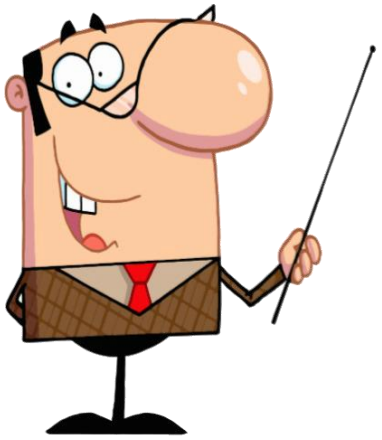
Client-side Technology JavaScript and Bootstrap

4

Web Technology



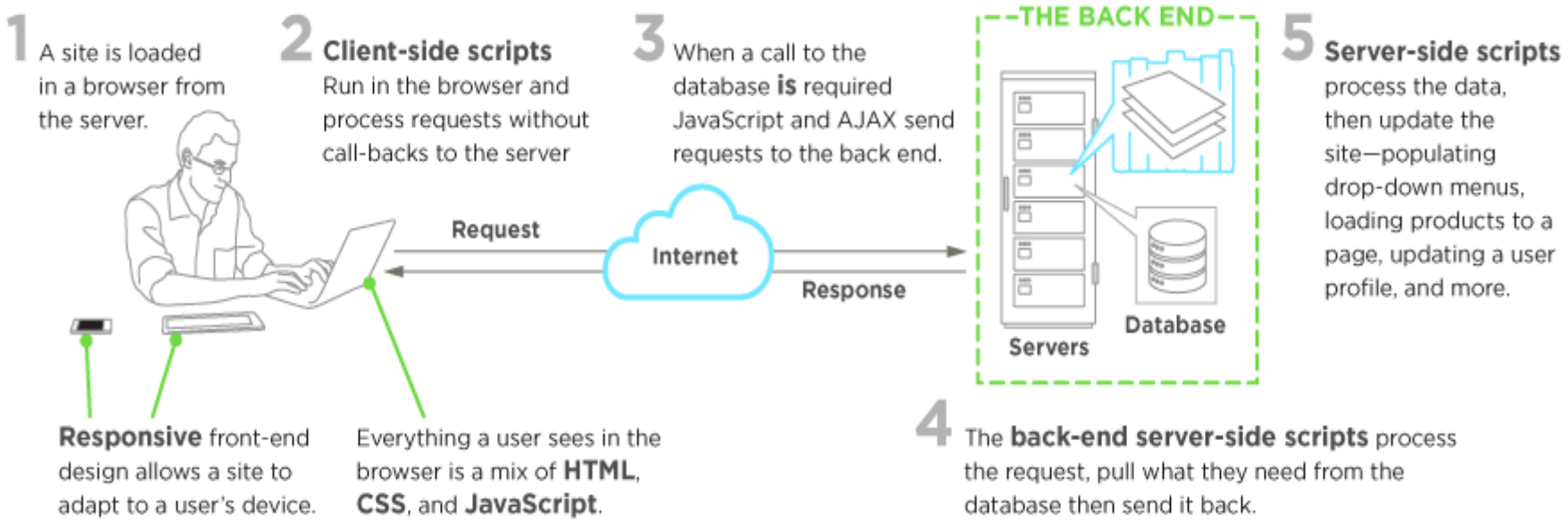
Asst. Prof. Manop Phankokkruad, Ph.D.
Faculty of Information Technology
King Mongkut's Institute of Technology Ladkrabang

Outline

- Introduction
- JavaScript
  - Event-driven programming
  - Document Object Model
  - On to forms processing
- Bootstrap Framework

