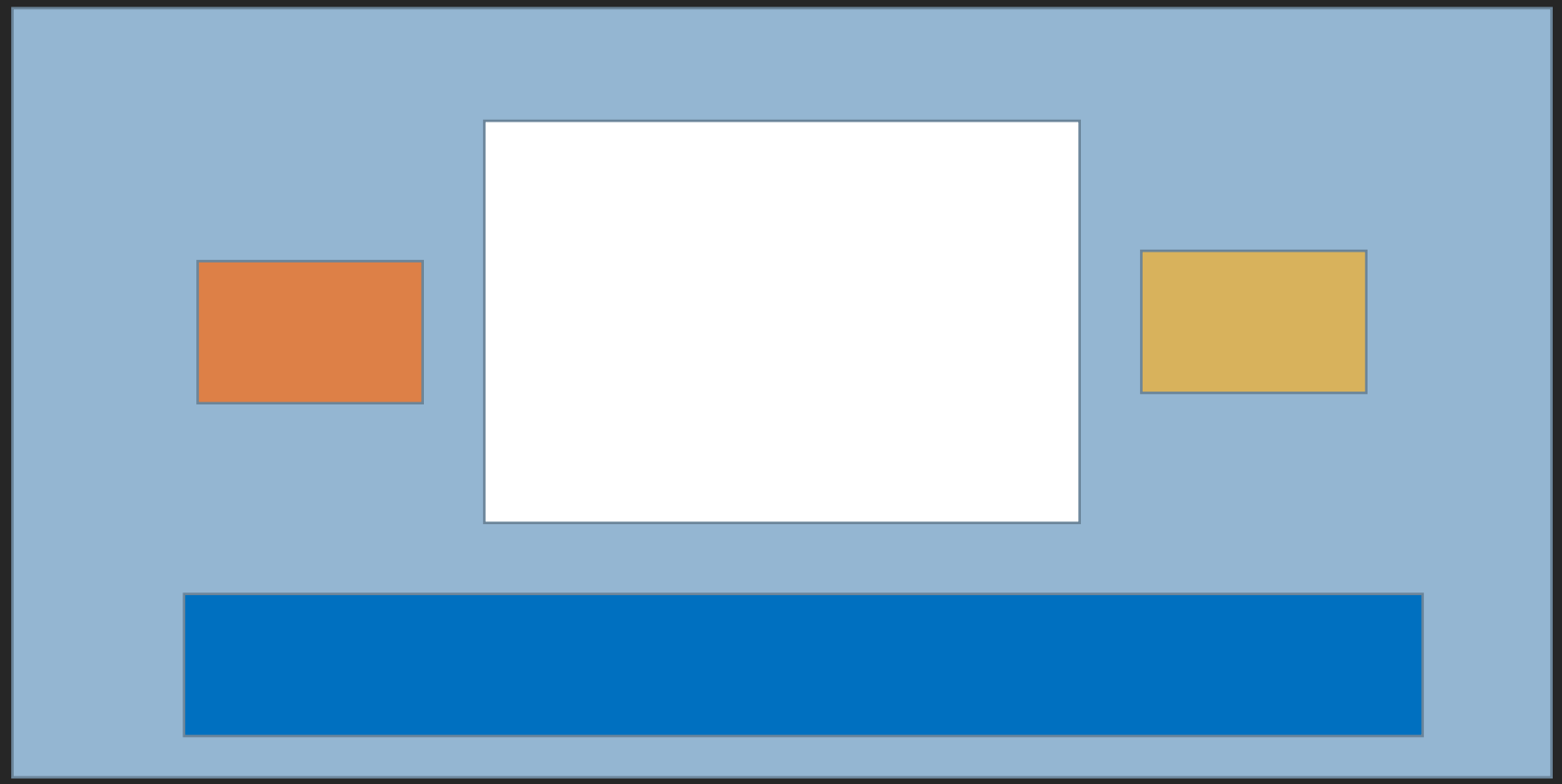
How to achieve this?

