

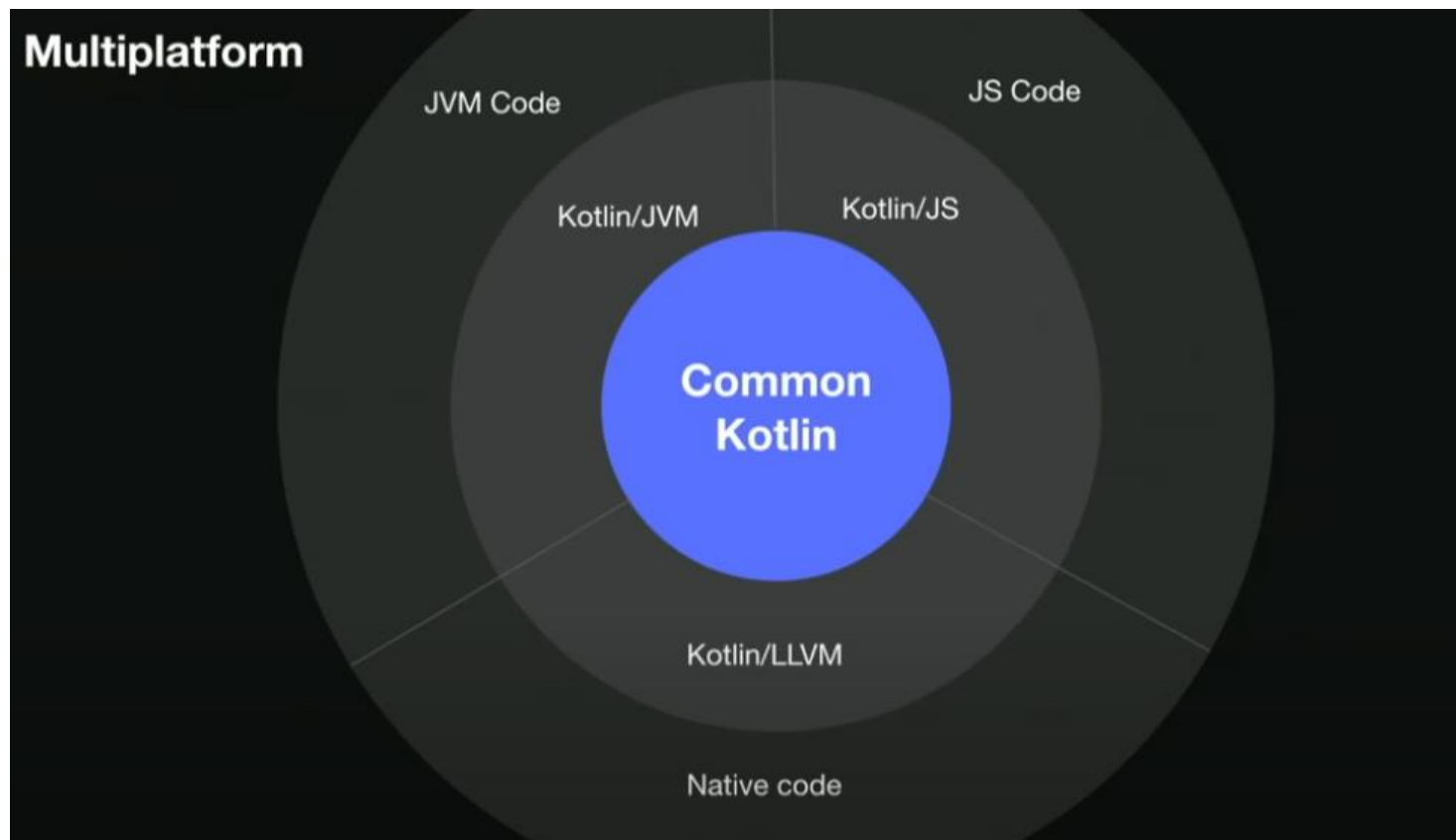
Kotlin на платформе JavaScript

Романов Владимир Юрьевич

6. KOTLIN НА ПЛАТФОРМЕ JAVASCRIPT

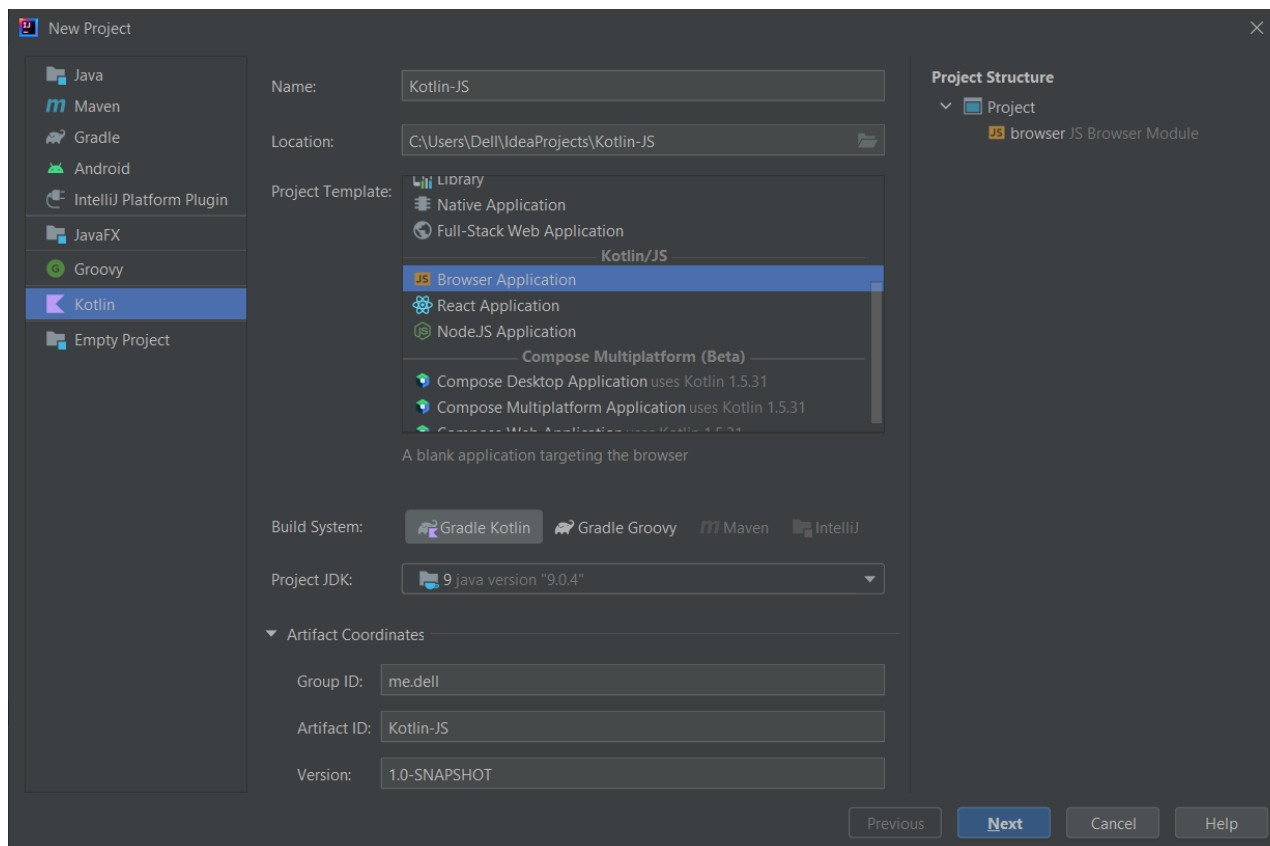
6.1 СОЗДАНИЕ ПРОЕКТА ДЛЯ ПЛАТФОРМЫ JAVASCRIPT