Introduction

Client-side Development

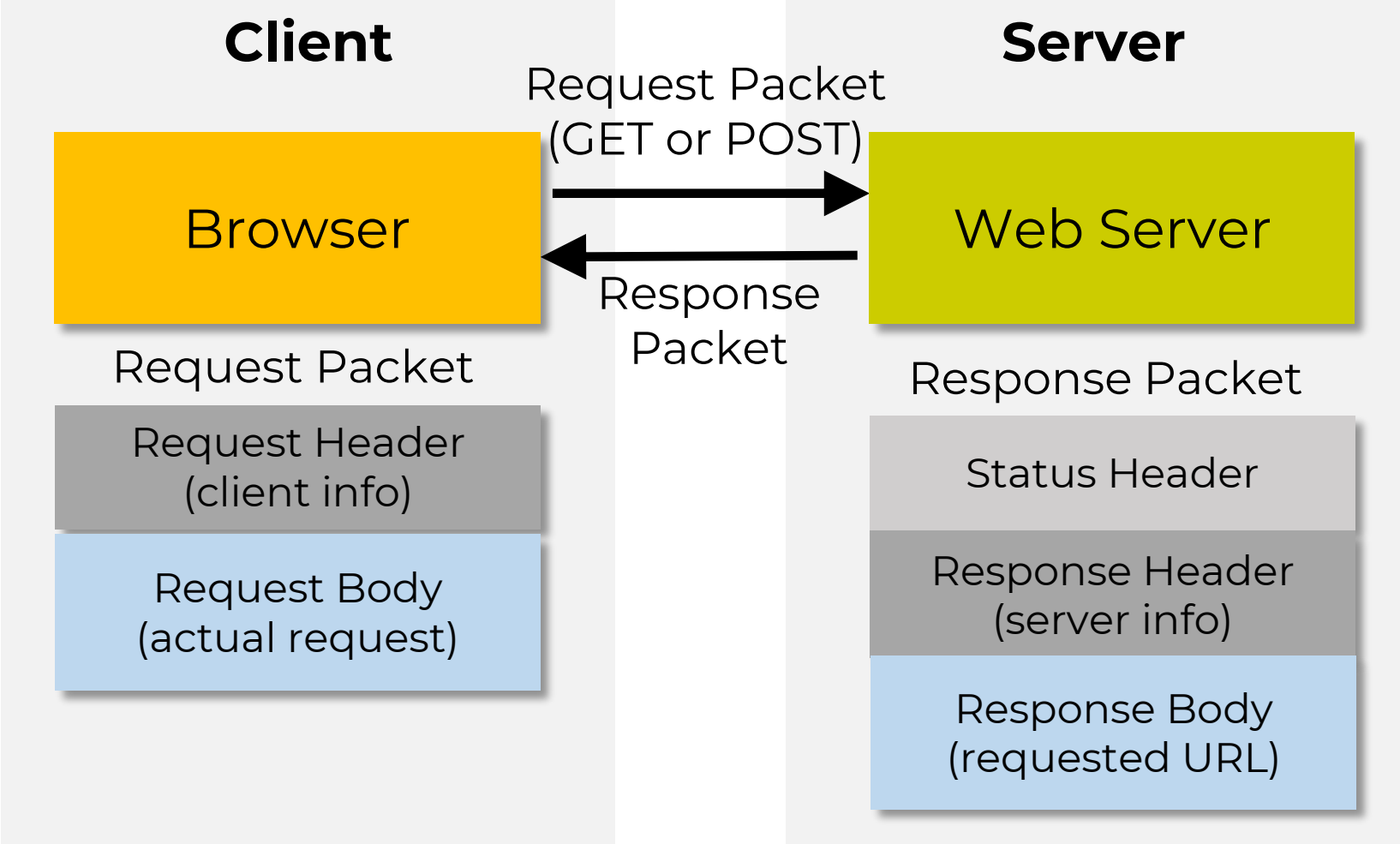


All websites run on three components: the server, the database, and the client.

- Server-side scripting is executed by a web server.
- Client-side scripting is executed by a browser.

Introduction

Request-Response Model



Introduction : GET and POST

- In the HTTP protocol, data from the client is sent to the server in one of two primary ways:
- **GET**

- attaches the forms data (URL encoded) to the requested URL and places the URL in the body of an http request packet.
- the URL has the data attached it is visible in the Location box of the browser, Since this allow the data to be seen by any with in view of the screen it is not preferred.

Introduction : GET and POST

- In the HTTP protocol, data from the client is sent to the server in one of two primary ways: (cont.)
- **POST**
  - places the forms data into the body of the http request packet, this way the data will not be seen by snooping eyes. It is more private than GET. This is the preferred way to send data to the server.