- Get html element into js
- Set the src attribute

Or

- Get all html elements into js
- Show current element
- Hide other elements

Break down components



Break down components

what

how

Button press

- get button element
- register event
- execute (show next)

querySelector

addEventListener

Break down into operations

On button press (->): show next image

what

Next image:
- list of img-tags
  - [“<img>”, “<img>”,…]

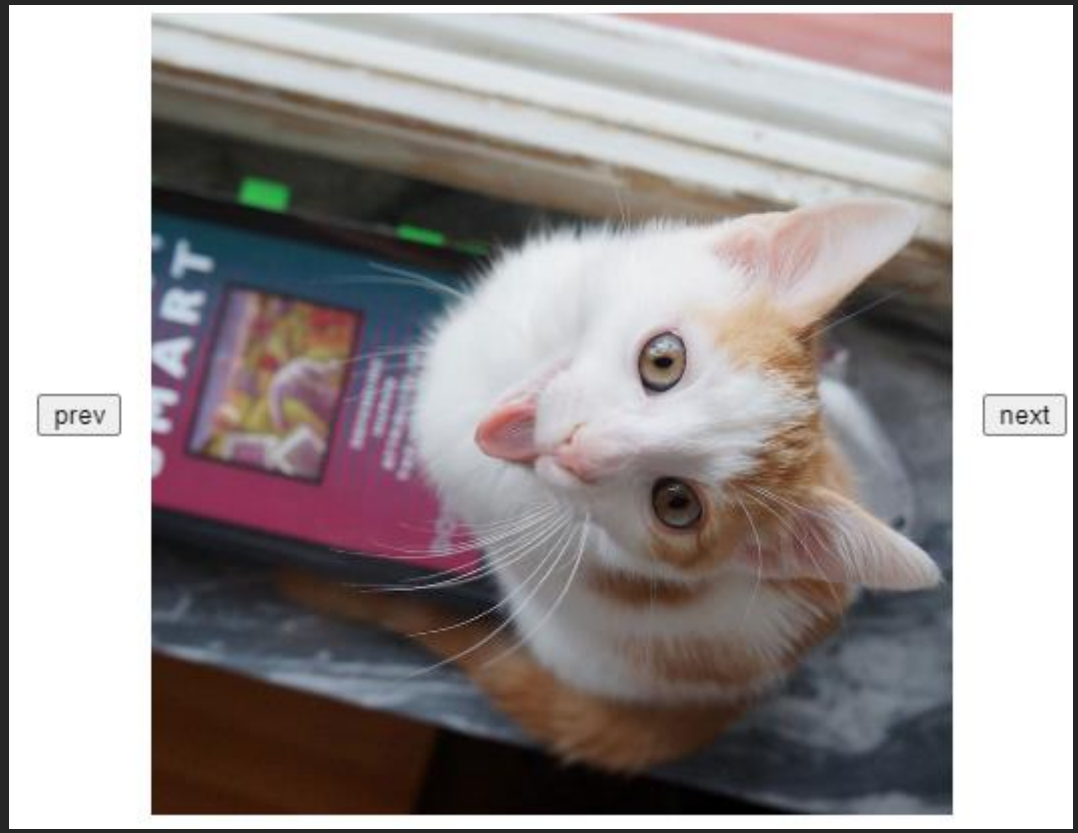
- systematic
  - image1.png
  - image2.png