Платформа Kotlin/JS



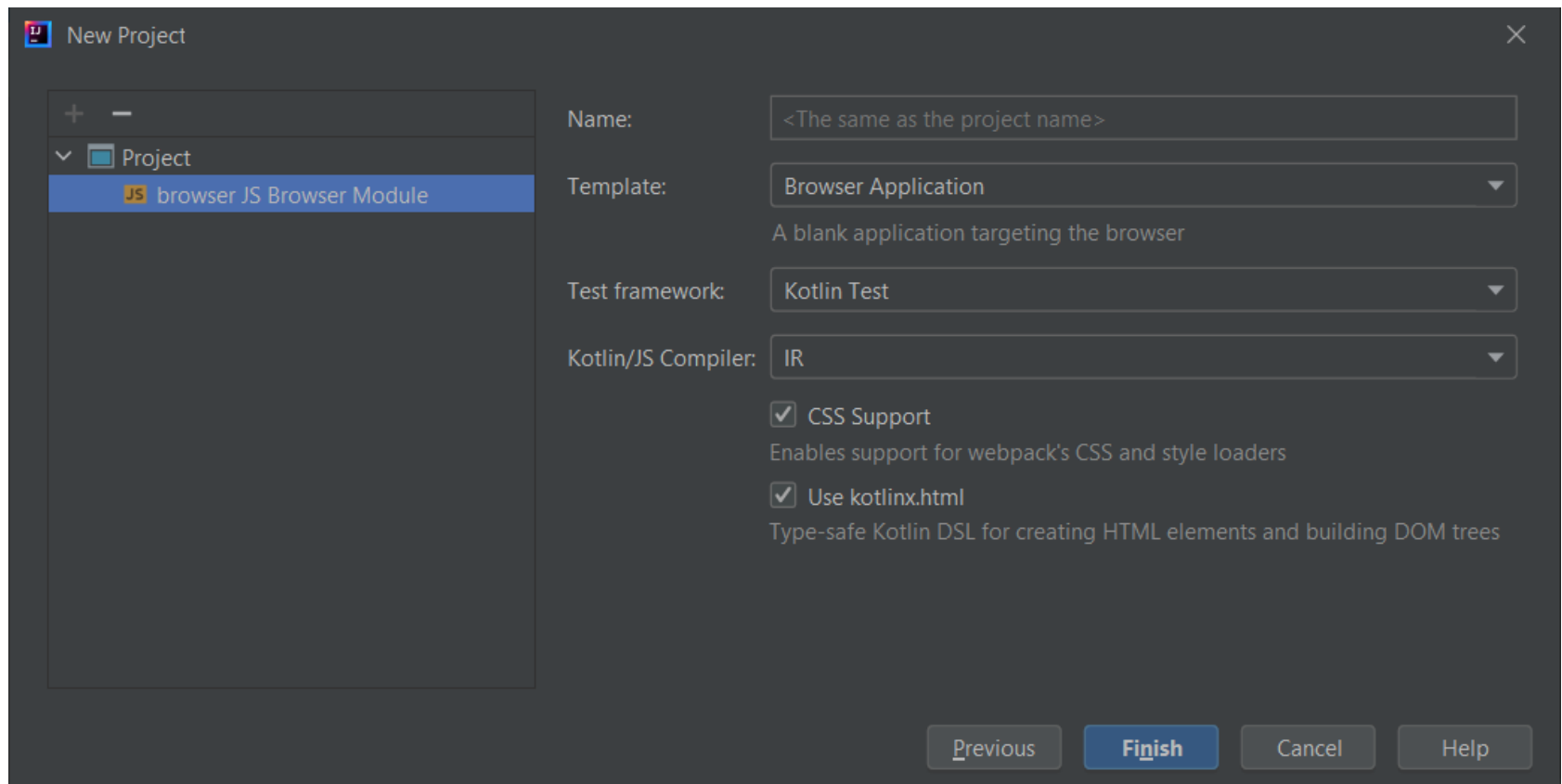
Создание проекта для платформы JavaScript

