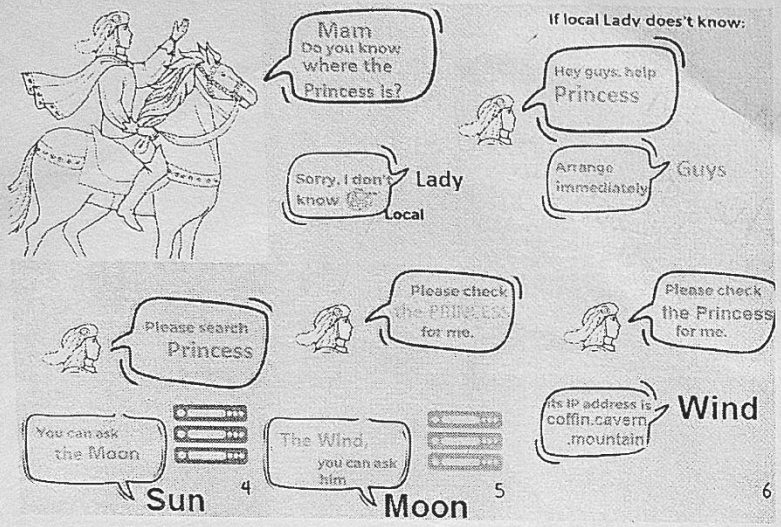
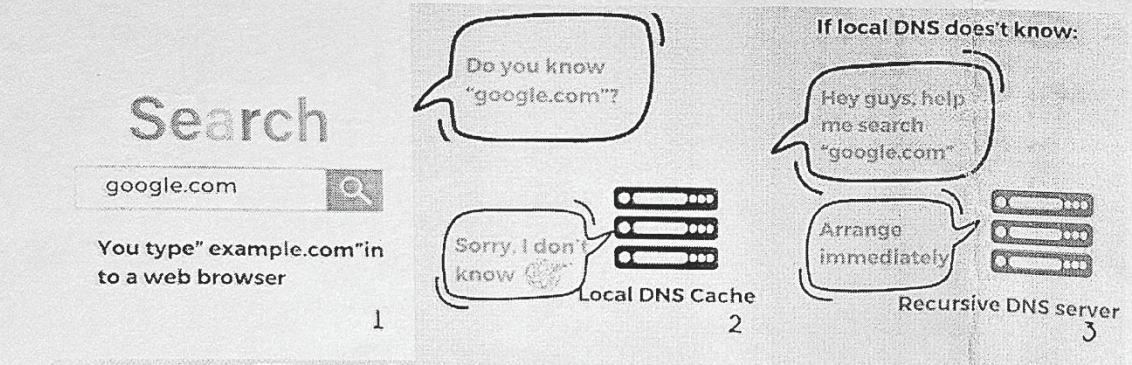
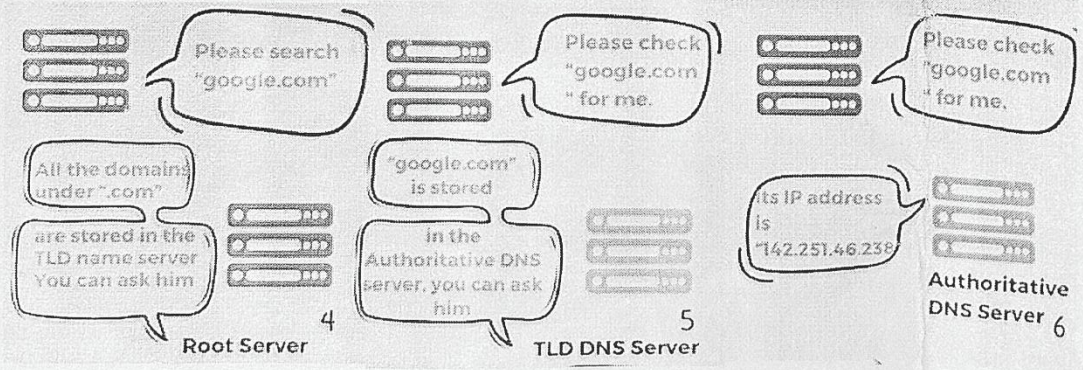
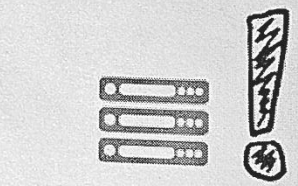