how

images[i].style.display

img.src = `image${i}.png`

Break down components



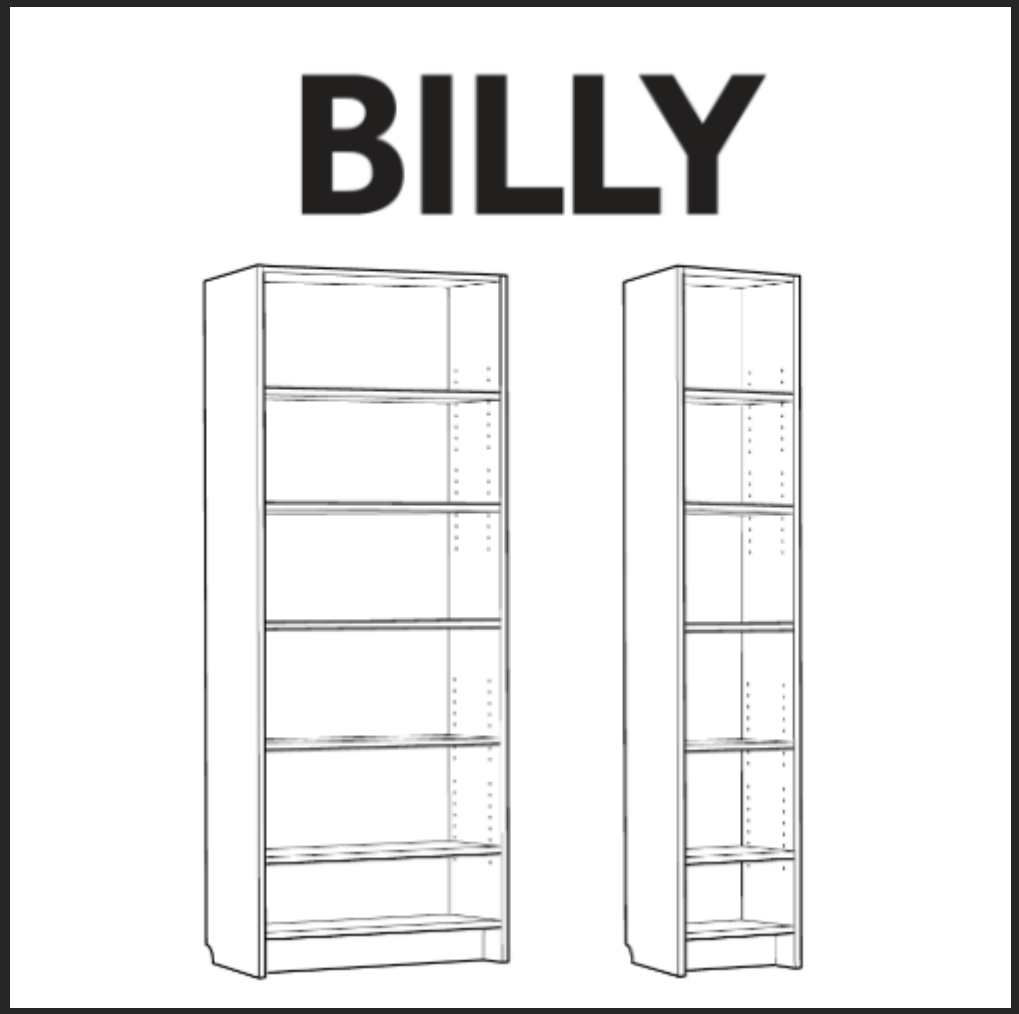
```
const $prev = document.querySelector("button:nth-child(1)")
const $next = document.querySelector("button:nth-child(3)")
const $output = document.querySelector("img")
const url = "images/image%i.jpg"
const MAX = 5
const MIN = 1
let index = MIN;
function update(i){
    index += i;
    if (index > MAX){
      index = MIN
    }
    if (index < MIN){
      index = MAX
    }

    $output.src=url.replace("%i", index)
}
$next.addEventListener("click",()=>{
    update(1)
});
$prev.addEventListener("click",()=>{
    update(-1)
});
update(0)
```