Kotlin/JS в браузере



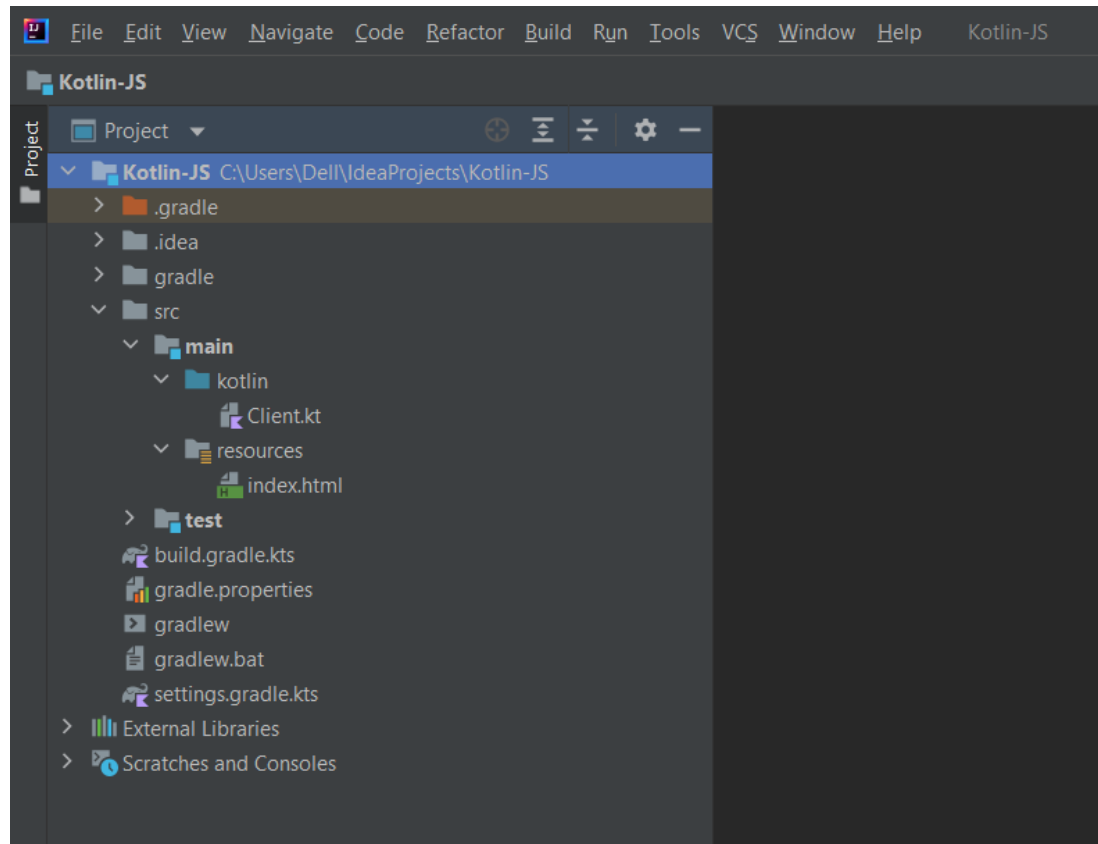
Kotlin/JS в браузере

Описание модуля Kotlin/JS в браузере



Описание модуля Kotlin/JS в браузере

Структура проекта



Структура проекта

Описание проекта *build.gradle.kts* (1)

```
plugins {  
    kotlin("js") version "1.6.0"  
}  
group = "me.dell"  
version = "1.0-SNAPSHOT"  
  
repositories {  
    mavenCentral()  
    maven("https://maven.pkg.jetbrains.space/public/p/kotlinx-  
html/maven")  
}  
dependencies {  
    testImplementation(kotlin("test"))  
    implementation("org.jetbrains.kotlinx:kotlinx-html:0.7.2")  
}
```


Описание проекта *build.gradle.kts* (2)

```
kotlin {  
    js(IR) {  
        binaries.executable()  
        browser {  
            commonWebpackConfig {  
                cssSupport.enabled = true  
            }  
        }  
    }  
}
```

Функция *kotlin* (использование)

```
plugins {  
    kotlin("js") version "1.6.0"  
}
```

Функция *kotlin* (объявление)

```
/**  
 * Configures the  
 [kotlin][org.jetbrains.kotlin.gradle.dsl.KotlinJsProjectExtension]  
 extension.  
 */  
fun org.gradle.api.Project.`kotlin`(  
    configure:  
    Action<org.jetbrains.kotlin.gradle.dsl.KotlinJsProjectExtension>  
): Unit =  
    (this as org.gradle.api.plugins.ExtensionAware)  
        .extensions.configure("kotlin", configure)
```

Функция *version*

Использование:

```
plugins {  
    kotlin("js") version "1.6.0"  
}
```

Объявление:

```
/**  
 * Specify the version of the plugin to depend on.  
 *  
 * Infix version of [PluginDependencySpec.version].  
 */  
infix fun PluginDependencySpec.version(version: String?)  
    : PluginDependencySpec = version(version)
```

Pecypc *index.html*

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
  <meta charset="UTF-8">
```

```
  <title>JS Client</title>
```

```
</head>
```

```
<body>
```

```
<script src="Kotlin-JS.js"></script>
```

```
<div id="root"></div>
```

```
</body>
```

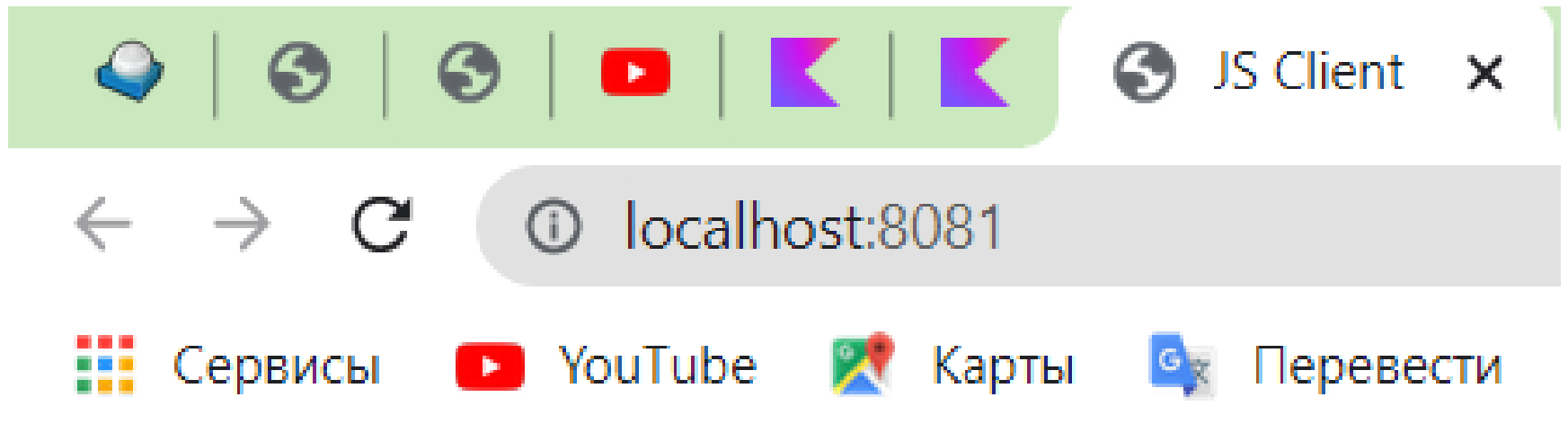
```
</html>
```

Клиент *Client.kt*

```
import kotlinx.html.div
import kotlinx.html.dom.append
import org.w3c.dom.Node
import kotlinx.browser.document
import kotlinx.browser.window