www.google.com

Recursive DNS Server

www.google.com → Go to ".com" TLD DNS Server — Root Server

www.google.com → Go to "google.com" Authoritative DNS Server — TLD DNS Server

www.google.com → 142.251.46.236 — Authoritative DNS Server

142.251.46.236

Domen by

DNS

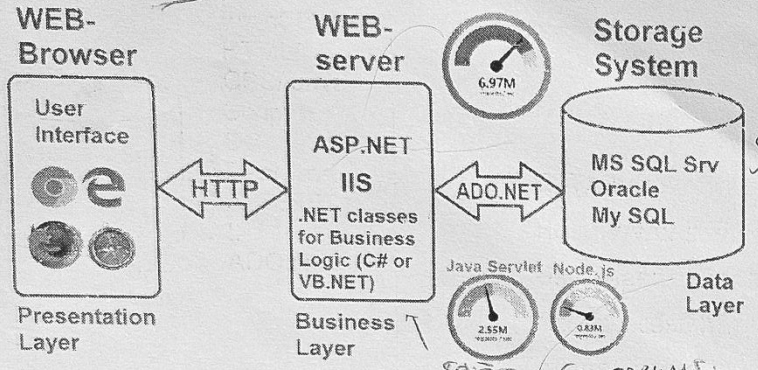
sb.bsu.by отдаёт index

sbmt.bsu.by index, Defau

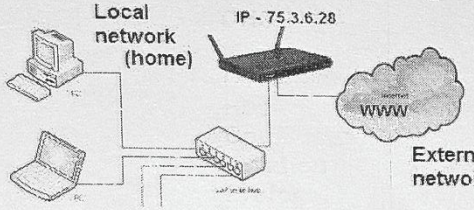
lecan - куратор.

по упр. домены. имена

+0.2 ~~from~~ to fin Gradle 8.9.25 15.9.25 +0.1 +0.1 18.9.25



Front End
HTML/ CSS/ Java Script и DOM
BackEnd
ASP.NET
IIS
ADO.NET



DHCP (engl. Dynamic Host Configuration Protocol)

tracert – traceroute (in Apple) www.geotraceroute.com

172 . 16 . 254 . 1

10101100.00010000.11111110.00000001

8 bits

32 bits (4 bytes)

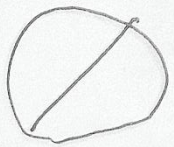
4 billion addresses – 4.294.967.296

DNS – clearer address.

217.21.43.40 – sbmt.by, sbmt.bsu.by sb.bsu.by

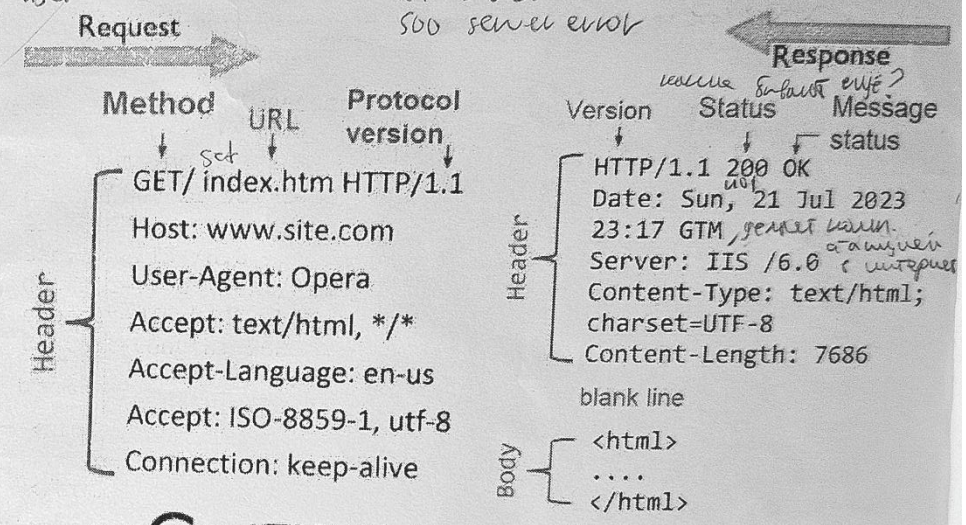
ping sb.bsu.by tracert sb.bsu.by https://geotraceroute.com/

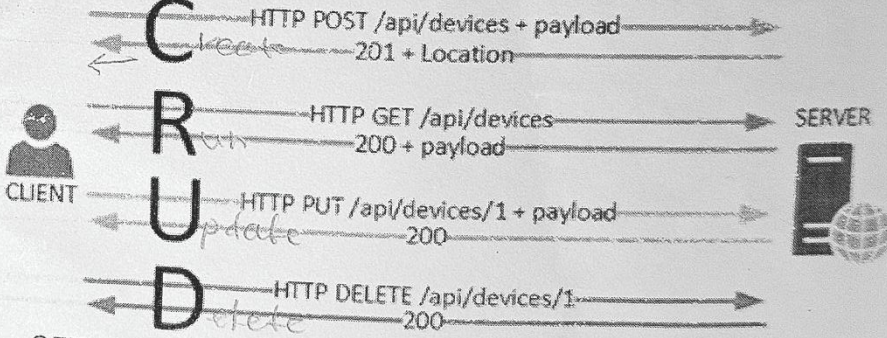
+375 (29) 254 07 92 - ANDREY O. YAROSHEVICH