https://codepen.io/Philipp-Hock/pen/rNPKVre

LINKÖPING UNIVERSITY

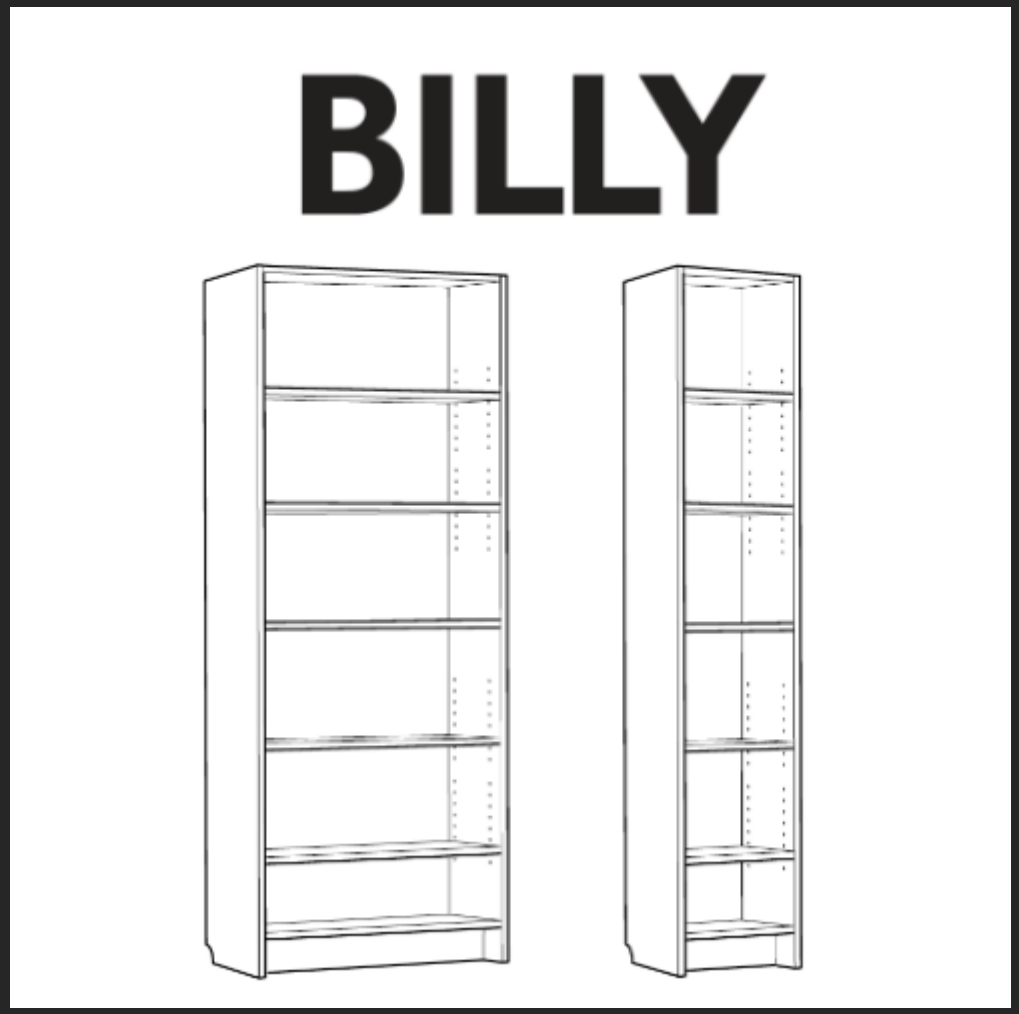
Object Oriented Programming
OOP

Classes, Objects, Instances



- Has properties
  - Height
  - Width
  - Elements
- Has one or more functions
  - Store things
  - Retrieve things

Classes, Objects, Instances



- Concept
  - Blueprint
  - Single description
  - Multiple instances

- Billy = Class

Classes, Objects, Instances

- Created from Blueprint
- Can be multiple instances
- Differ from each other
  - Width
  - Height
  - Elements

- This Billy is an object (instance)

```
class BillyShelf {
  constructor(width, height) {
    this.width = width;
    this.height = height;
    this.elements = [];
  }

  addElement(element) {
    this.elements.push(element);
  }

  displayInfo() {
    console.log(`Billy Shelf - Width: ${this.width}, Height: ${this.height}`);
    console.log("Elements:", this.elements.join(", "));
  }
}
```

```
// Creating instances of BillyShelf
const billy1 = new BillyShelf(80, 200); // Width: 80, Height: 200
billy1.addElement("Books");
billy1.addElement("Decor");

const billy2 = new BillyShelf(60, 180); // Width: 60, Height: 180
billy2.addElement("Candles");
billy2.addElement("Plants");

// Displaying information about the Billy shelves
billy1.displayInfo();
console.log("\n");
billy2.displayInfo();
```

```
// Subclass inheriting from BillyShelf
class Bookshelf extends BillyShelf {
  constructor(width, height, numShelves) {
    // Call the constructor of the superclass using super()
    super(width, height);
    this.numShelves = numShelves;
  }

  displayBookshelfInfo() {
    console.log(`Number of Shelves: ${this.numShelves}`);
  }
}

// Creating instances
const smallBookshelf = new Bookshelf(60, 150, 3);
const largeBookshelf = new Bookshelf(80, 200, 5);

// Adding elements to bookshelves
smallBookshelf.addElement("Books");
largeBookshelf.addElement("Novels");
smallBookshelf.displayInfo();
smallBookshelf.displayBookshelfInfo();
```

ARIA Attributes

https://www.w3.org/TR/wai-aria-1.1/#widget_roles

- **A**ccessible
- **R**ich
- **I**nternet
- **A**pplications

- Set of defined attributes that web content and web applications more accessible (e.g. when using a screen reader).
- Attributes inform ARIA aware browsers and application of the semantic meaning of user interface components