fun main() {
    window.onload = { document.body?.sayHello() }
}
fun Node.sayHello() {
    append {
        div {
            +"Hello from JS"
        }
    }
}
```

Клиент в браузере



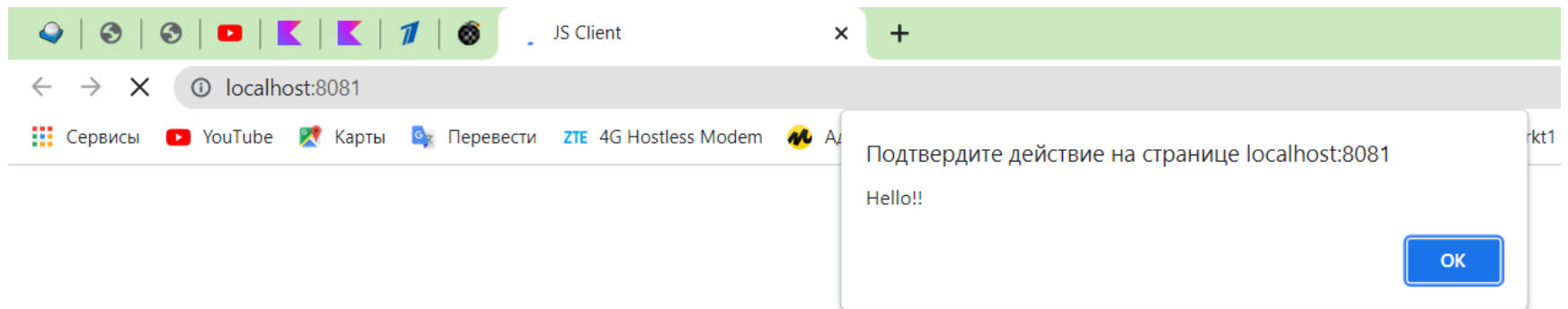
Hello from JS

Клиент в браузере

Взаимодействие с Document Object Model (DOM)

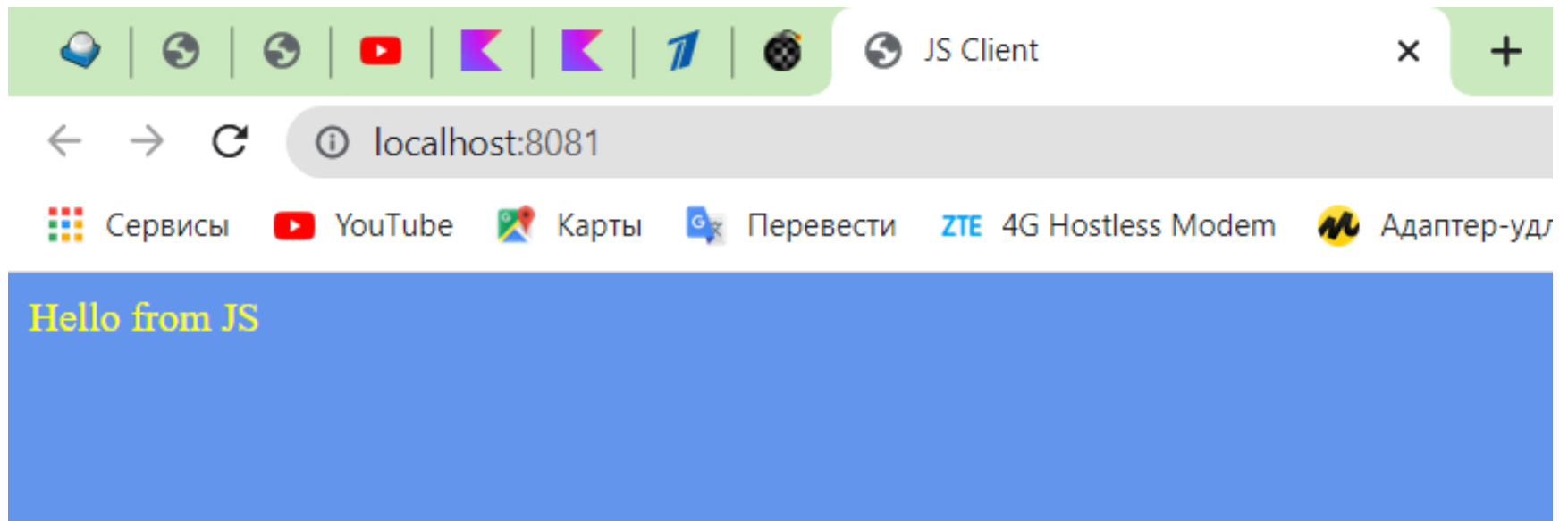
```
fun main() {  
    document.backgroundColor = "6495ED"  
    document.fgColor = "yellow"  
    window.alert("Hello!!")  
  
    window.onload = { document.body?.sayHello() }  
}
```


Выдача сообщения



Выдача сообщения

Раскраска текста и фона страницы

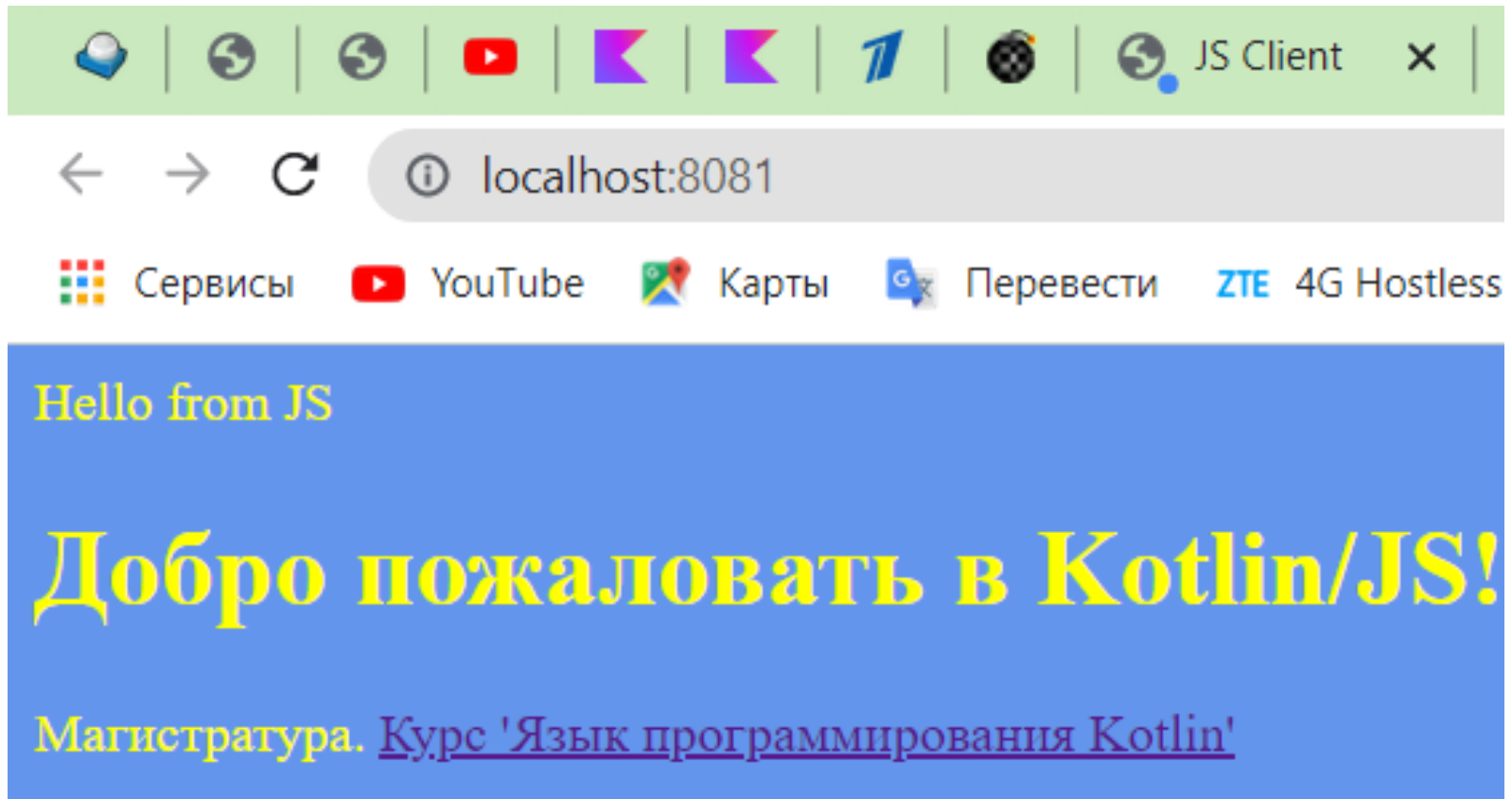


Раскраска текста

Использование типизированного *HTML*