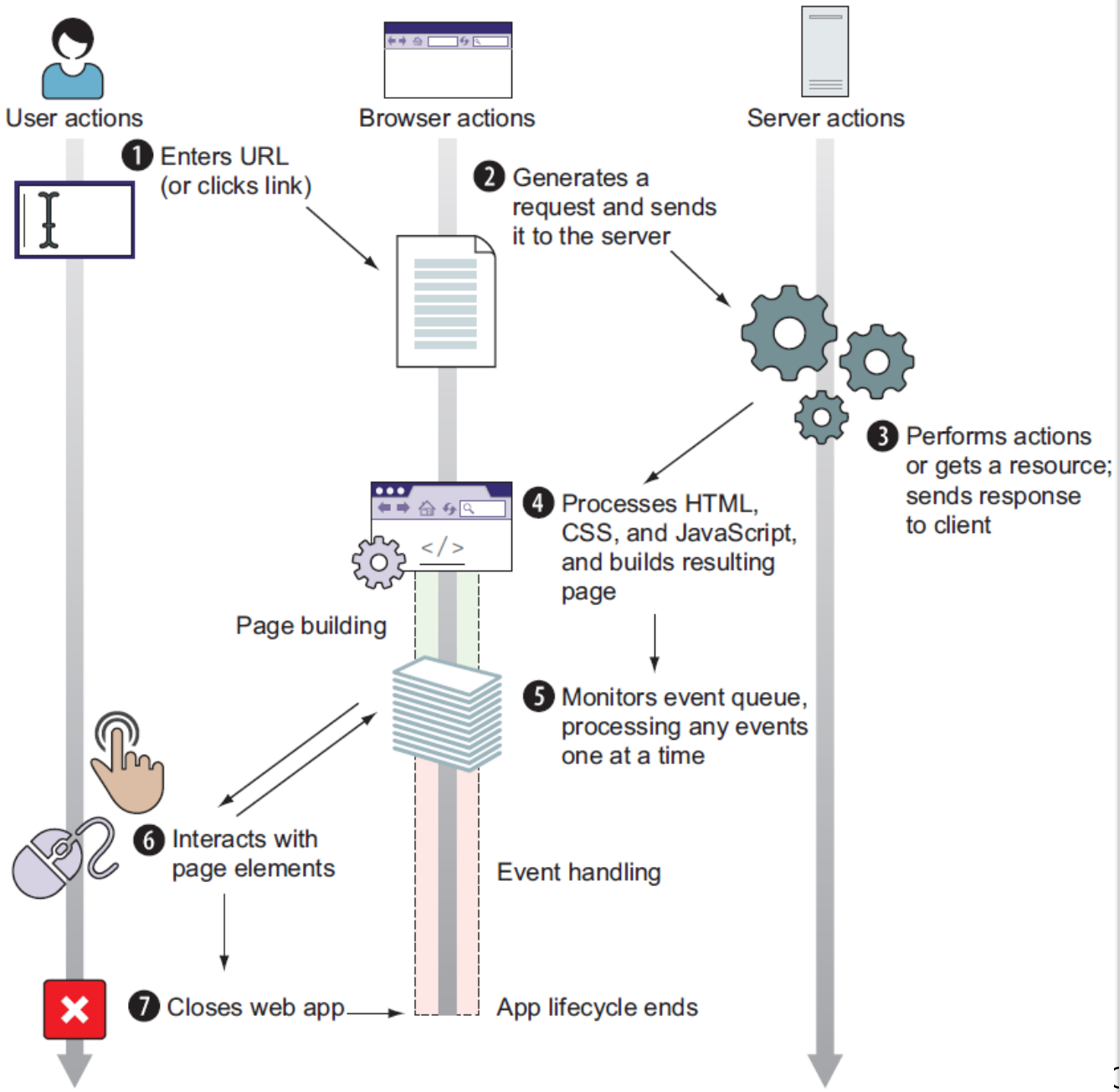
Introduction

Client-side scripting benefits:

- **Scripts** are embedded within and interact with the HTML of Website. **Scripts** interact with a cascading style sheet (CSS) file that styles the way the page looks. **Scripts** put less stress on the server because they don't require processing on the server.
- **Usability**: can modify a page without having to post back to the server (faster UI).
- **Efficiency**: can make small, quick changes to page without waiting for server.
- **Event-driven**: can respond to user actions like clicks and key presses.

Introduction

The lifecycle of a client-side web application.



JavaScript

2

- **JavaScript** is a high-level, interpreted programming language. It is a language which is also characterized as dynamic, weakly typed, prototype-based and multi-paradigm.
- **JavaScript** is one of the three core technologies of the World Wide Web. JavaScript enables interactive web pages and thus is an essential part of web applications.
- **JavaScript** is NOT related to Java other than by name and some syntactic similarities

JavaScript

- A lightweight programming language
- used to make web pages interactive.
- react to events (event-driven).
- get information about a user’s computer.
- perform calculations on user's computer.
- more relaxed syntax and rules
- key construct is the function rather than the class.
- contained within a web page and integrates with its HTML/CSS content

JavaScript Frameworks

- JavaScript frameworks are a type of tool that makes working with JavaScript easier and smoother. These frameworks also make it possible for the programmer to code the application as a device responsive. The popular JavaScript frameworks such as.
  - Angular-JS
  - React.JS
  - Vue.js
  - jQuery
  - Ajax
  - and more...

JavaScript : Basic

HTML

```
<script src="filename.js" > </script>
```

```
<script>
        JavaScript code.....
</script>
```

- JavaScript code can be placed directly in the HTML files <body> or <head> tag.
- The scripts inside an HTML document is interpreted in the order they appear in the document. Scripts in a function is interpreted when the function is called. So where you place the **<script>** tag matters.