+0.1 OK

HTTPS Status 100 continue 200 OK 301 redirection 401 error 500 server error





- GET — getting a resource
- POST — resource creation
- PUT — resource update
- DELETE — deleting a resource

Methods		Request		Response	
		URL	Request body	Status	Response body
1	GET	Yes	No	Yes	Yes
2	PUT	Yes	Yes	Yes	No
3	POST	Yes	Yes	Yes	Yes
4	DELETE	Yes	No	Yes	No

```
class ACat {
string name;
public ACat(n){
 this.name=n;
}
}
```

```
class ACat {
 constructor(n) {
  this.name = n;  //property
 }
}
mycat = new ACat("Barsik");
```

```
Say() {
  return "meou";
}
s =
"My cat says <b>"
 +
 myCat.Say()+
"</b> !"
```

ACat mycat= new ACat("Barsik");

```
<script>
class APet {
    Say() {
        alert("No");
    }
}
class ACat extends APet {
    Say() {
        return "Miou";
    }
}
var myPet =
 new ACat("Barsik");

 S = "My pet says  <b>"
      + myPet.Say()+
      "</b> !"
</script>
```

class Women — To describe a Woman: name, age, job
can do: eat, drink, sleep, walk, ...

Real world objects

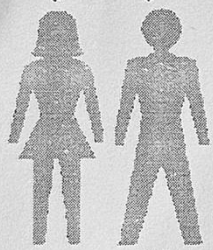
object — Jane 19 Student

object — Emma 45 Doctor

object — Ann 30 Engineer

Polymorphism in Biology: Queen, Drone, Worker


APerson


AWoman extends APerson

AMan extends APerson

function **WashDishes()**

Men's version
wipe dry

Female version
wet with water


ira= new AWoman()


ivan= new AMan()

```
class AMan {
  WashDishes() {
    return 'wipe dry';
  }
}
```

```
class AWomen {
  WashDishes() {
    return 'wet';
  }
}
```

```
var family = [new AWomen(), new AMan()];
for(i=0;i<2;i++){ alert(" "+ family[i].WashDishes());} //dry wet
```

```
women = new AWomen();  man = new AMan();  var family = [women,man];
```

+0.1 2025.9.24

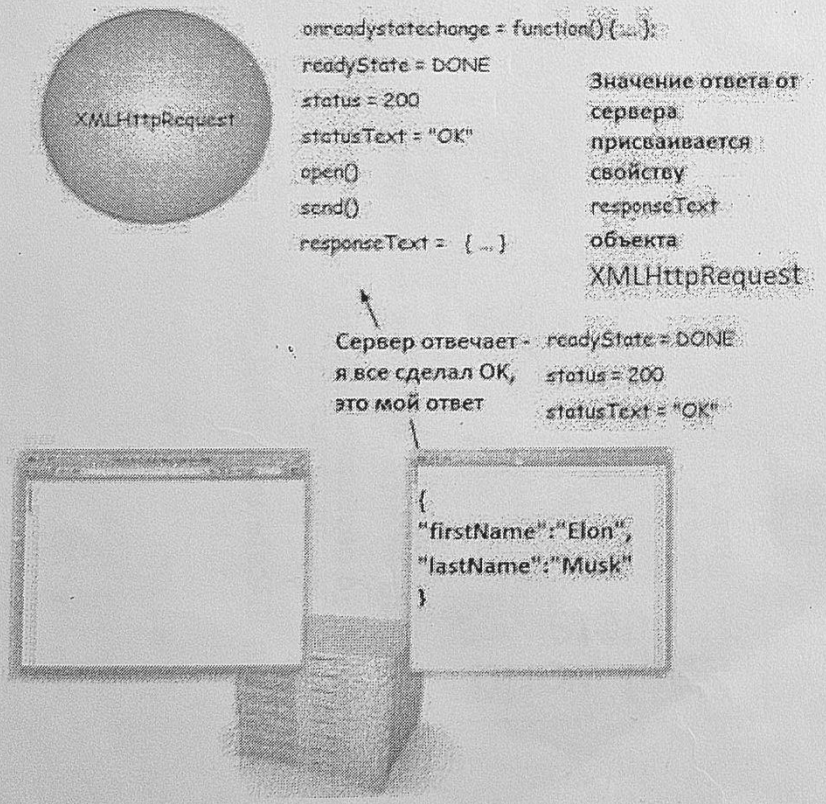
```
{"managers":[
 { "firstName":"Bill", "lastName":"Gates" },
 { "firstName":"Mike", "lastName":"Dell" },
 { "firstName":"Elon", "lastName":"Musk" }
]}
```