ARIA Attributes

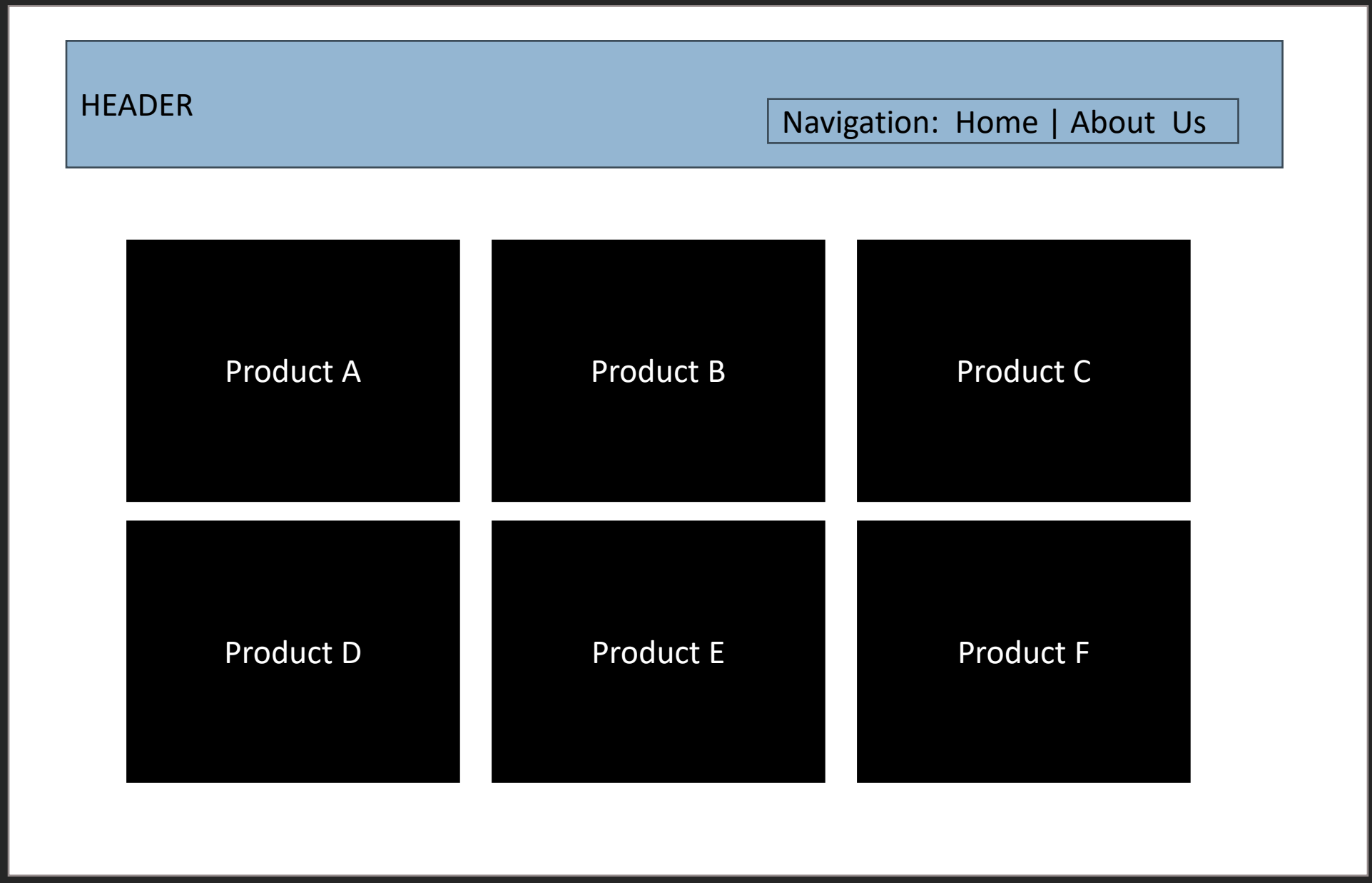
```
<div
  id="percent-loaded"
  role="progressbar"
  aria-valuenow="75"
  aria-valuemin="0"
  aria-valuemax="100">
</div>
```

More ARIA attributes (incomplete)

aria-label: Text label for element.
aria-labelledby: Referenced label IDs.
aria-describedby: Descriptive info IDs.
aria-hidden: Visibility for assistive tech.
aria-expanded: Collapsible state.
aria-selected: Selection status.
aria-disabled: Disabled state.
aria-checked: Checkbox/radio status.
aria-haspopup: Popup presence.
aria-live: Live-update region.
aria-controls: Controlled elements.
aria-posinset, aria-setsize: Item position/total.
aria-activedescendant: Active descendant.
aria-valuemin, aria-valuemax, aria-valuenow: Range values.
aria-roledescription: Element role description.
aria-atomic: Atomic updates.
aria-modal: Modal dialog indication.
aria-flowto, aria-labelledby: Reading order.
aria-haspopup: Activates popup.
aria-owns: Owned elements association.