JavaScript : Variables

Declaration

var, const, let

- **var** - The most common variable. Can be reassigned but only accessed within a function. Variables defined with var move to the top when code is executed.
- **const** - Can not be reassigned and not accessible before they appear within the code.
- **let** - Similar to **const**, however, let variable can be reassigned but not re-declared

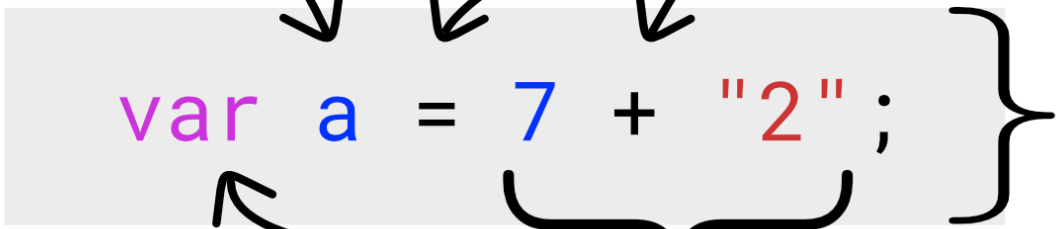
JavaScript : Data types

- Numbers : var num = 23.08
- Strings : var str = ”Hello”
- Boolean (true or false) : var b = true
- Null : var   n = null
- Undefined : var und = undefined
- Symbol : var sym = Symbol(‘Something’)
- Constant numbers : const PI = 3.14
- Objects : var name = { firstName:"John", lastName:”Doe” }
- Arrays : var fruit = ["Banana", "Apple", "Pear"];

JavaScript : Data types

Variable
A named reference to a value is a variable.

Operator
Operators are reserved-words that perform action on values and variables.
Examples: `+` `-` `=` `*` `in` `===` `typeof` `!=` ...



Statement
A group of words, numbers and operators that **do a task** is a statement.

Note: `var`, `let` & `const` are all valid keywords to declare variables. The difference between them is covered on page 7 of this cheatsheet.

Keyword / reserved word
Any word that is part of the vocabulary of the programming language is called a keyword (a.k.a reserved word).
Examples: `var` `=` `+` `if` `for`...

Expression
A reference, value or a group of reference(s) and value(s) combined with operator(s), **which result in a single value.**

JavaScript : Conditional Statements

JavaScript has the following conditional statements:

Syntax

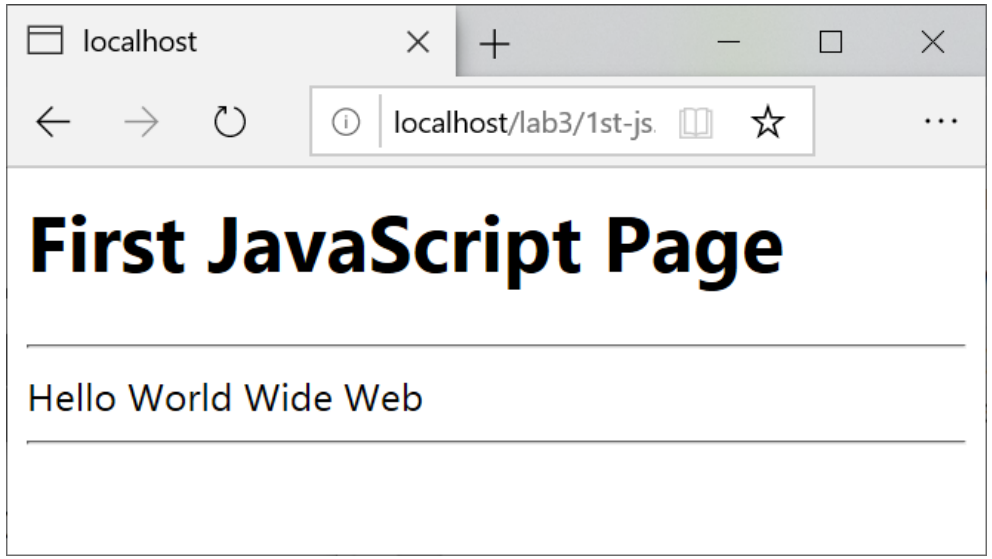
```
if (condition) {
  //  block of code to be executed if the condition is true
} else {
  //  block of code to be executed if the condition is false
}
```

Example

```
if (hour < 18) {
  greeting = "Good day";
} else {
  greeting = "Good evening";
}
```