<=JSON vs XML=>

```
< managers >
 <manager>
   <firstName>Bill</firstName>
    <lastName>Gates</lastName>
 </manager>
 <manager>
   <firstName>Mike</firstName>
    <lastName>Dell</lastName>
 </manager>
 <manager>
   <firstName>Elon</firstName>
    <lastName>Musk</lastName>
 </manager>
</managers>
```

1) **JSON**.parse(minskJSON);

2) **localStorage**.setItem() getItem()



7) Работа с массивами myArr[2];

9) `obj.birth = new Date(obj.birth);`

10) "age":"function() {return 48;} " – ф-ция как строка

Yes	8.0	3.5	4.0	10.0

11) JSON.stringify() 14) Object => в JavaScript-строку

```
x = myObj["name"] + " " + myObj["Networth"];
// x = myObj. name + " " + myObj.Networth;
```

16) 17) 18) myObj.Occupation.pos3 myObj.Occupation["pos3"]

yunitever.bsite.net/web/Projects/

ES6:

asp.net.by/es6/1.htm, asp.net.by/es6/02.htm

```
{
 let
 const
}
```

asp.net.by/es6/03.htm **Arrow functions:**

```
x = (x, y) => y + ":" + x;
                    x("sbmt.by", "SBMT");
```

asp.net.by/es6/04.htm function myExam(y = 4) – **параметр по умолчанию**

asp.net.by/es6/05.htm Array.**find**(myFunction)

asp.net.by/es6/06.htm Array.**findIndex**()

asp.net.by/es6/07.htm Классы

```
class ACat {

  constructor(n) {

    this.name = n;     //property

  }

}
```

```
mycat = new ACat("Барсик");
```

```
class ACat{
.....
 Name()
  return 'Меня зовут ' + this.name;
 }
}
mycat = new ACat("Барсик");
document.write(mycat.Name());
```

asp.net.by/es6/08.htm asp.net.by/es6/09.htm Array.**forEach**() var colors = ["red", "green", "blue"];	Меня зовут Барсик item var colors = ["red", "green", "blue"]; 0 1 2 index

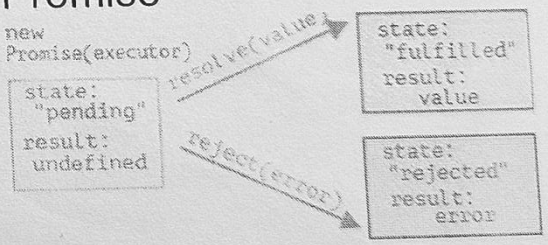
```
colors.forEach(myFunction);
function myFunction(item, index) {
   item ... index
}
```

```
Function One (){
     // Do something
 }
Function Two (call_One){
     // Do something else
     call_One()
}
Two(One);
```

code is being executed

Callback illustrated

Promise

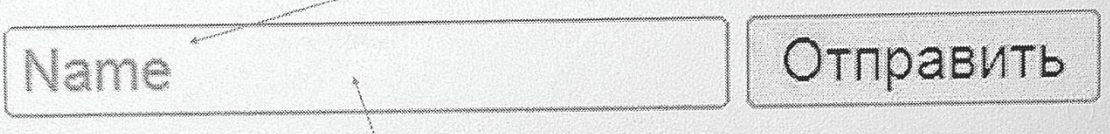


HTML forms

The placeholder - the thing the user sees filled into that input field originally

automatically focus this input field

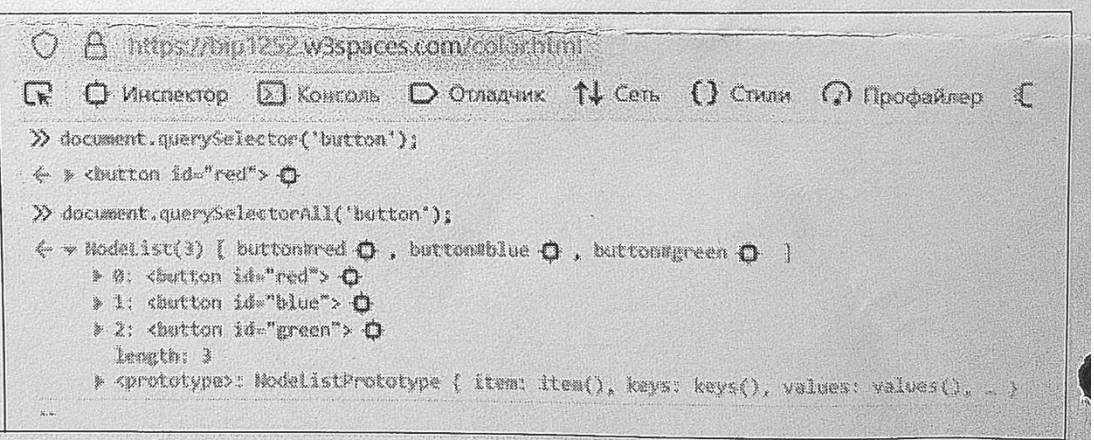
```
<form>
    <input autofocus id="name" placeholder="Name" type="text">
    <input type="submit">
</form>
```