LINKÖPING UNIVERSITY

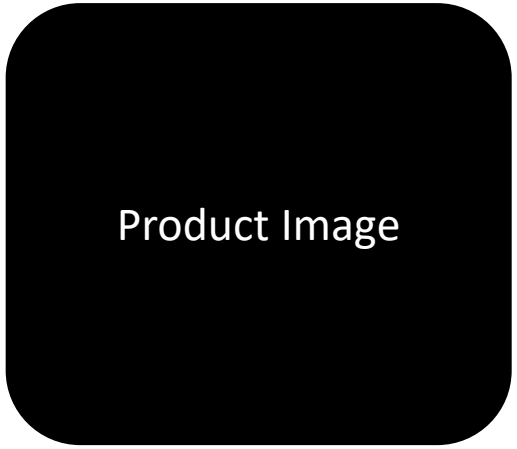
Multiple Page Interacton


HEADER
Navigation: Home | About Us
Product A
Product B
Product C
Product D
Product E
Product F

HEADER

Navigation: Home | About Us

Product C



Product Details

Description
Price

Passing data between pages

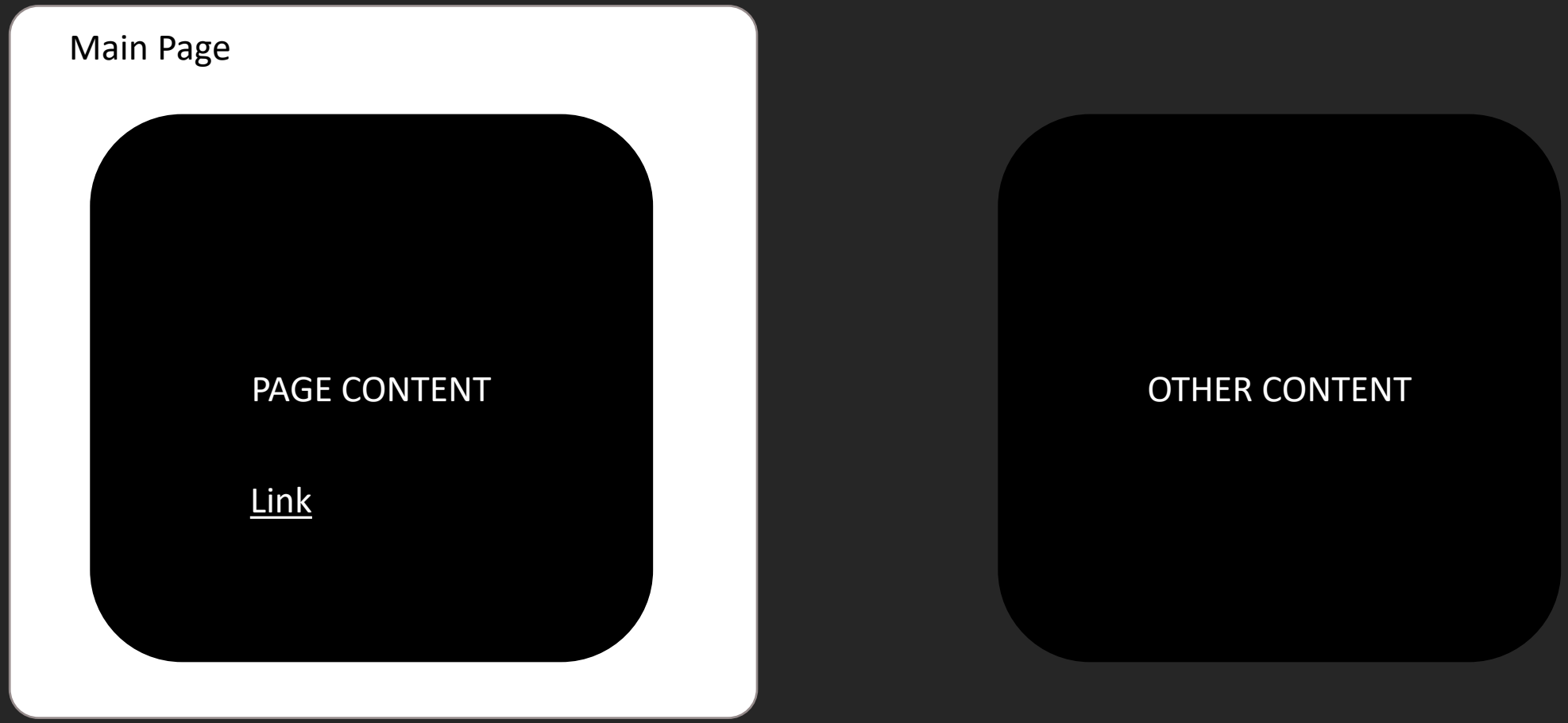
- Server-side logic
- Single-page application
- URL Parameter
- Browser storage