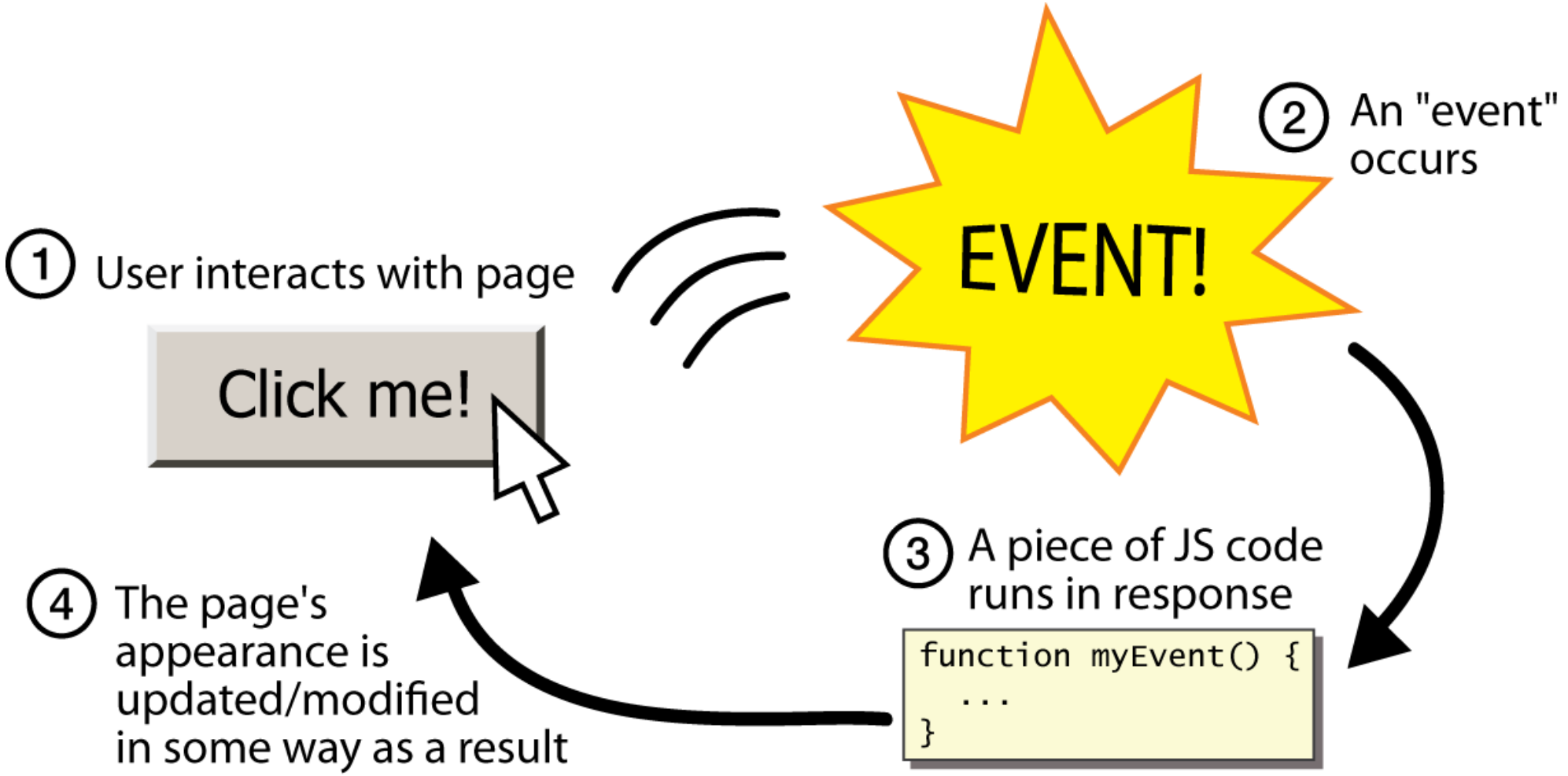
JavaScript : A Simple Script

```
<html>
<head></head>
<body>
<h1>First JavaScript Page</h1>
<script type="text/javascript">
    document.write("<hr>");
    document.write("Hello World Wide Web");
    document.write("<hr>");
</script>
</body></html>
```



JavaScript : Event driven

Event-driven programming



JavaScript : Event driven

Event-driven programming (cont.)