input field where the user can type in some text

```
<script>
    document.addEventListener('DOMContentLoaded', function()
{document.querySelector('form').onsubmit = function() {
   let name = document.querySelector('#name').value;
                    alert(`Hello, ${name}!`);
                };
            });
 </script>
```

```
<button
id="red">Red
</button>
<button
id="blue">Blue
</button>
<button
id="green">Green
</button>
```




```
document.addEventListener('DOMContentLoaded',function(){
document.querySelector('button').onclick = Hello; }
);
document.querySelector('#red').onclick = function() {
   document.querySelector('#hello').style.color = 'red'; };
```

ES6: event-driven programming

using JS Events

Inline
```
<script>
alert("hi");
</script>
```

add Event listener ()

Traditional Event Handler

querySelector

First we need to access the button element in our JavaScript code.

```
const button = document.querySelector("button");
```

- `button` — *variable name. We store the button in theis variable to reuse it later*
- `document.querySelector` — *selects the first HTML element which the query applies to*
- `const` — *variable defination, we could use let or var instead. But const id preferred*
- `document` — *The context in which the querySelector will search for the given query. Here this is the HTML file*
- `"button"` — *same syntax as CSS selectors. you can refer to elements, ids or classes here.*

f12 - konsom

```
<html>
<head>
  <title>Hello</title>
  <script>
  function Hello(){
document.querySelector('h2').innerHTML=
                    "Hello JavaScript!";}
  </script>
</head>
<body>
<h2></h2>
<button onclick="Hello()">Click
Me!</button>
</body>
</html>
```

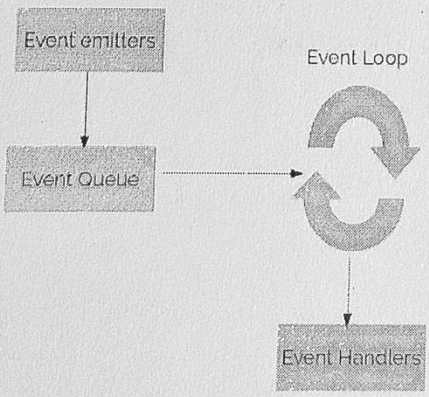
```
document.addEventListener
('DOMContentLoaded',function(){
document.querySelector('button').onclick
                    = Hello; }
);
```

```
<html>
<head>
  <title>Hello</title>
</head>
  <script>
  function Hello(){
document.querySelector('h2').innerHTML="Hello JS!";}
  </script>
 <body>
 <h2></h2>

<script>
 document.querySelector('button').onclick =
    Hello;
</script>
```

▶ Uncaught counter.htm:13
TypeError: Cannot set properties of null (setting 'onclick')
    at counter.htm:13:45

```
<button onclick="Hello()">Click Me!</button>
<script>
  document.querySelector('button').onclick = Hello;
</script>
</body>
</html>
```

```
document.querySelector('button').addEventListener('click',Hello);
```



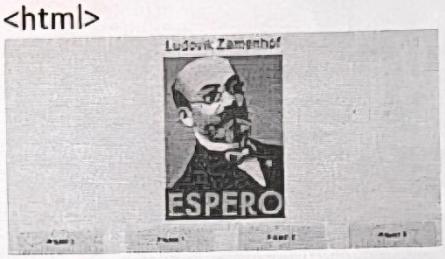

```
<button onclick="Hello()">Click
Here</button>
<script>
let counter = 0;
function Hello(){
      counter++;
      const heading=
        document.querySelector('h1');
      heading.innerHTML = counter;
  }
  </script>
```

```
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta name="robots" content="noindex">
```

Web Spider

```
<style>
        #container {
           display: flex;
           flex-wrap: wrap;
           align: center;
        }
        #container > div {
           background-color: gray;
           font-size: 20px;
           margin: 20px;
           padding: 20px;
           width: 200px;
           text-align: center;
         }
</style>
```

```
<body>
.....
      <div id="container">
         <div>Project 0</a></div>
         <div>Project 1</div>
         <div>Project 2</div>
         <div>Project 3</div>
      </div>

   </body>
<html>
```