```
window.onload = {  
  document.body?.sayHello()  
  document.body!!.append.div {  
    h1 {  
      +"Добро пожаловать в Kotlin/JS!"  
    }  
    p {  
      +"Магистратура. "  
      a("http://master.cmc.msu.ru/") {  
        +"Курс 'Язык программирования Kotlin'"  
      }  
    }  
  }  
}
```

Использование типизированного *HTML* (в браузере)



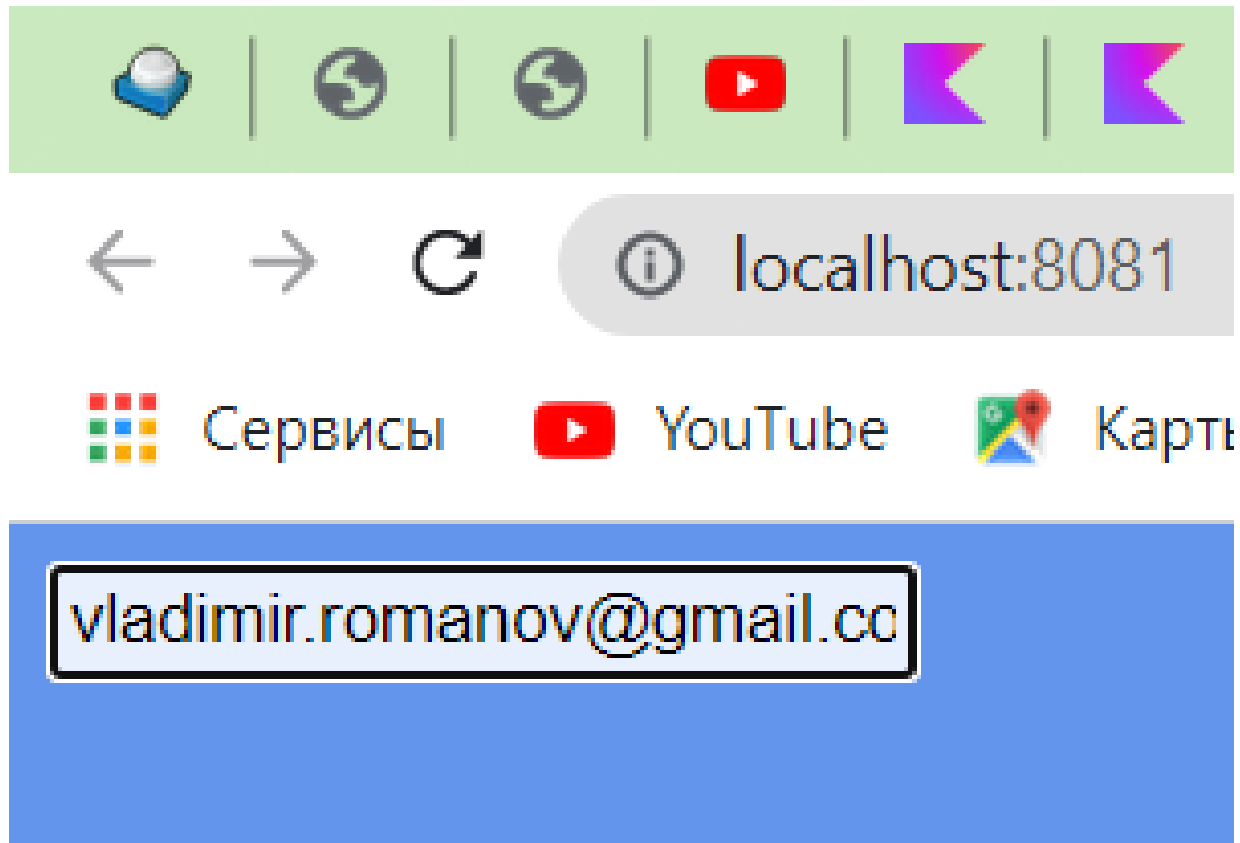
Использование типизированного *HTML*

Использование управляющих элементов

```
// <input type="text" name="email" id="email"/>
window.onload = {
    document.body!!.append.div {
        input {
            type = InputType.text
            name = "email"
            id = "email"
        }
    }
}

val email = document.getElementById("email") as
HTMLInputElement
email.value = "vladimir.romanov@gmail.com"
```

Использование управляющих элементов (в браузере)



Использование управляющих элементов

Изменение содержимого элемента *HTML* (1)

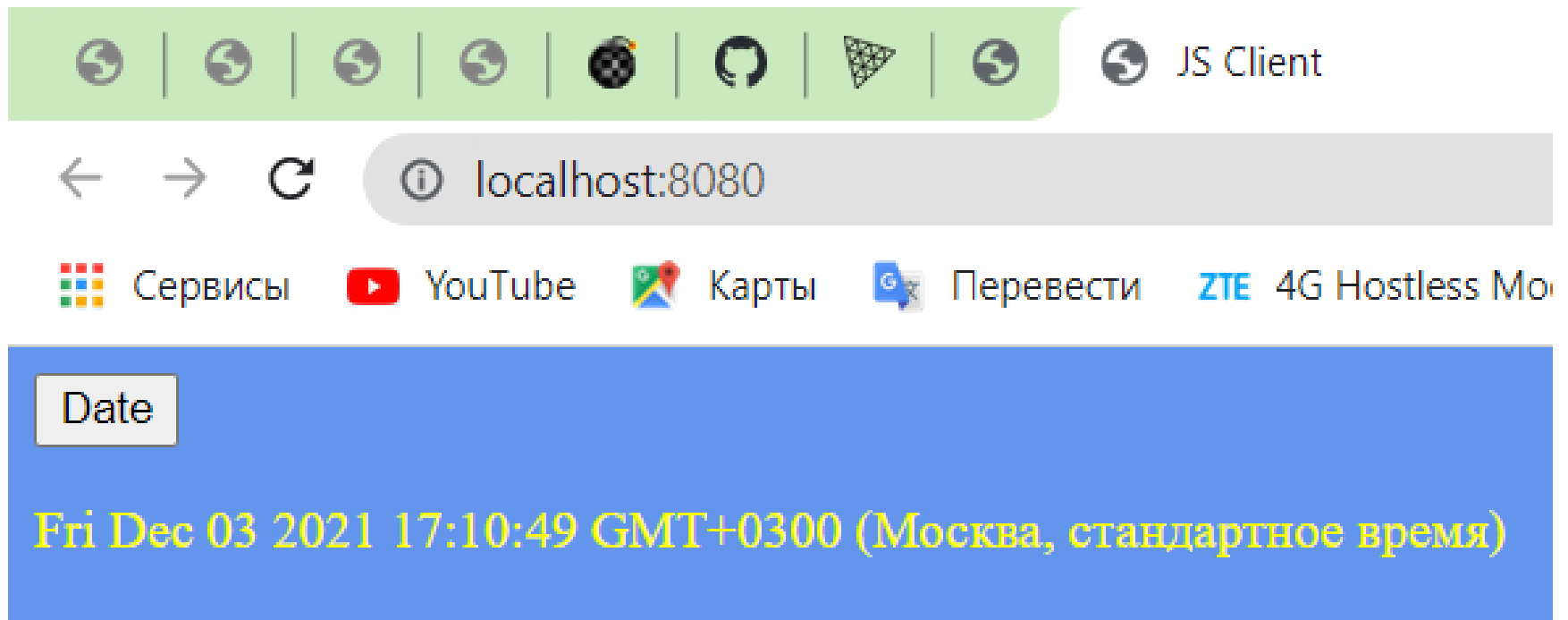
При нажатии на кнопку должна показываться текущая дата