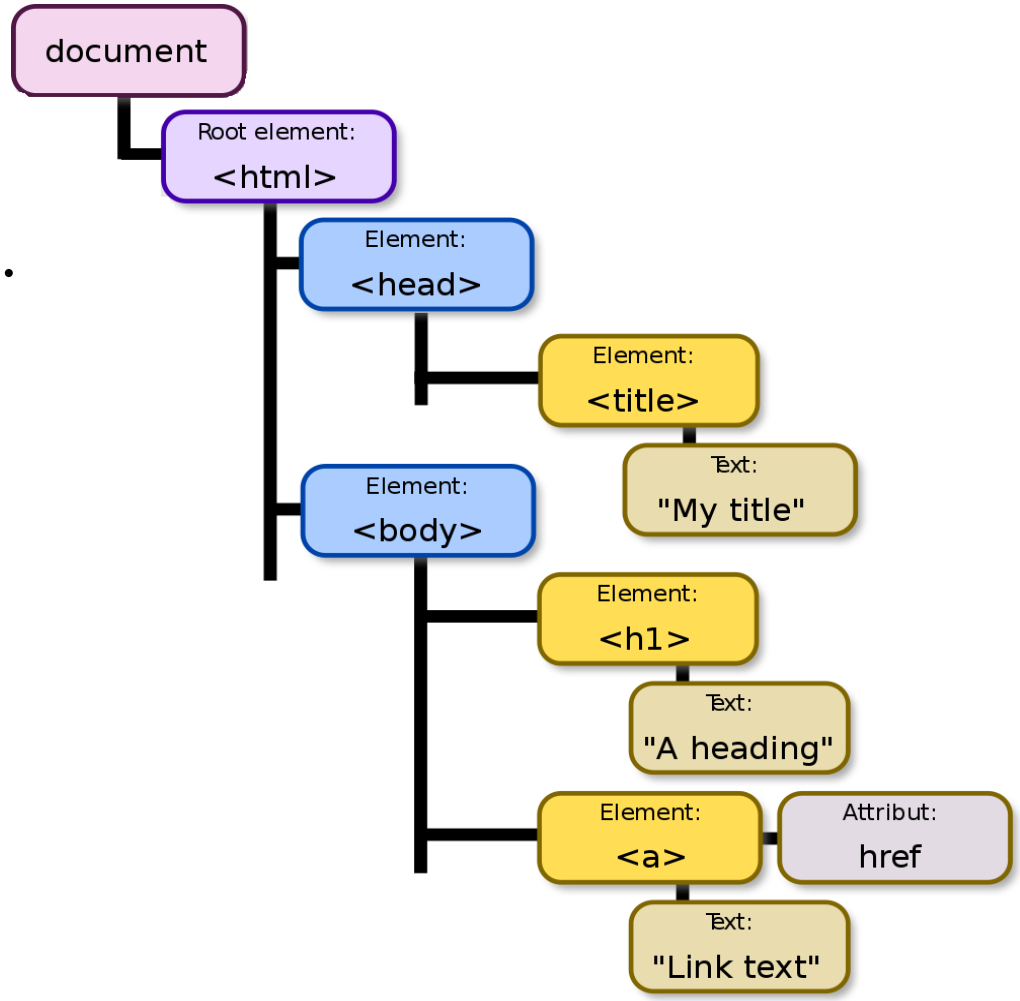
- JavaScript programs instead wait for user actions called ***events*** and respond to them.
- event-driven programming: writing programs driven by ***user events.***

JavaScript : DOM

- Most JavaScript code manipulates elements on an HTML page

We can

- examine elements state.
- change state
- change styles



JavaScript : DOM

DOM element objects

HTML

```
<p>
  Look at this octopus:
  <img src="octopus.jpg" alt="an octopus" id="icon01" />
  Cute, huh?
</p>
```

DOM Element Object

Property	Value
tagName	"IMG"
src	"octopus.jpg"
alt	"an octopus"
id	"icon01"

JavaScript

```
var icon = document.getElementById("icon01");
icon.src = "kitty.gif";
```

JavaScript : Accessing elements

Accessing elements:

```
document.getElementById(“id”)
document.getElementByClassname(“Classname”)
document.getElementByName(“Name”)
document.getElementByTagName(“TagName”)
document.forms[“Formname”][“ElementId”]
```

- **document.getElementById** returns the DOM object for an element with a given id. It can change the text inside most elements by setting the **innerHTML** property. Also It can change the text in form controls by setting the value property.

Accessing elements

JavaScript

```
let name = document.getElementById("id");
let fname = document.forms["myForm"]["FirstName"].value;
```

HTML

```
<button onclick="changeText();">Click me!</button>
<span id="output">replace me</span>
<input id="textbox" type="text" />
```

JavaScript

```
function changeText() {
  let span = document.getElementById("output");
  let textBox = document.getElementById("textbox");
   textbox.style.color = "red";    }
function getValue() {
     let fname = document.forms["myForm"]["FirstName"].value;
}
```

JavaScript : On to forms processing

The processing of a form is done in two parts:

- **At the browser**, before the data is passed to the back-end processors.
  - **Client-side :** should be done via JavaScript.
  - The data validation done locally to relieve the back-end processors of having to do it.
- **Server-side** : processing done in the back-end system (CGI Scripts, VBScript, JavaScript, etc.

JavaScript : Validation Functions

Validation functions can usually be written in either of two ways

1. **Iterative programming methodologies** : write a small function that iterates through the user input and validates the data type.
2. **Pattern matching** : create a regular expression pattern for the data type and let the system validate the user input via the pattern matching capabilities of the pattern matching is done at a low level in the JavaScript processor, so validation is very quick.

1: JavaScript Form Validation

One major use of JavaScript in conjunction with HTML forms is for validation of forms data before allowing the data to be submitted to a backend server for processing.