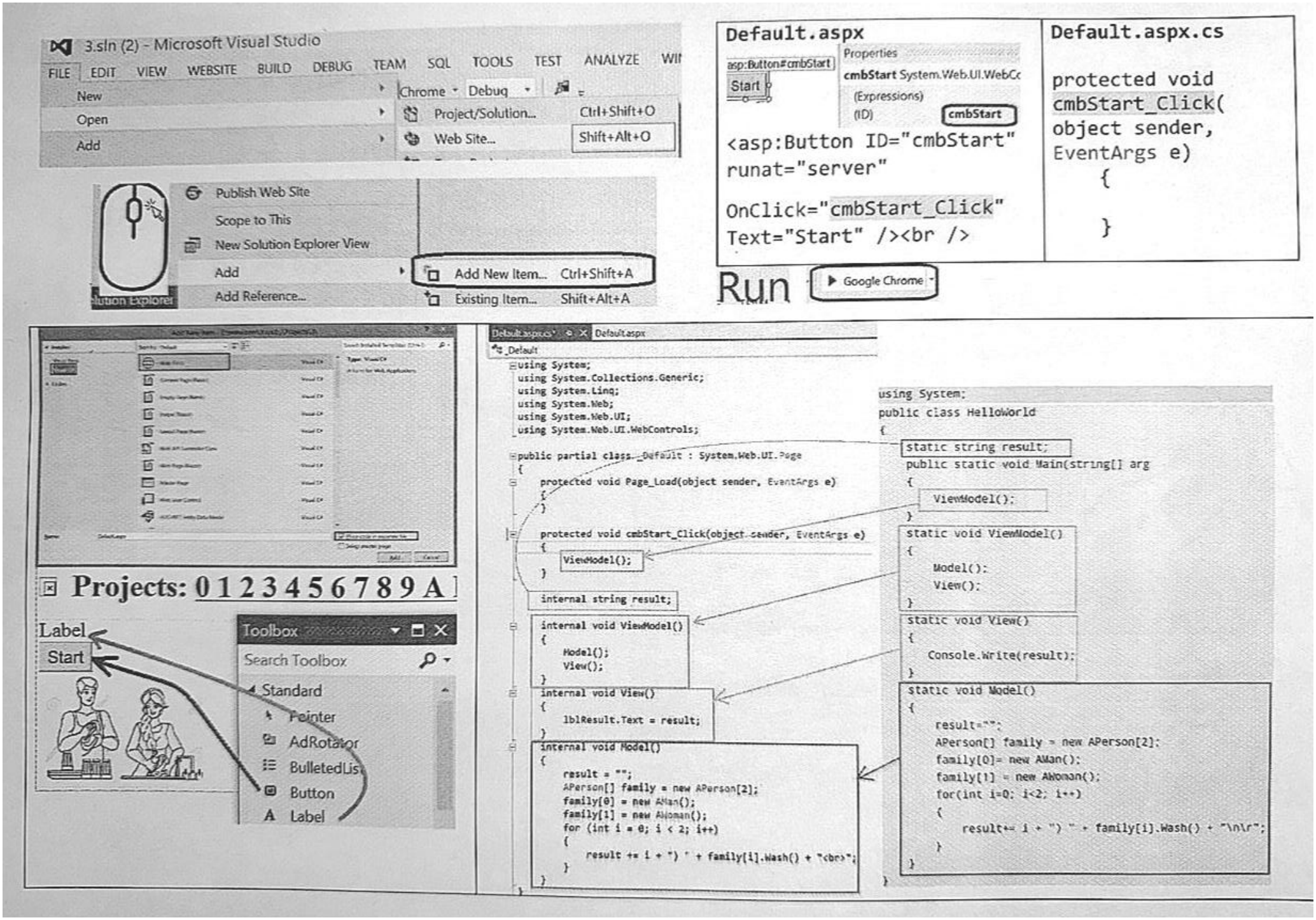


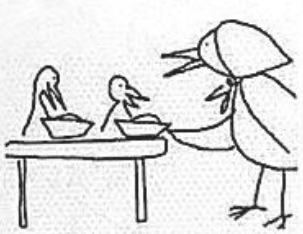
https://dot.net.by/esperanto/

```
<!– Google Font -->
  <link href="https://fonts.googleapis.com/css2?family=Poppins:wght@300;500;700&display=swap"
rel="stylesheet">
  <style>
    body {
      font-family: 'Poppins', sans-serif;
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0-alpha3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <div class="buttons">
    <a href="project1.html" class="btn btn-primary btn-project">Project 1</a>
    @media (min-width: 768px) {
      .profile-image {
        width: 200px;
        height: 200px;
      }
      .name {
        font-size: 2.5rem;
      }
}
```