```
fun main() {  
    document.bgColor = "6495ED"  
    document.fgColor = "yellow"  
  
    window.onload = {  
        document.body?.addDateButton()  
    }  
}
```

Изменение содержимого элемента *HTML* (2)

```
fun Node.addDateButton() {  
    append {  
        div {  
            button {  
                +"Date"  
                type = ButtonType.button  
                onClickFunction = { sayDate() }  
            }  
            p { id = "demo" }  
        }  
    }  
}  
  
fun sayDate() {  
    document.getElementById("demo").innerHTML  
        = Date().toString()  
}
```


Изменение содержимого элемента *HTML* (в браузере)



Изменение содержимого элемента *HTML*. Текущая дата

Изменение атрибута элемента *HTML* (1)

При нажатии на кнопку должна включаться/выключаться лампа

```
fun main() {  
    document.backgroundColor = "6495ED"  
    document.fgColor = "yellow"  
  
    window.onload = {  
        document.body?.addToggleBulbButton()  
    }  
}
```

Изменение атрибута элемента *HTML* (2)

```
fun Node.addToggleBulbButton() {  
  append {  
    div {  
      button {  
        +"Toggle Bulb"  
        type = ButtonType.button  
        onClickFunction = { toggleBulb() }  
      }  
      p {  
        img {  
          id = "bulb"  
          src = "pic_bulboff.gif"  
        }  
      }  
    }  
  }  
}
```

Изменение атрибута элемента *HTML* (3)

```
// ...
```

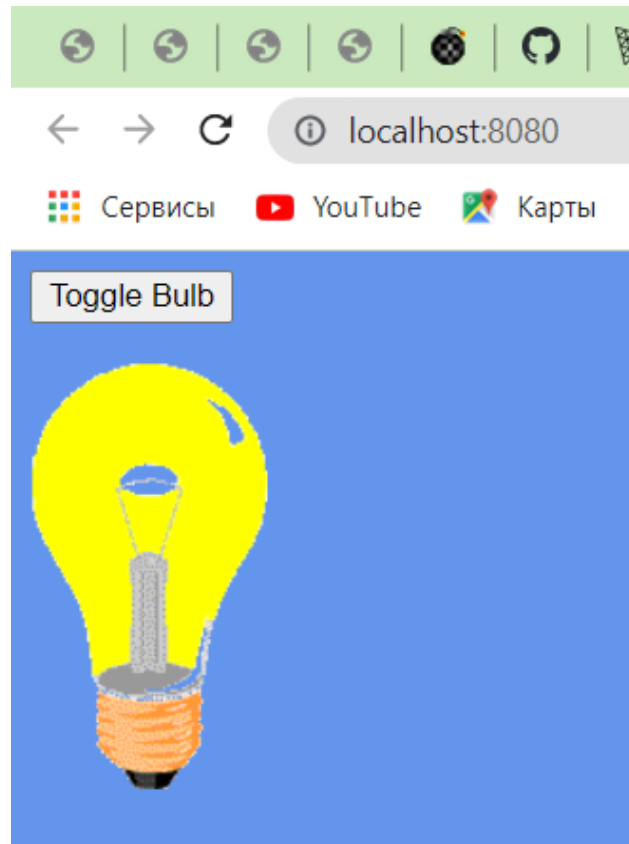
```
    p {  
        img {  
            id = "bulb"  
            src = "pic_bulboff.gif"  
        }  
    }
```

```
// ...
```

```
var isOn = false
```

```
fun toggleBulb() {  
    val bulbImage = document.getElementById("bulb") as Image  
    bulbImage.src = if (isOn) "pic_bulboff.gif" else "pic_bulbon.gif"  
    isOn = !isOn  
}
```

Изменение атрибута *HTML* (в браузере)



Изменение атрибута *HTML*. Смена изображения

Изменение стиля элемента *HTML* (1)

При нажатии на кнопку должен изменяться стиль текста

```
fun main() {  
    document.backgroundColor = "6495ED"  
    document.fgColor = "yellow"  
  