- doing the data validation at the client (browser) relieves the server from having to do it
  - on a busy server this can free up a lot of processing cycles that can be better used to process more user requests and/or backend database functions.

1: JavaScript Form Validation

Validation constraints on input elements starting simple.

HTML

```
<form name="myForm"
onsubmit="validateForm()" method="post">
Name: <input type="text" name="fname">
<input type="submit" value="Submit">
</form>
```

JavaScript

```
function validateForm() {
    let x = document.forms["myForm"]["fname"].value;
    if (x == "") {
        alert("Name must be filled out");
        return false;
    }
}
```

2: HTML5 Validation

New attributes for <input>

- required =‘required’
- pattern= ‘a regular expression’ can be used with <input> types, **text**, **search**, **url**, **telephone**, **email**, and **password**.
- min = max = step =
  can be used with <input type=range>

2: HTML5 Validation

The simplest HTML5 validation feature to use is the **required** attribute

HTML

```
<form name="myForm"  method="post">

Name: <input type="text" name="fname“ required>
Age :<input type="number" min="12" max="120"
step="1" name=“age”  pattern="\d+">
Email : <input type="email"  name="email">

<input type="submit" value="Submit">
</form>
```

Bootstrap

3

What is Bootstrap?

- Bootstrap is the most popular front-end framework for quickly styling website and help you with website development.
- The framework basically consists of the set of two files; Bootstrap.js and Bootstrap.css. If you include them on your HTML page or website, it will enable you to use its HTML components and features.

Bootstrap

What Bootstrap Package Includes?

1. Scaffolding: Bootstrap provides a basic structure with Grid System, link styles, background.
2. CSS: Bootstrap comes with feature of global CSS settings, fundamental HTML elements styled and enhanced with extensible classes, and an advanced grid system.
3. Components: Bootstrap contains over a dozen reusable components built to provide iconography, dropdowns, navigation, alerts, popovers, and more.

Bootstrap

What Bootstrap Package Includes? (cont.)

4. JavaScript Plugins: Bootstrap contains over a dozen custom jQuery plugins. You can easily include them all, or one by one.
5. Customize: You can customize Bootstrap's components, LESS variables, and jQuery plugins to get your very own version.

Bootstrap

Using Bootstrap

- You can start using Bootstrap 4 in your website by including it from
- CDN (Content Delivery Network)

```
<link
href="//maxcdn.bootstrapcdn.com/bootstrap/4.1.1/css/bootstrap.min.css" rel="stylesheet">

<script
src="//maxcdn.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"></script>
```

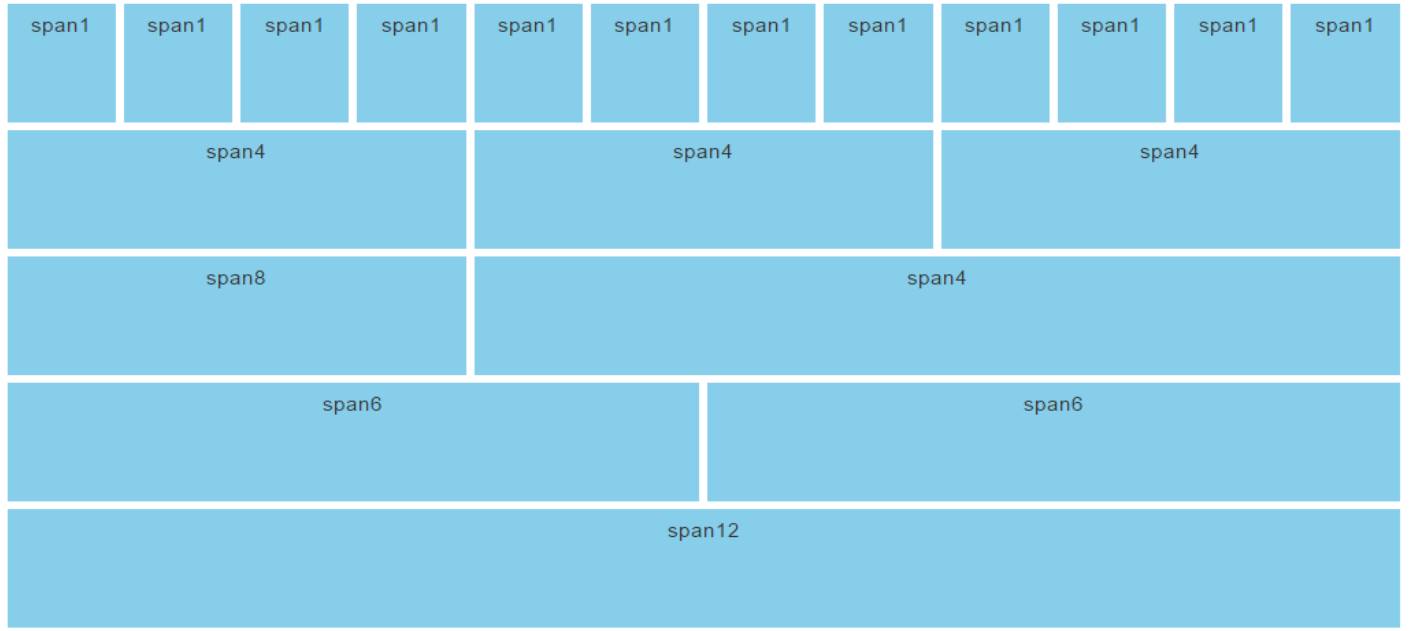
- Downloading from getbootstrap.com.