Server-side logic

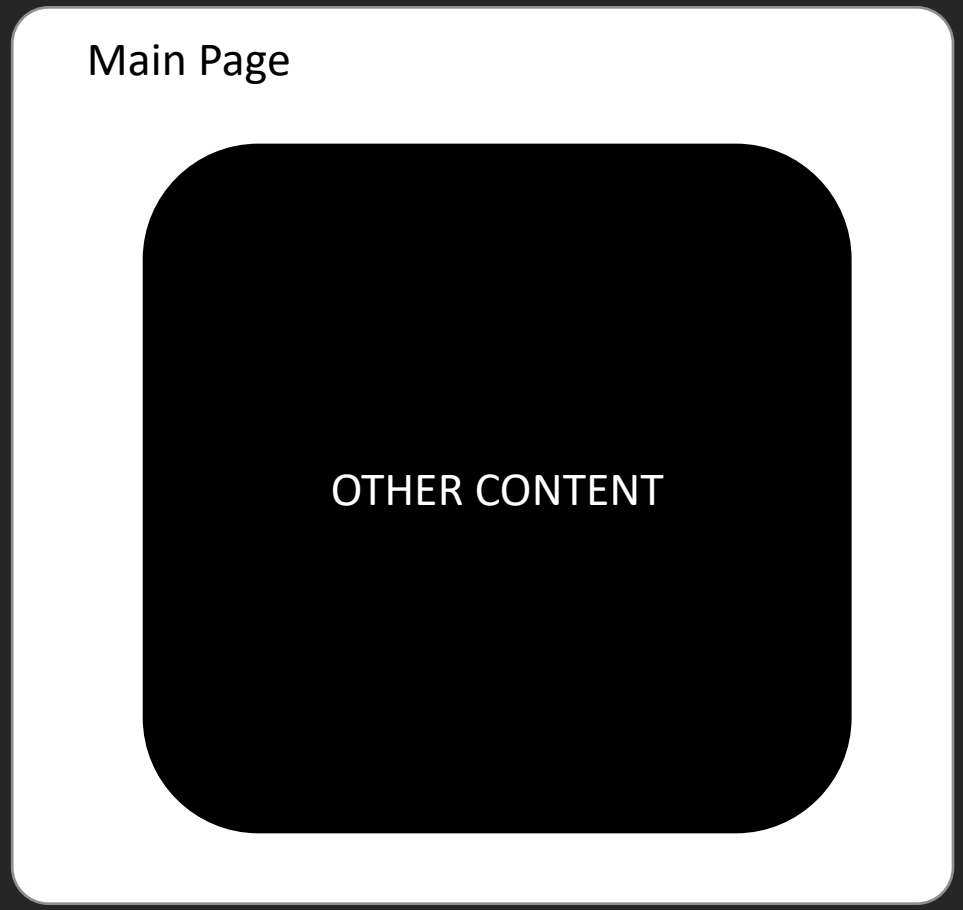
Product

```
{
  "id": 12345,
  "name": "Example Product",
  "description": "product description.",
  "price": 29.99
}
```

Single Page Application



Single Page Application



- Page itself is never changed
- Javascript handls content loading
- Typically done with a framework
  - React
  - Svelte
  - Vue
  - Angular

URL Parameter

https://mywebsite.com/product?name=product1&description=some%20text

Page 1

```
<a href="product?name=product1&description=some%20text">
Visit Product
</a>
```

URL Parameter

```
<script>
  // Read parameters from the URL
  const urlParams = new URLSearchParams(window.location.search);
  const productName = urlParams.get('name');
  const productDescription = urlParams.get('description');

  // Render the parameters in HTML
  const displayElement = document.body;
  displayElement.innerHTML += `<p><strong>Name:</strong> ${productName}</p>`;
  displayElement.innerHTML += `<p><strong>Description:</strong> ${productDescription}</p>`;
</script>
```