to enlarge photos on large screens from 150 to 200px

The same goes for fonts


3.sln (2) - Microsoft Visual Studio
FILE EDIT VIEW WEBSITE BUILD DEBUG TEAM SQL TOOLS TEST ANALYZE
New
Open
Add
Chrome Debug
Project/Solution... Ctrl+Shift+O
Web Site... Shift+Alt+O
Publish Web Site
Scope to This
New Solution Explorer View
Add
Add Reference...
Add New Item... Ctrl+Shift+A
Existing Item... Shift+Alt+A
Default.aspx
asp:Button#cmbStart
Start
Properties
cmbStart System.Web.UI.WebCo
(Expressions)
(ID) cmbStart
<asp:Button ID="cmbStart"
runat="server"
OnClick="cmbStart_Click"
Text="Start" />
Default.aspx.cs
protected void
cmbStart_Click(
object sender,
EventArgs e)
{
}
Run
Google Chrome
Projects: 0 1 2 3 4 5 6 7 8 9 A
Label
Start
Toolbox
Search Toolbox
Standard
Pointer
AdRotator
BulletedList
Button
Label
Default.aspx
_Default
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
public partial class _Default : System.Web.UI.Page
{
protected void Page_Load(object sender, EventArgs e)
{
}
protected void cmbStart_Click(object sender, EventArgs e)
{
ViewModel();
}
internal string result;
internal void ViewModel()
{
Model();
View();
}
internal void View()
{
lblResult.Text = result;
}
internal void Model()
{
result = "";
APerson[] family = new APerson[2];
family[0] = new AMan();
family[1] = new AWoman();
for (int i = 0; i < 2; i++)
{
result += i + ") " + family[i].Wash() + "
";
}
}
using System;
public class HelloWorld
{
static string result;
public static void Main(string[] arg
{
ViewModel();
}
static void ViewModel()
{
Model();
View();
}
static void View()
{
Console.Write(result);
}
static void Model()
{
result="";
APerson[] family = new APerson[2];
family[0]= new AMan();
family[1] = new AWoman();
for(int i=0; i<2; i++)
{
result+= i + ") " + family[i].Wash() + "\n\r";
}
}
}



Сорока-Белобока кашу варила,
Деток кормила:
- Этому дала, - Этому дала,

querySelectorAll() ⇨ [,]
NodeList

[,].forEach( );

```
>> document.querySelectorAll('button');
← ▼ NodeList(3) [ button#red , button#blue , button#green ]
    ▶ 0: <button id="red">
    ▶ 1: <button id="blue">
    ▶ 2: <button id="green">
      length: 3
    ▶ <prototype>: NodeListPrototype { item: item(), keys: keys(), values: values(), … }
```

```
document.querySelectorAll('button').forEach(function(button) {
    button.onclick = function() {
        document.querySelector('#hello').style.color =
            button.dataset.color;
}); });
```

Red Blue Green

When the page is done loading

```
<script>
    document.addEventListener('DOMContentLoaded', function() {
        document.querySelectorAll('.color-change').forEach(
            function(button) {
 button.onclick = function() {
document.querySelector('#hello').style.color = button.dataset.color;
            };
        });
    });
</script>
```

I'm going to document.querySelectorAll, looking for all of the buttons

looked for things of the particular class

HTML

```
<button class="color-change" data-color="red">Red</button>
<button class="color-change" data-color="blue">Blue</button>
<button class="color-change" data-color="green">Green</button>
```

```
<head>
  <script>
  document.addEventListener('DOMContentLoaded', () => {
      document.querySelector('#color-change').onchange = function() {
          document.querySelector('#hello').style.color = this.value;
      };
  });
  </script>
</head>
<body>
  <h1 id="hello">Hello!</h1>
  <select id="color-change">
    <option value="red">Red</option>
    <option value="blue">Blue</option>
    <option value="green">Green</option>
  </select>
</body>
```

Hello!
Red
Black
Red
Blue
Green

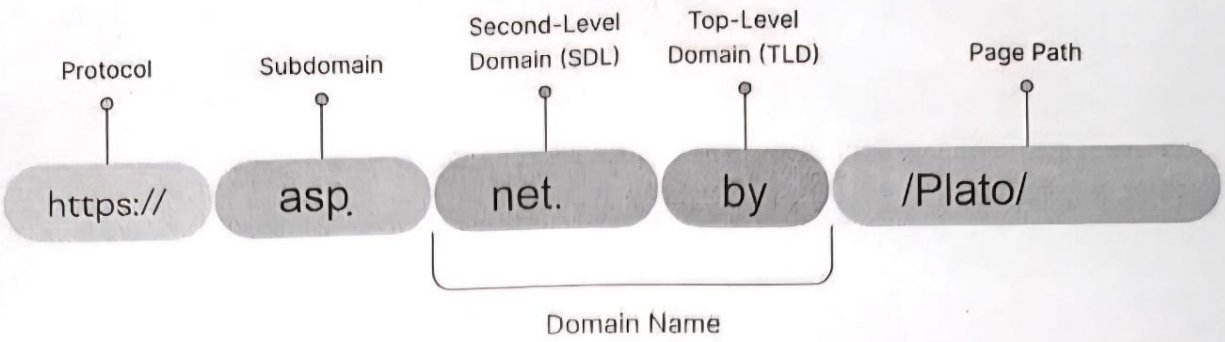
```
<button class="color-change"
data-color="red">Red</button>
```