    window.onload = {  
        document.body?.addToggleButton()  
    }  
}
```

Изменение стиля элемента *HTML* (2)

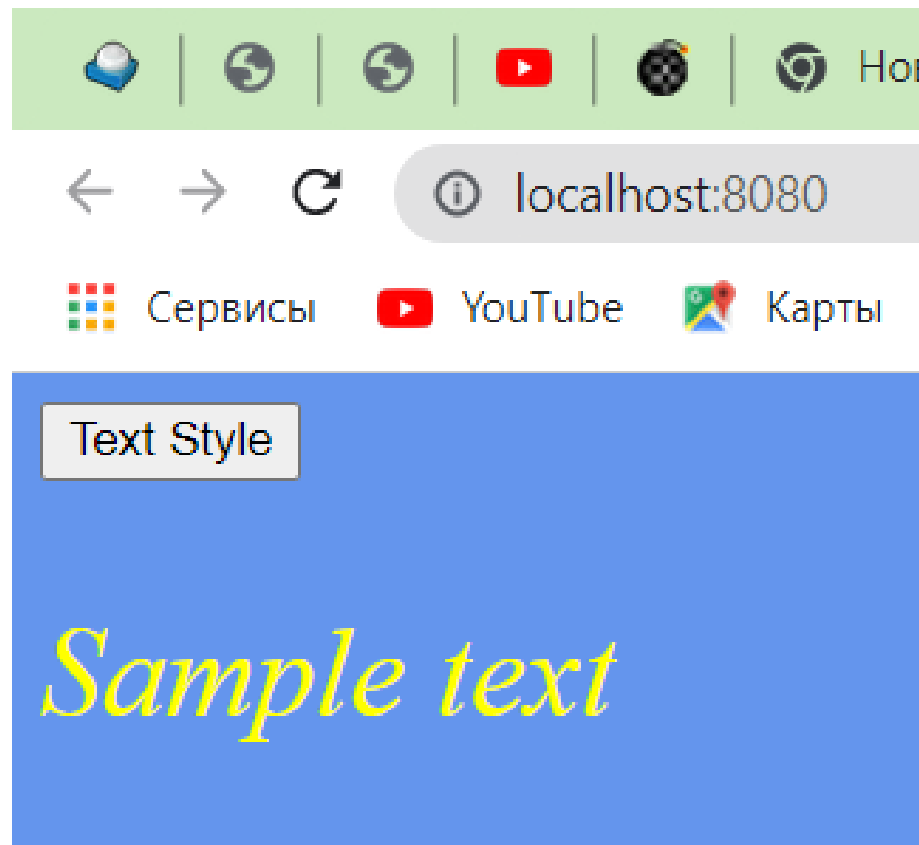
```
fun Node.addToggleStyleButton() {  
    append {  
        div {  
            button {  
                +"Text Style"  
                type = ButtonType.button  
                onClickFunction = { toggleStyle() }  
            }  
            p { id = "demo"  
                +"Sample text " }  
        }  
    }  
}
```

Изменение стиля элемента *HTML* (3)