Bootstrap Components

- Grids – Defining pages in terms of columns and rows
- Typography – Headings / body elements
- Glyphs – icons that are font based, scalable
- Navigation – responsive navigation
- Images – responsive images
- Helper Classes – Clearfix / show / hide / centers
- Responsive Utilities – hide / show content via class
- Javascript - Carousel, Tooltip, Tab, Modal, Alerts

Bootstrap Grid System

- Bootstrap divides a page into a grid of 12 columns and multiple rows for easier positioning of elements.
- Grid system is responsive, and columns will rearrange automatically depending on the screen size.



Bootstrap Grid System

Grid System Basics

- Place columns inside for each row.
- Set each column width using class prefixes: .col-xs-*, .col-sm-*, .col-md-*, .col-lg-*

```
<div class="row">
<div class="col-md-3">.col-md-3</div>
<div class="col-md-6">.col-md-6</div>
<div class="col-md-3">.col-md-3</div>
</div>
```

Bootstrap Form

Three types of forms in Bootstrap are:

- Vertical or Basic form (By default)
- Horizontal form
- Inline form

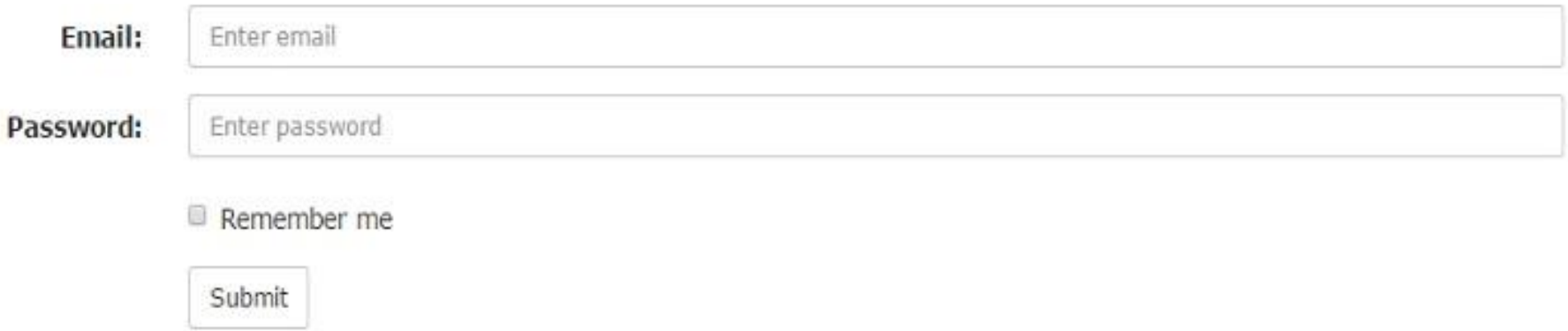
Rules For Form Layouts

- Use of role attribute in <form role="form"> this improves accessibility for screen readers.
- Wrap up labels and form controls in for better spacing.
- Use of class ="form-control" in all text elements (<input>, <select>, <textarea>)

Bootstrap Form

❑ Vertical / Basic form

❑ Horizontal form



Bootstrap Form

- Inline form

HTML of Inline form

```
<form class="form-inline" role="form">
<div class="form-group">
<label for="email">Email:</label>
<input type="email" class="form-control" id="email" placeholder="Enter email">
</div>
<div class="form-group">
<label for="pwd">Password:</label>
<input type="password" class="form-control" id="pwd" placeholder="Enter password">
</div>
<div class="checkbox">
<label><input type="checkbox"> Remember me</label>
</div>
<button type="submit" class="btn btn-default">Submit</button>
</form>
```

Bootstrap Tables

- The Bootstrap 4 has .table class that you may use in the <table> tag. By default, the table is created with light-grey horizontal dividers as shown below:

Bootstrap 4 Table Demo

Month	Number of Sales	Amount
Jan	105	$15,000.00
Feb	95	$12,000.00
Mar	150	$20,000.00

More Information

- JavaScript Tutorial
  https://www.w3schools.com/js/default.asp
- Bootstrap 4
- https://www.w3schools.com/bootstrap4/