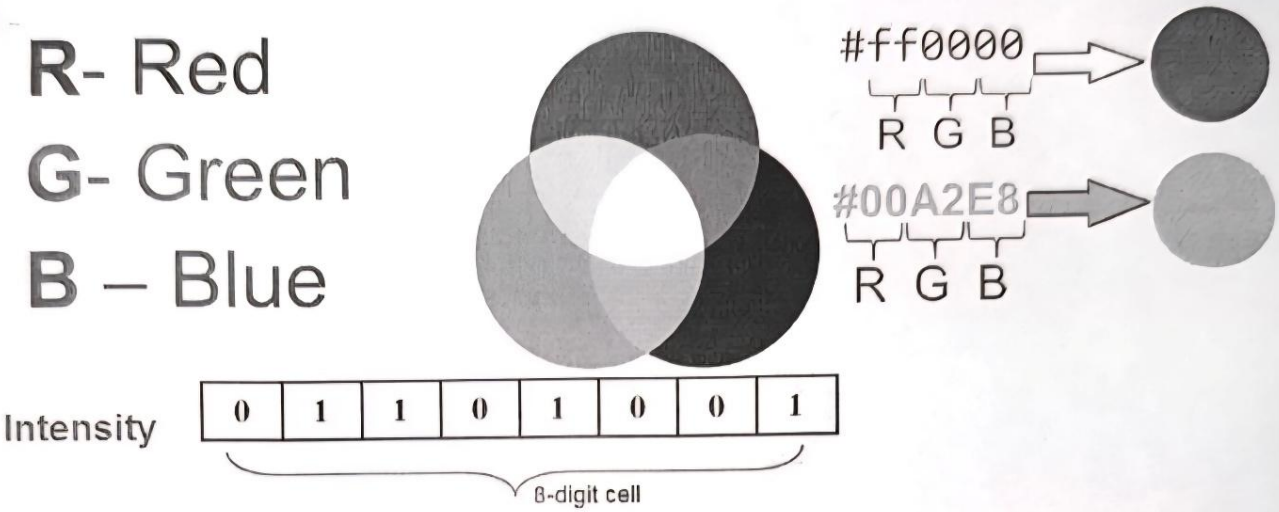
function that takes as input, the button

It's first going to pass in as input the first button (Red), then the second button (Blue), then the third button (Green)

```
function(button) {
    button.onclick = function() {

document.querySelector('#hello').
    style.color =
        button.dataset.color;
    };
```

```
function() {...}

function(but) {
but.onclick =
  function() {...}
        }

() => {...}

but => {
 but.onclick =
        () => {...}
    }
```

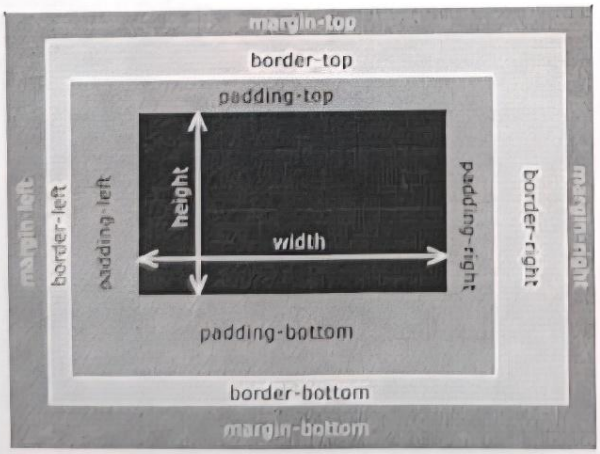




rgb(255,0,0), rgb(0,0,255), rgb(0,162,232), rgb(31,78,121)

rgb(100%,0%,0%), rgb(0%,0%,100%)

Box model




```
<style>
div
{
  color:white;
  background-color:brown;
  height: 250px; width: 100px;
  padding: 20px 11px 10px 15px;
  border: 15px solid yellow;
  margin: 120px 10px 15px 20px;
}
</style>

<div><b>Box Model</b><br>
```

```
<style>
img {
  position: absolute;
  left: 2px;
  top: 10px;
  z-index: -1; // behind the text }
</style>
```

```
class APerson
{
    virtual internal string Wash()
    {
        return "Wet and Dry";
    }
}
```

```
class AMan:APerson
{
    override internal string Wash()
    {
        return "Wet";
    }
}
```

```
class AWoman:APerson
{
    override internal string Wash()
    {
        return "Dry";
    }
}
```