```
var isItalic = false
```

```
fun Node.toggleStyle() {  
    val text = document.getElementById("demo") as  
    HTMLParagraphElement  
    text.style.fontSize = "35px"  
    text.style.fontStyle = if (isItalic) "italic" else "normal"  
    isItalic = !isItalic  
}
```


Изменение стиля *HTML* (в браузере)



Изменение стиля *HTML*

События от мыши (1)

При нажатии на кнопку мыши должен изменяться цвет текста и его фона

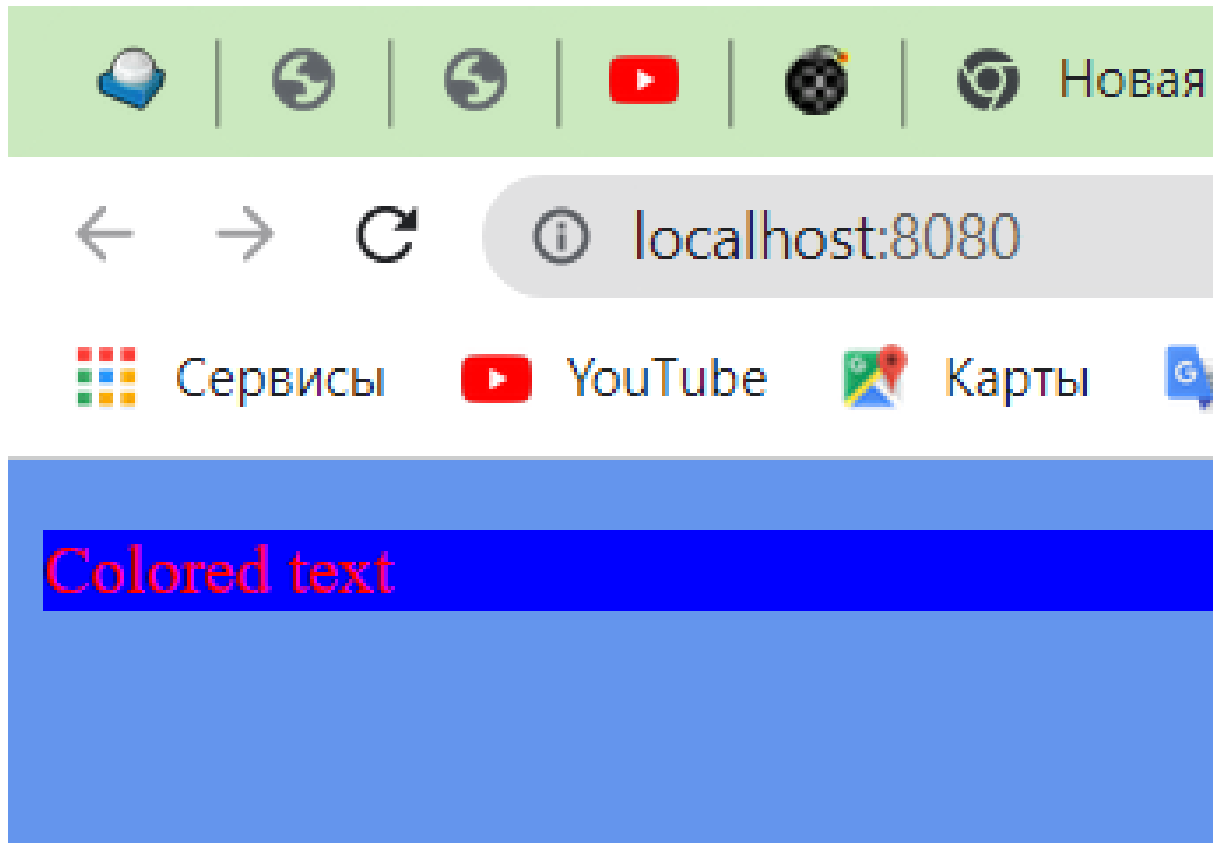
```
fun Node.addMouseDownButton() {  
    append {  
        p {  
            +"Colored text "  
            id = "demo"  
            onMouseDownFunction = { paint(id, "red", "blue") }  
            onMouseUpFunction = { paint(id, "green", "yellow") }  
        }  
    }  
}
```

События от мыши (2)

При нажатии на кнопку мыши должен изменяться цвет текста и его фона

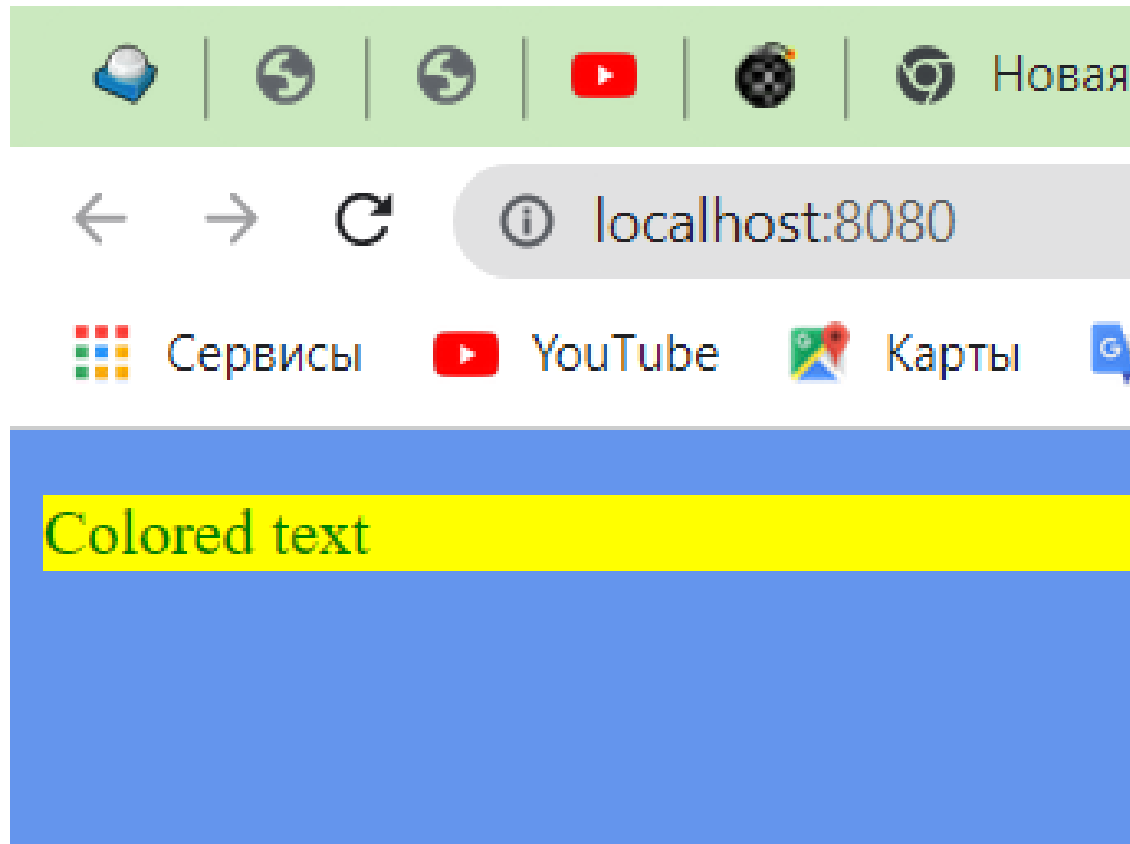
```
fun paint(id: String, color: String, background: String = "White") {  
    val text = document.getElementById(id)  
        as HTMLParagraphElement  
    text.style.color = color  
    text.style.background = background  
}
```

События от мыши (нажатие)



События от мыши (нажатие)

События от мыши (отпускание)



События от мыши (отпускание)

Выбор элемента мышью(1)

При нажатии на кнопку мыши выводятся имя тега и идентификатор элемента

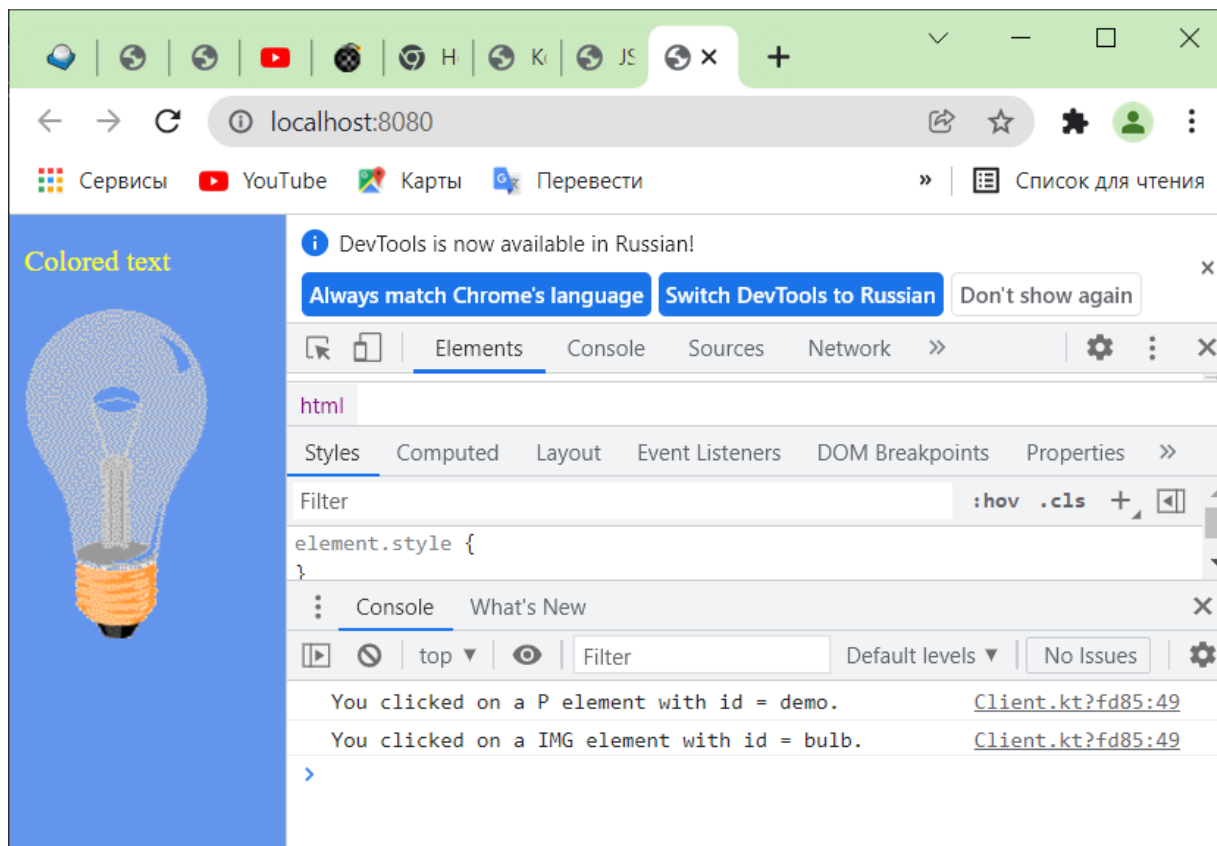
```
fun HTMLElement.addMouseSelectButton() {  
    onmousedown = { mouseSelection(it) }  
    append {  
        p {  
            +"Colored text "  
            id = "demo"  
        }  
        img {  
            id = "bulb"  
            src = "pic_bulboff.gif"  
        }  
    }  
}
```

Выбор элемента мышью(2)

При нажатии на кнопку мыши выводятся имя тега и идентификатор элемента

```
fun mouseSelection(e: MouseEvent) {  
    val target: EventTarget = e.target ?: return  
  
    when (target) {  
        is HTMLElement -> {  
            val tagName = target.tagName  
            val id = target.id  
            console.info("You clicked on a $tagName element with id =  
$id.");  
        }  
    }  
}
```

Вывод выбора элемента мышью на консоль



При нажатии на кнопку мыши выводятся имя тега и идентификатор элемента