Browser storage

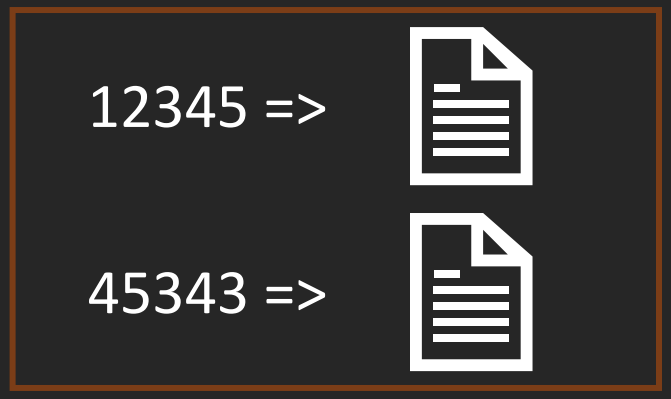
Product

```
{
  "id": 12345,
  "name": "Example Product",
  "description": "product description.",
  "price": 29.99
}
```

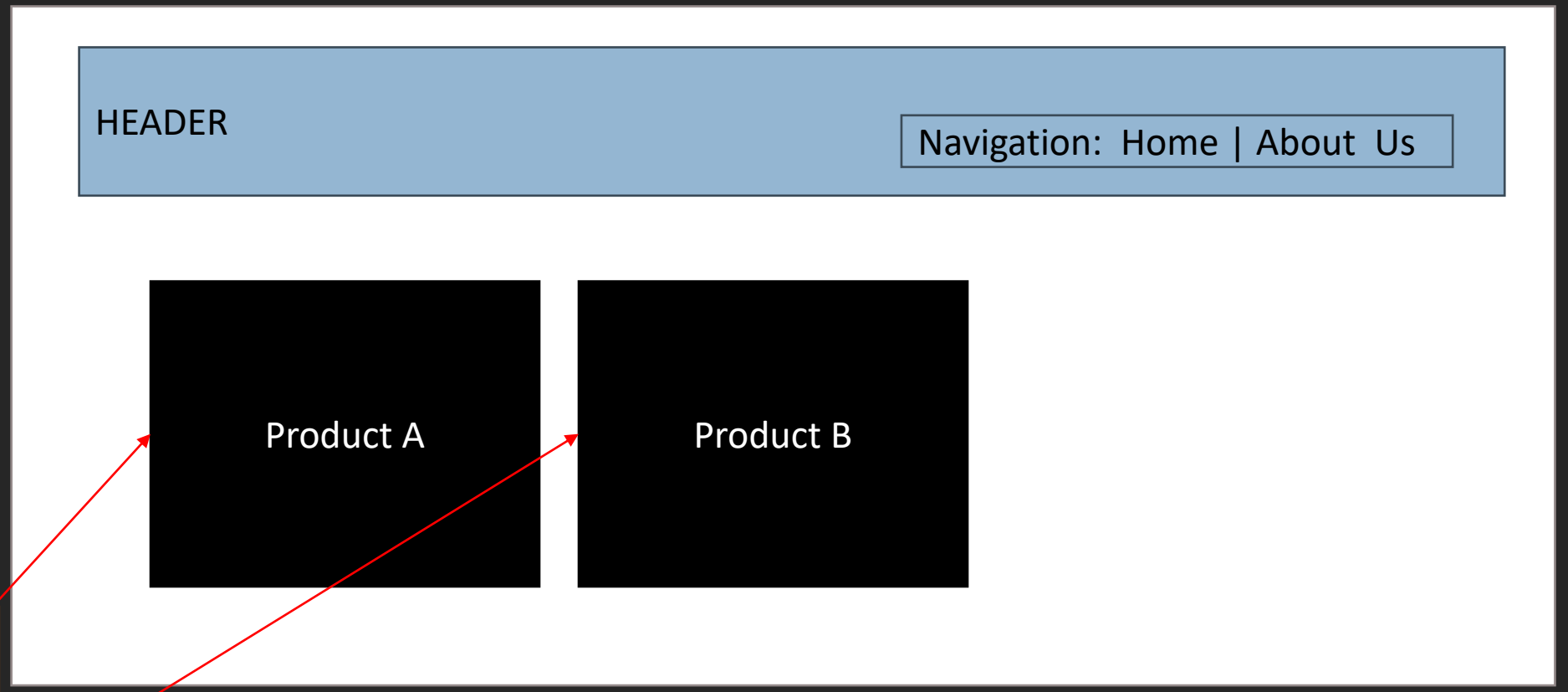
Product

```
{
  "id": 45343,
  "name": "Example Product 2",
  "description": “different description.",
  "price": 232.44
}
```

Local Storage



Browser storage



Local Storage

12345 =>

45343 =>

https://gitlab.liu.se/729g87/multipage-data-js
https://multipage-data-js-729g87-4c5f7f4c3de0d318fda4b67bb168b21ad28a21.gitlab-pages.liu.se/

Browser storage